

PhantomSDR-Plus

Potpuna dokumentacija

16 dokumenata · izrađeno 2026-10-03

Sadržaj

1. README	8
Ključne značajke	8
Podržani hardver	10
Brzi početak	11
Mjerenja performansi	12
Upotreba	12
Konfiguracija	13
Donji prag slapa — waterfall.sh	14
Prilagodba	14
Dokumentacija	16
Mrežni popis WebSDR-ova	16
Rješavanje problema	16
Doprinosi	17
Licenca	17
Autori i zasluge	17
Poveznice	17
Podrška	17
Savjeti za performanse	18
2. Vodič za instalaciju	19
Sadržaj	19
Sistemske zahtjevi	19
Priprema prije instalacije	20
Instalacija ovisnosti	21
Instalacija Node.js-a i npm-a	22
Instalacija OpenCL-a (neobavezno, ali preporučeno)	23
Instalacija PhantomSDR-Plusa	25
Instalacija audiokodeka Opus	30
Autorun Spot Reporter (FT8/FT4/WSPR)	31
Emulacija KiwiSDR klijenata (neobavezno)	33
Konfiguracija	33
Postavljanje ovisno o SDR uređaju	36
Testiranje i provjera	39
Postavljanje automatskog pokretanja	40
Toplinska zaštita procesora	42
Rješavanje problema	44
Optimizacija performansi	47
Ažuriranje PhantomSDR-Plusa	48
Sigurnosna kopija i vraćanje	51
Sigurnosna razmatranja	51
Kako doći do pomoći	52
Dodatak A: potpuni popis ovisnosti	52
Dodatak B: primjeri konfiguracije	53
3. Postavljanje administratorske ploče	55
Pregled	55
Što vam administratorska ploča donosi	55

Preduvjeti	56
Korak 1 — Datoteke	56
Korak 2 — Pokrenite skriptu za postavljanje	56
Korak 3 — Pokretanje / zaustavljanje / ponovno pokretanje	57
Korak 4 — Otvorite port na vatrozidu	59
Korak 5 — Značajka Kick Users	60
Konfiguracija — admin_config.json	60
Postavka sdr_host	61
proxy.py — što je i kada vam treba	61
Promjena portova nakon postavljanja	62
Korisne ručne naredbe	62
Datoteke zapisnika	63
Sažetak pristupa	65
Brisanje poruka razgovora	65
Emitirane poruke na slapu	65
Grafikoni sustava	66
Thermal Guard	67
Spot Reporting (autorun FT8/FT4/JS8/WSPR)	70
Rješavanje problema	75
4. Thermal Guard	78
Sadržaj	78
1. Što radi	78
2. Brzi početak	79
3. Kako odlučuje	79
4. Četiri načina rada	80
5. Odabir načina rada	82
6. Referenca postavki	83
7. Rad bez administratorske ploče	83
8. Uključivanje throttle faze bez roota	84
9. Rad bez nadzora	85
10. Isprobajte ga prije nego mu povjerite stroj	86
11. Čitanje dnevnika	87
12. Lockout i ograničenje broja zaustavljanja	87
13. Rješavanje problema	88
14. Isključivanje ili uklanjanje	88
Vidi također	88
5. Ograničenja veza	89
Sadržaj	89
1. Od čega se zapravo branite	89
2. Tri sloja	90
3. Ograničenja po adresi	90
4. Neaktivne veze	91
5. Zaštita u jezgri	92
6. Referenca postavki	93
7. Što vidi odbijeni slušatelj	93
8. Čitanje zapisa i brojača	94
9. Odabir vaših brojeva	95
10. Zamke koje vrijedi poznavati	95
11. Isključivanje svega	96

6. Više prijemnika	97
Sadržaj	97
1. Kako radi	97
2. Dva načina objave prijemnika	98
3. Dodavanje drugog prijemnika	98
4. receivers.toml	100
5. Pokretanje i zaustavljanje	100
6. Što vide slušatelji	101
7. S-metar iznad 30 MHz	101
8. Sigurnost	102
9. Koliko prijemnika stane	102
10. Zamke koje treba znati	103
7. Poslužitelj statistike	104
Pregled	104
Preduvjeti	104
Metode instalacije	104
Konfigurirajte svoju PhantomSDR aplikaciju	107
Testiranje instalacije	109
Upravljanje uslugom	110
Ručna instalacija (Opcija 2)	110
Promjena porta nakon instalacije	112
Rješavanje problema	112
Što ćete vidjeti	115
Otkrivanje temperature	115
Sažetak značajki	116
Sigurnosne napomene	116
Podrška i resursi	116
8. Dekoderi	119
Sadržaj	119
1. Kako pokrenuti dekodier	119
2. FT8	122
3. FT4-FT2	123
4. JS8	124
5. CW — Morseov kod	125
6. QRSS Grabber	126
7. WSPR	129
8. HF FAX / WEFAX	130
9. NAVTEX	131
10. FSK / RTTY — uključujući PSK31 i Oliviju	132
11. SSTV	136
12. Opći savjeti	138
9. RADE README	139
Instalacija — kraći put	139
Sadržaj	139
Kako to radi	140
Pregled arhitekture	141
RADEL naspram RADEU	141
Preduvjeti	142

Ručna instalacija	142
Postavljanje zakrpanih datoteka	142
Upravljanje sidecarom	143
Vrata 8074	144
Korištenje RADE-a u pregledniku	144
Provjera i otklanjanje pogrešaka	145
Varijable okruženja	146
Promijenjene datoteke	147
Sažetak toka signala	147
Rješavanje problema	148
Mjerenje istodobnosti na vlastitom hardveru	150
Ažuriranje RADE v1	152
10. RADE opća instalacija (Linux)	155
Korak 1 — Sistemski paketi	155
Korak 2 — Python paketi	156
Korak 3 — Kloniranje i izgradnja repozitorija radae	157
Korak 4 — Provjera lanca dekodiranja	157
Korak 5 — Izgradnja sučelja PhantomSDR-Plusa	159
Korak 6 — Upravljanje sidecarom	159
rade.sh i dalje dolazi uz projekt za samostalnu upotrebu — pokretanje sidecara bez PhantomSDR-Plus pokretača ili njegovo zasebno testiranje. Njegove naredbe, dnevnik i pripadajuća upozorenja o čuvanju nalaze se u Upravljanje sidecarom.	159
Korak 7 — Otvorite port 8074	159
Korak 8 — Pokrenite poslužitelj	160
Korak 9 — Pokrenite RADE	160
Korak 10 — Korištenje RADE-a u pregledniku	161
Napomena o procesoru	161
Buduće ažuriranje RADE-a	162
11. Uređivanje varijanti	163
1. Što su varijante	163
2. Koja datoteka što radi	164
3. Izmjena koja utječe samo na neke varijante	166
4. Gdje su prekidači varijanti	166
5. Ostale postavke koje vrijedi znati	167
6. Nakon uređivanja — ponovna izgradnja	168
7. Kako provjeriti izmjenu prije nego joj povjerujete	168
12. Korisnički vodič	170
Sadržaj	170
Uvod	170
Početak rada	171
Pregled sučelja	172
Osnovne radnje	173
Napredne značajke	176
Načini demodulacije	179
Digitalni dekoderi	182
Tipkovni prečaci	185
Oznake	186
Upotreba na mobilnim uređajima	187
Savjeti i dobre prakse	188

Rješavanje problema	189
Često postavljana pitanja	190
Izvori	191
Dodatak: uobičajene frekvencije	192
Pojmovnik	193
13. Emulacija KiwiSDR klijenta	194
Sadržaj	194
1. Što dobiva Kiwi klijent	194
2. Instalacija mosta	195
3. Uključivanje	195
4. Pregled postavki	196
5. Spajanje klijenta	196
6. Razina zvuka	196
7. S-metar	197
8. Slap i spektar	198
9. Ako nešto ne izgleda dobro	199
10. Dokle je provjereno	199
14. Diverziti prijam	200
Što radi, a što ne	200
Brzi početak	200
Korištenje WebSDR-a	202
Čitanje ploče	203
Remote SNR trim	204
Odabir drugog prijamnika	205
Rješavanje problema	205
Ograničenja	206
15. Upravljanje primopredajnikom	207
Što vam treba	207
Podržani prijemnici	207
Što radi, a što ne	208
Brzi početak	208
Odabir načina povezivanja	209
Postavke serijskog porta	210
Sinkronizacija	210
Načini rada	211
Širina filtra	212
Utišavanje pri odašiljanju	212
Pomak frekvencije	212
Izbornik Rig	213
Linux	213
Windows	213
TCI-CAT na stranici prijemnika	213
Za operatere prijemnika	222
Za programere: sučelje stranice	223
Rješavanje problema	224
Poznata ograničenja	224
16. Struktura projekta	225
Sadržaj	225

Stablo direktorija	226
Korijenski direktorij	239
Izvorni kod (src/)	242
Sučelje (frontend/)	243
Popisi frekvencija (frequencylist/)	247
Konfiguracijske datoteke	248
Sustav izgradnje	249
Ovisnosti datoteka	250
Tok podataka	250
Vodič za izmjenu datoteka	251
Važne putanje	251
Uobičajene radnje s datotekama	252
Upravljanje verzijama	252

README

README.md

PhantomSDR-Plus

Unaprijeđeni WebSDR poslužitelj otvorenog koda s naprednim mogućnostima i futurističkim dizajnom

PhantomSDR-Plus je fork projekta PhantomSDR koji pruža web-poslužitelj softverski definiranog radija (SDR) visokih performansi, sposoban posluživati stotine istodobnih korisnika. Nudi poboljšano korisničko sučelje, podršku za više dekodera, prikaz plana pojaseva i kompatibilnost s raznim SDR hardverskim platformama.

Ključne značajke

Performanse i skalabilnost

- **Podrška za više korisnika:** stotine istodobnih korisnika, ovisno o hardveru
- **Visoka frekvencija uzorkovanja:** podrška za SDR uređaje do 70 MSPS (realno) / 35 MSPS (IQ)
- **Hardversko ubrzanje:** podrška za OpenCL i CUDA za obradu ubranu GPU-om
- **Optimizirano emitiranje:** FLAC i Opus audiokompresija niske latencije. Preglednik dekodira svaki FLAC paket u trenutku kad stigne, a njegov se spremnik reprodukcije prilagođava vezi slušatelja i drži samo onoliko zvuka koliko treba — obično 100–150 ms od prijamnika do zvučnika na telefonu preko Wi-Fi

Korisničko sučelje

- **Futuristički dizajn:** moderno, prilagodljivo web-sučelje
- **Optimizirano za mobilne uređaje:** poboljšano mobilno sučelje za slušanje u pokretu
- **Prikaz plana pojaseva:** interaktivni slap (waterfall) s preklopima frekvencijskih pojaseva
- **Prilagodljive palete boja:** više shema boja za slap
- **Dvostruki S-metar:** izbor između analognog i digitalnog prikaza signala
- **Tipke dekodera na jedan pritisak:** red Decoders na glavnoj ploči, odmah ispod izbornika načina rada (FT8, FT4, FT2, JS8, CW, WSPR, FAX, SSTV, NAVTX, RTTY), uz RADEL i RADEU pokraj samog izbornika načina rada te ponovljene u skočnim prozorima načina rada i pojaseva. Jedan pritisak pokreće dekodera i otvara njegov prozor, drugi ga zaustavlja. Dosadašnji red za širinu pojasa je uklonjen.
- **Skener kanala:** pretražuje pojas, ili točno ono što prikazuje slap, i zaustavlja se na prvom kanalu koji se izabranim brojem dB uzdigne iznad šuma samoga pojasa, a zatim nastavlja sam čim taj kanal utihne. Korak prati način rada, a zaustavljanja padaju na mrežu kanala (1 kHz SSB, 100 Hz CW, 5 kHz kratkovalni AM, 9/10 kHz srednji val, 9 kHz dugi val). Način *Preskoči prazne* čita spektar slapa i skoči ravno na signale, a tipka za zaključavanje izbacuje stalno zauzet kanal iz pretraživanja. Sve se odvija lokalno u pregledniku slušatelja, pa pretraživanje nikome drugom ne pomiče prijamnik (vidi [Korisnički priručnik](#))
- **Diverziti prijam:** povezuje prijamnik s drugim na nekoj drugoj lokaciji i reproducira onaj koji trenutačno ima bolji signal, pa propadanje na jednoj lokaciji pokriva druga. Drugi prijamnik može biti još jedan PhantomSDR-Plus, KiwiSDR, UberSDR ili WebSDR; prva tri ne traže ništa osim preglednika

slušatelja, a samo WebSDR treba mali relej na vašem vlastitom poslužitelju. Poravnanje traje otprilike petnaest sekundi, a udaljeni SNR ugađanje uravnotežuje dvije postaje jednu prema drugoj. Svaki slušatelj u vlastitom pregledniku drži imenovani popis drugih prijarnika, redoslijedom kakav želi, i može ga izvesti u datoteku te ponovno uvesti (vidi [Diverziti prijam](#))

- **Više prijarnika, jedna stranica:** jedno računalo može istodobno pokretati dva ili više prijarnika — na primjer RX-888 za kratki val i RTL-SDR za 2 m — a slušatelji prelaze s jednog na drugi gumbima u zaglavlju stranice. Svaki prijarnik zadržava vlastiti vodopad, chat, oznake, popis slušatelja i podatke o stanici, a iznad 30 MHz S-metar pokazuje VHF S-jedinice ($S9 = -93$ dBm) (vidi [Više prijarnika](#))

Obrada signala

- **Više načina demodulacije:** AM, FM, USB, LSB, CW i drugi. Također je implementiran RADE dekodirer verzije 1.
- **Sinkrona AM detekcija:** bolja kvaliteta AM prijema
- **Smanjenje šuma:** NR (spektralno), NB (prigušivač impulsnog šuma), NS (potiskivanje pozadinskog šuma) i AN (automatski notch) — svi na putu slušanja, nikada na dekodirima
- **AGC mogućnosti:** više načina automatske regulacije pojačanja
- **Automatski squelch:** automatski prag squelcha na temelju šuma
- **CTCSS squelch podtonom:** FM zvuk prolazi samo dok se prima odabrani podton — ili bilo koji standardni; ton se bira u malom prozoru pokraj gumba CTCSS, koji ga zatim prikazuje
- **Smanjenje šuma umjetnom inteligencijom:** neuronska mreža RNNoise u pregledniku slušatelja, uklanja šum pojasa iz govora (samo govorni načini; vidi [Korisnički priručnik](#))

Napredne značajke

- **Digitalni dekodireri:** FT8, FT4, FT2, JS8, CW, QRSS Grabber, WSPR, HF FAX, SSTV, NAVTEX, FSK/RTTY, PSK31, Olivia i FreeDV RADE, svaki radi u vlastitoj pozadinskoj dretvi pa dekodiranje nikada ne prekida zvuk (vidi [Dekodireri](#))
- **Decoder ID:** imenuje digitalni način rada u propusnom pojasu i jednim klikom nudi odgovarajući dekodirer, na temelju izmjerene širine pojasa, razmaka tonova, simbolske brzine i vremenskog rasporeda emisija, uzimajući frekvenciju u obzir kao dodatni pokazatelj — što ujedno razdvaja **FT8 od JS8**, dva načina rada s istovjetnim signalom. Samo predlaže, nikada ne prebacuje sam od sebe, i radije šuti nego nagađa kada je signal preslab. Slabljenje ga ne ušutkava: kada QSB razlomi dvominutnu WSPR emisiju na komade, i dalje imenuje način rada, čitajući ritam iz cijele povijesti slušanja, i snižava prikazanu pouzdanost kako bi naznačio da je dokaz slabiji. Između otprilike 400 Hz i 2700 Hz položaj u propusnom pojasu nije važan; izvan toga pojavljuje se jantarni gumb **Recentre & retry**, koji jednom pomakne ugađanje i pokreće novo mjerenje. Kada prihvatite prijedlog, prijarnik se namjesti za taj način rada: ugođena frekvencija dolazi na sredinu slapa, pogled se postavlja na oko 100 kHz oko nje, a bočni pojas i propusni pojas postaju oni s kojima se taj način radi — podpojas od 3 kHz za obitelj FT8, 1350–1650 Hz za WSPR, ± 250 Hz i CW za telegrafiju, i tako dalje. Isključeno po zadatome, a u radu oko 0,5 % jedne jezgre (vidi [Dekodireri](#))
- **Predložci digitalnih modova:** pritisak na FT8, FT4, FT2, JS8 ili WSPR postavlja i bočni pojas i propusni pojas
- **JS8 prozor razgovora:** poruke iz više okvira sastavljaju se u cijele rečenice
- **Nadzorna ploča statistike:** statistika poslužitelja i korisnika u stvarnom vremenu
- **Popis povezanih korisnika:** svi koji slušaju, s gumbom za ugađanje u svakom retku. Vaša vlastita sesija označena je s **you** — i u prozoru sučelja i kad se `users.html` otvori kao zasebna stranica
- **Automatska prijava spotova:** autorun daemon dekodira FT8, FT4 i WSPR na poslužitelju i šalje spotove na PSK Reporter i WSPRnet, s dva brojača na administratorskoj ploči — pločice po dekodireru za tekući rad i ukupan zbroj od početka uz svaki potvrdni okvir pojasa/načina rada

- **Grafikoni poslužitelja:** stranica **Grafikoni** na administratorskoj ploči s frekvencijom procesora, opterećenjem, temperaturom i brojem korisnika kroz zadnjih 15 minuta do 24 sata (vidi [Administratorska ploča](#))
- **Toplinska zaštita:** čuvar u administratorskoj ploči zaustavlja poslužitelj ako se procesor pregrije i pokreće ga ponovno kad se ohladi, s pragovima izvedenima iz kritičnog praga vašeg vlastitog procesora i bez ikakvih pretpostavki o tome kako pokrećete ili zaustavljate poslužitelj; isporučuje se u načinu samo zapisivanja, pa ne čini ništa dok ga ne uključite (vidi [Thermal Guard](#)) Za prijamnik bez nadzora **snažno se preporučuje** ploču pokretati kao systemd jedinicu — čuvar živi u njoj, pa inače ponovno dizanje sustava odnese i zaštitu (vidi [Administratorska ploča](#)).
- **Emulacija KiwiSDR klijenata:** neobavezan most koji na istom računalu i istim vratima odgovara i na KiwiSDR protokol, pa se Kiwi softver poput **AetherSDR-a** i kiwiclienta spaja izravno na prijamnik — uz stvarno ugađanje, vodopad i S-metar na istoj ljestvici kao onaj na web stranici. Isključen dok u konfiguraciju ne dodate `[kiwi_emulation] enabled = true` (vidi [Instalacija](#))
- **Upravljanje primopredajnikom (CAT):** drži vaš uređaj i prozor prijmnika na istoj frekvenciji, načinu rada i širini filtra, u jednom ili oba smjera — putem izbornika Rig u aplikaciji [Desktop PhantomSDR+](#) (Icom, Yaesu, Kenwood, Elecraft, FlexRadio i QRP Labs izravno, svaki drugi uređaj preko Hamliba, ili flrig) ili alatom CATsync Tool for WebSDRs u pregledniku. Stranice nude funkcije `catsync_*` za frekvenciju, način rada, širinu filtra i utišavanje. Radi s prijemnicima PhantomSDR-Plus, KiwiSDR, PA3FWM WebSDR i UberSDR (vidi [Upravljanje primopredajnikom](#))
- **Sustav oznaka:** uvoz/izvoz frekvencijskih oznaka
- **Tipkovni prečaci:** učinkovita navigacija i upravljanje
- **Podrška za kotačić miša:** intuitivno ugađanje frekvencije
- **WebSDR imenik:** integracija s <https://sdr-list.xyz>

Podržani hardver

PhantomSDR-Plus podržava širok raspon SDR prijmnika:

Uređaj	Frekvencija uzorkovanja	Format	Sučelje
RX888 MK2	Do 64 MHz	16-bitni	Izvorno
RTL-SDR	Do 3.2 MHz	8-bitni	rtl_sdr
HackRF One	Do 20 MHz	8-bitni	hackrf_transfer
Airspy HF+ / Discovery	Do 912 kHz	16-bitni	SoapySDR (SoapyAirspyHF) + rx_sdr
SDRplay RSP1A	Do 10 MHz	16-bitni	SoapySDR (libmirisdr-5 + SoapyMiri, ili SDRplay API) + rx_sdr
RigExpert Fobos SDR	50 MHz (HF izravno uzorkovanje) / do 20 MHz (RF)	16-bitni / 32-bitni float	SoapySDR (SoapyFobosSDR) + rx_sdr
Ostali uređaji	Promjenjivo	Razno	SoapySDR/rx_tools

Brzi početak

Sistemske zahtjevi

Minimalni:

- Ubuntu 22.04 LTS (preporučeno) ili Fedora
- Dvojezgreni procesor
- 4 GB RAM-a
- 10 GB prostora na disku

Preporučeni:

- Ubuntu 24.04 LTS
- Procesor s 4+ jezgre (Ryzen 5 2600 ili Intel i5-6500T ili bolji)
- 8 GB RAM-a
- GPU s podrškom za OpenCL (nije obavezno, ali je vrlo preporučljivo)
- SSD pohrana

Instalacija

```
# Kloniranje repozitorija
git clone --recursive https://github.com/sv1btl/PhantomSDR-Plus
cd PhantomSDR-Plus

# Postavljanje skripti kao izvršnih
chmod +x *.sh

# Pokretanje automatske instalacije
./install.sh
```

Napomena: nakon pokretanja `install.sh` ponovno pokrenite terminal prije nastavka.

Već vam radi PhantomSDR-Plus? Nemojte ga instalirati ponovno — ažurirajte ga. Dohvatite alat za ažuriranje jednom i pokrenite ga; vaša konfiguracija, oznake, administratorska lozinka, popis frekvencija i povijest razgovora nikada se ne diraju, a ono što ste sami mijenjali predočava vam se umjesto da bude prepisano:

```
bash cd ~/PhantomSDR-Plus curl -fLO https://raw.githubusercontent.com/sv1btl/PhantomSDR-Plus/main/update.sh
chmod +x update.sh ./update.sh
```

Posljednji redak samo *prijavljuje* što bi se promijenilo i ništa ne zapisuje; `./update.sh --apply` to i obavi. Pojedini su u poglavlju *Ažuriranje PhantomSDR-Plusa*. Ponovno pokretanje instalacijske skripte na postaji koja radi potrebno je jedino ako izgradnja padne zbog nedostajućih sistemskih paketa.

Na Fedori koristite `install_fedora.sh`, na Archu `install_arch.sh`, a na openSUSE Tumbleweedu `install_opensuse.sh` — rade isti posao s drugim upraviteljem paketa. Svako izdanje Debiana i Ubuntu pokriva sam `install.sh`.

Ništa se ne mora instalirati ručno unaprijed. Skripta instalira ovisnosti za građenje i Node.js, gradi backend, gradi upravljački program za prijamnik koji odaberete (uz RX888 i udev pravila), otvara `site_information.json` da ga ispunite, gradi frontend, instalira OpenCL kada ga hardver podržava, zatim postavlja **administratorsku ploču**, **FreeDV RADE dekoder** i **poslužitelj statistike** — sva tri prema zadanim, s `n` preskačete pojedini — i završava potpunim `recompile.sh`. Interaktivna je od početka do kraja i traje od dvadesetak minuta do više od sat vremena.

Opis korak po korak nalazi se u [INSTALLATION.md](#) → Što instalacijska skripta radi. Za detaljne upute o instalaciji pogledajte [INSTALLATION.md](#).

Mjerenja performansi

Hardver	Frekvencija uzorkovanja	Opterećenje CPU-a	Kapacitet korisnika
Ryzen 5 2600 (sve jezgre)	64 MHz (32 MHz IQ)	38-40 %	100+ korisnika
AMD RX 580 (GPU)	64 MHz (32 MHz IQ)	28-35 %	100+ korisnika
Intel i5-6500T (s OpenCL)	60 MHz (30 MHz IQ)	10-12 %	100+ korisnika

Napomena: dodatno opterećenje CPU-a po korisniku minimalno je (<1 % po korisniku) kada je omogućeno hardversko ubrzanje.

Upotreba

Osnovni rad

1. **Konfigurirajte svoj SDR:** uredite odgovarajuću konfiguracijsku datoteku (npr. `config-rtl.toml`)
2. **Ažurirajte podatke o lokaciji:** uredite `frontend/site_information.json`
3. **Pokrenite poslužitelj:** pokrenite odgovarajuću početnu skriptu: `bash ./start-rtl.sh` # Za RTL-SDR `./start-rspla.sh` # Za SDRplay RSP1A `./start-airspyhf.sh` # Za Airspy HF+ `./start-fobos-hf.sh` # Za RigExpert Fobos SDR, HF1/HF2 (0-25 MHz) `./start-fobos.sh` # Za RigExpert Fobos SDR, RF (25-6000 MHz) `./start-hackrf.sh` # Za HackRF One `./start-rx888mk2.sh` # Za RX888 MK2 RSP1A, Airspy HF+ i Fobos rade preko SoapySDR-a i `rx_sdr`; instalacijska skripta postavlja njihov upravljački program kad ih odaberete (opcije 3, 6 i 5), a na već instaliranoj stanici pokrenite `./setup-rspla.sh`, `./setup-airspyhf.sh` ili `./setup-fobos.sh`. Vidi [INSTALLATION.md](#). HackRF One (opcija 7) treba samo paket `hackrf` iz distribucije: `./setup-hackrf.sh`. Svaka početna skripta je **samostalna**: zaustavlja svaku pokrenutu instancu, pokreće prijamnik + `spectrumserver`, **odvaja se u pozadinu** (preživljava zatvaranje terminala) i pokreće **watchdog** koji automatski ponovno pokreće lanac ako se on ugasi. Napredak se bilježi u `logwebsdr.txt` — pratite ga naredbom `tail -f logwebsdr.txt`. Ponovno pokretanje početne skripte znači čisto ponovno pokretanje, a `flock` zaključavanje osigurava da istodobno radi samo jedan prijamnik. Neobavezno: `SPECTRUM_CORES=0-3` za vezanje uz određene CPU jezgre, `RX_ARGS="..."` za zamjenu argumenata prijmnika bez uređivanja datoteke (`RX888_ARGS` za RX888), `RX_DRIVER=miri|sdrplay` za odabir upravljačkog programa za RSP1A.
4. **Pristupite sučelju:** otvorite preglednik na `http://localhost:PORT` (zadani port ovisi o konfiguraciji)

Zaustavljanje poslužitelja

Jedna zajednička skripta za zaustavljanje radi za **svaki** prijamnik — prvo zaustavlja `watchdog` (kako se ne bi automatski ponovno pokrenuo), zatim prijamnik i `spectrumserver`:

```
./stop-websdr.sh
```

Konfiguracija

Osnovne konfiguracijske datoteke

1. `config-[uređaj].toml` – konfiguracija poslužitelja i SDR-a
 2. Postavke poslužitelja (port, dretve, HTML korijen)
 3. Ulazne postavke (frekvencija uzorkovanja, veličina FFT-a, frekvencija)
 4. Mogućnosti registracije u WebSDR imenik
 5. Postavke audiokompresije
 6. Nakon izmjene ponovno pokrenite poslužitelj.
 7. `frontend/site_information.json` – javni podaci o lokaciji
 8. Podaci o operateru (pozivni znak, e-pošta, lokacija)
 9. Podaci o hardveru i anteni
 10. Postavke regije i širine pojasa
 11. Uključivanje/isključivanje razgovora (chat)
 12. Nakon izmjene pokrenite `recompile.sh` u terminalu.
 13. `markers.json` – frekvencijske oznake i plan pojaseva
 14. `frontend/src/bands-config.js` – konfiguracija plana pojaseva Ta se datoteka nalazi u `frontend/src/bands-config.js` i definira pojaseve koje stvara SysOp.
 15. Vidjet ćete nešto poput sljedećeg i slobodno to uredite po želji: ``
 16. `const bands = .....`
 17. `{ ITU: 1, name: '40m', min: -30, max: 110, initFreq: '7120000', publishBand: '1', startFreq: 7000000, endFreq: 7200000, stepi: 1000, color: 'rgba(50, 168, 72, 0.6)', modes: [{ mode: MODES.CW, startFreq: 7000000, endFreq: 7040000 }, { mode: MODES.LSB, startFreq: 7040000, endFreq: 7200000 }] }, { ITU: 2, name: '40m', min: -30, max: 110, initFreq: '', publishBand: '1', startFreq: 7000000, endFreq: 7300000, stepi: 1000, color: 'rgba(50, 168, 72, 0.6)', modes: [{ mode: MODES.CW, startFreq: 7000000, endFreq: 7050000 }, { mode: MODES.LSB, startFreq: 7050000, endFreq: 7300000 }] }, { ITU: 3, name: '40m', min: -30, max: 110, initFreq: '', publishBand: '1', startFreq: 7000000, endFreq: 7200000, stepi: 1000, color: 'rgba(50, 168, 72, 0.6)', modes: [{ mode: MODES.CW, startFreq: 7000000, endFreq: 7040000 }, { mode: MODES.LSB, startFreq: 7040000, endFreq: 7200000 }] } etc `` Pri čemu:`
 18. **ITU** je regija u kojoj se poslužitelj nalazi,
 19. **min i max** su granice svjetline slapa za zadani pojas,
 20. **initFreq** je početna frekvencija na koju se ugađa kada se odabere pojas,
 21. **publishBand** određuje prikazuje li se pojas: 1 za amaterske pojaseve, 2 za radiodifuzijske,
 22. **startFreq i endFreq** određuju granice pojasa,
 23. **stepi** je zadani korak kotačića miša za pojas, a **modes** je preferirani zadani način rada za pojas.
 24. Nakon izmjena **pokrenite recompile.sh u terminalu** iz mape PhantomSDR-Plus.
- Za primjere konfiguracije pogledajte priložene ogledne datoteke:
- `config-rtl.toml` – konfiguracija za RTL-SDR
 - `config-rsp1a.toml` – konfiguracija za SDRplay RSP1A
 - `config-airspyhf.toml` – konfiguracija za Airspy HF+
 - `config-fobos-hf.toml` / `config-fobos.toml` – konfiguracija za RigExpert Fobos SDR (HF / RF)
 - `config-hackrf.toml` – HackRF One

- `config-rx888mk2.toml` – konfiguracija za RX888 MK2

Donji prag slapa — `waterfall.sh`

`waterfall.sh` (u korijenu repozitorija) mijenja **zadanu minimalnu razinu slapa** u dB — vrijednost od koje slap kreće pri novom posjetu.

Ta vrijednost nalazi se u dvije izvorne datoteke na pet različitih mjesta (`frontend/src/waterfall.js` → `this.minWaterfall`, te `frontend/src/App.svelte` → početni `let min_waterfall`, dva slučaja „min“ resetiranja i zadana vrijednost oznaka). Ručno uređivanje lako se pogriješi, pa to skripta radi umjesto vas. Nikada ne koristi tvrdo upisane brojeve redaka — svako mjesto pronalazi po uzorku, pa nastavlja raditi i nakon ažuriranja izvornog koda.

- **Viša** vrijednost (npr. `-15`) daje **tamniji** slap.
- **Niža** vrijednost (npr. `-45`) daje **svjetliji** slap.

```
./waterfall.sh          # interaktivno – prikazuje trenutne vrijednosti i traži novu
./waterfall.sh -v -15    # postavi vrijednost bez pitanja
./waterfall.sh -v -15 -y # ... i preskoči sve potvrde
./waterfall.sh -s        # samo prikaži trenutne vrijednosti, bez izmjena
```

Prije izmjene izrađuje sigurnosnu kopiju obiju datoteka s vremenskom oznakom (`*.bak-GGGGmmdd-HHMMSS`) i ispisuje naredbu za vraćanje. Dira se samo literale jednake vrijednosti koja je trenutno u upotrebi, pa se nepovezani brojevi u tim datotekama nikada ne mogu slučajno prepisati. Ako se nova vrijednost nakon toga ne pojavi, skripta sama vraća sigurnosne kopije i izlazi s greškom. Također upozorava kada dvije datoteke nisu usklađene i postavlja obje.

- Nakon izmjene frontend se mora ponovno izgraditi — skripta nudi da za vas pokrene `recompile.sh` (uz `-y` preskače izgradnju, pa sami pokrenite `./recompile.sh` kada budete spremni).

Prilagodba

Promjena pozadinske slike

Zamijenite `frontend/src/assets/background.jpg` slikom po želji (zadržite isti naziv datoteke).

- Nakon izmjene pokrenite `recompile.sh` u terminalu.

Odabir audiokodeka

Odaberite između FLAC-a i Opusa u svojoj `.toml` konfiguraciji:

```
[input]
audio_compression="opus" # or "flac"
```

- Nakon izmjene ponovno pokrenite poslužitelj.

Kalibracija S-metra

Dva su instrumenta namjerno na **odvojenim lancima**, pa se ugađaju s dvije različite postavke.

Dvije postavke

Obje se nalaze u odjeljku `[input]` konfiguracije s kojom se pokreće poslužitelj (`config.toml`, `config-rx888mk2.toml` itd.):

```
[input]
smeter_offset=5      # samo digitalna traka
analog_smeter_offset=5 # samo analogna kazaljka
```

Čitaju se u `src/websocket.cpp` i šalju pregledniku u prvom `basic_info` okviru, pa je **dovoljno ponovno pokrenuti poslužitelj** — nije potrebna ponovna izgradnja frontenda.

Analogni lanac

```
prikazani dBm = -130 + (rawDb + analog_smeter_offset + 130) x 1,1
```

- Vizualno pojačanje 1,1 djeluje *nakon* pomaka, s uporištem na -130 dBm. Jedna jedinica `analog_smeter_offset` pomiče očitavanje za 1,1 dB. Za pomak prikaza za X dB vrijedi `analog_smeter_offset ≈ 0,91 x X`. TOML vrijednost čita se kao cijeli broj, pa su mogući samo cijeli koraci.
- Zakon kazaljke (`powerFromDbm` u `frontend/src/lib/SMeterAnalog.svelte`) stavlja **S9 na -73 dBm**, na 60 od 100 jedinica kazaljke, uz krivulju `pow(...,0,6)` od -130 do -73 ispod S9 i krivulju `pow(...,0,8)` od -73 do -13 (S9+60) iznad.
- To **nije** linearna ljestvica od 6 dB po S-jedinici, pa je smisljena točka umjeravanja S9 — oznake ispod S9 neće pratiti generator u koracima od 6 dB.

Digitalni lanac

```
value      = (rawDb / 150) x 100 + smeter_offset  -> ograničeno na [-100, 0]
segments = round((value + 100) x 35 / 100) + DIGITAL_BAR_TRIM
```

- Primijetite `/150 x 100`: digitalna je ljestvica **sažeta na 2/3** i nikada ne vidi ni `analog_smeter_offset` ni vizualno pojačanje 1,1. Zato se traka i kazaljka nikada ne poklapaju točno — to je namjerno.
- 35 segmenata na 100 jedinica znači da je jedan segment 2,86 jedinica pomaka, otprilike 4,3 dB sirovo. Dakle `smeter_offset=3` približno je jedan segment.
- `DIGITAL_BAR_TRIM` u `frontend/src/lib/SMeterDigital.svelte` (trenutno 0) pomiče traku za cijele segmente. Njegova je promjena zahvat u izvorni kod i **zahtijeva** ponovnu izgradnju frontenda.

LED prozori su u oba instrumenta vođeni analognim lancem

Prozori dBm / dBμV / SNR / NF u **oba** instrumenta čitaju analogno umjerenu vrijednost i dodaju fiksni `VISUAL_DBM_OFFSET` od 5, čvrsto upisan u `SMeterAnalog.svelte` i `SMeterDigital.svelte`. Stoga:

- `smeter_offset` **ne** mijenja nijedan broj u prozorima, samo duljinu trake.
- `dBμV = dBm + 107` (50 oma), a `NF = dBm - SNR` po konstrukciji, pa NF slijedi automatski.
- Želite li ispravne **brojeve** bez diranja položaja kazaljke, postavka je `VISUAL_DBM_OFFSET` — ali to je konstanta u izvornom kodu, traži ponovnu izgradnju i mora se promijeniti u **obje** datoteke da ostane dosljedno.

Postupak

- Dovedite poznatu razinu na antenski ulaz (generator na -73 dBm = S9), u SSB/CW sa stabilnom propusnom širinom. Očitavanja ovise o propusnoj širini i AGC-u, pa prvo fiksirajte način rada i širinu.
- Zabilježite što pokazuje analogni dBm prozor, pa postavite `analog_smeter_offset ≈ 0,91 x (-73 - prikazano)`. Ponovno pokrenite poslužitelj i provjerite. Jedna iteracija je dovoljna; kazaljka bi trebala sjesti na S9.
- Prijeđite na digitalni instrument i namjestite `smeter_offset` tako da duljina trake postavi S9 gdje želite — otprilike 3 jedinice po segmentu. Taj je korak samo estetski i ne utječe na očitavanja.

4. Bez generatora, dobra je zamjena mirni opseg s radiofarom poznate razine, ili jednostavno usidrite šumni prag na uvjerljivu vrijednost (npr. -120 dBm na 20 m s pristojnom antenom) — uz svijest da se time umjerava cijeli lanac uključujući dobitak antene, a ne samo prijamnik.

Varijante grafičkog sučelja

Sve četiri varijante sučelja (analogni/digitalni S-metar x raspored v1/v2) nalaze se u jednoj izgradnji. Posjetitelji ih prebacuju izbornikom ☞ u gornjem desnom kutu — ništa se ne učitava ponovno, pa zvuk, slap i svaki dekođer koji radi nastavlja — a odabir se pamti u svakom pregledniku.

- Pokrenite `recompile.sh` u terminalu da postavite *početnu* varijantu, onu koju posjetitelj vidi prvi put.
- **EDITING_VARIANTS.md** – kako uređivati varijante frontenda (S-metar i raspored) i ponovno ih izgraditi

Dokumentacija

- **INSTALLATION.md** – potpuni vodič za instalaciju za sistemske operatere
- **ADMIN_PANEL_SETUP.md** – vodič za instalaciju administratorske ploče
- **USER_GUIDE.md** – vodič za krajnje korisnike o radu s WebSDR-om
- **THERMAL_GUARD.md** - Priručnik za sysopa o zaštiti od pregrijavanja procesora: četiri načina rada i što učiniti u svakome
- **CONNECTION_LIMITS.md** - Priručnik za sysopa o ograničenjima veza: zaštita javnog prijemnika od poplave veza
- **MULTI_RECEIVER.md** - Priručnik za sysopa o više prijemnika na jednom računalu: instance, `receivers.toml`, izbornik prijemnika i `start-all.sh`
- **PROJECT_STRUCTURE.md** – struktura direktorija i organizacija koda
- **EDITING_VARIANTS.md** – kako uređivati varijante frontenda (S-metar i raspored) i ponovno ih izgraditi

Mrežni popis WebSDR-ova

Registrirajte svoj WebSDR u službenom imeniku: <https://sdr-list.xyz>

Postavite `register_online=true` u svojoj `.toml` konfiguracijskoj datoteci za automatsku registraciju.

Rješavanje problema

Česti problemi

OpenCL ne radi

- Provjerite jesu li upravljački programi ispravno instalirani
- Provjerite naredbom `clinfo`
- Za detaljno postavljanje OpenCL-a vidi **INSTALLATION.md**

Problemi s latencijom zvuka

- Zadržite FLAC kao kodek: on ima manju latenciju, dok Opus zadržava do 20 ms po paketu

- Otvorite stranicu s dodanim `?diag=1` na adresi (na primjer `/mobile?diag=1`) da biste vidjeli kašnjenje podijeljeno na paket, spremnik i izlaz; spremnik se sam smanji za minutu-dvije nakon problema s mrežom
- Ako se zvuk prekida, povećajte spremnik za jedan korak (vidi Korisnički priručnik, *Zvučni spremnik i kašnjenje*)
- Osigurajte dostatne CPU/GPU resurse

Neuspjeli build

- Provjerite jesu li instalirane sve ovisnosti
- Pokušajte očistiti direktorij za izgradnju: `rm -rf build && meson setup build`
- Provjerite ima li sukobljenih verzija biblioteka

Doprinosi

Doprinosi su dobrodošli! Ovo je nezavisni fork s dodatnim mogućnostima. Molimo:

1. Napravite fork repozitorija
2. Stvorite granu za novu značajku
3. Predajte (commit) svoje izmjene
4. Pošaljite granu na udaljeni repozitorij
5. Otvorite Pull Request

Licenca

Ovaj projekt licenciran je pod GNU General Public License v3.0 – pojedinih potražite u datoteci [LICENSE](#).

Autori i zasluge

- **SV1BTL i SV2AMK** – razvoj i poboljšanja projekta PhantomSDR-Plus
- Temeljeno na izvornom projektu PhantomSDR

Poveznice

- **Demonstracija uživo:** <http://phantomsdr.no-ip.org:8900/>
- **WebSDR imenik:** <https://sdr-list.xyz>
- **GitHub repozitorij:** <https://github.com/sv1btl/PhantomSDR-Plus>

Podrška

- **Problemi:** prijavite pogreške putem [GitHub Issues](#)
- **E-pošta:** kontakt sv1btl@otenet.gr

Savjeti za performanse

1. **Omogućite OpenCL/CUDA** za GPU ubrzanje – dramatično smanjuje opterećenje CPU-a
 2. **Koristite SSD pohranu** za bolje U/I performanse
 3. **Dodijelite dovoljno RAM-a** – preporučuje se najmanje 8 GB
 4. **Optimizirajte veličinu FFT-a** – veći FFT = bolja razlučivost, ali veće opterećenje CPU-a
 5. **Razmislite o namjenskom GPU-u** – AMD ili NVIDIA s podrškom za OpenCL
-

73 de SV1BTL & SV2AMK

Za detaljne upute o postavljanju vidi `INSTALLATION.md` Za vodič za rad namijenjen krajnjem korisniku vidi `USER_GUIDE.md`

POGLAVLJE 2

Vodič za instalaciju

INSTALLATION.md

Vodič za instalaciju PhantomSDR-Plusa za sistemske operatere

Ovaj sveobuhvatni vodič vodi vas kroz instalaciju i konfiguraciju PhantomSDR-Plusa na vašem poslužitelju.

Sadržaj

1. Sistemske zahtjevi
2. Priprema prije instalacije
3. Instalacija PhantomSDR-Plusa — skripta i što radi
4. Autorun Spot Reporter (FT8/FT4/WSPR)
5. Emulacija KiwiSDR klijenata (neobavezno)
6. Konfiguracija
7. Postavljanje ovisno o SDR uređaju
8. Testiranje i provjera
9. Postavljanje automatskog pokretanja
10. Toplinska zaštita procesora
11. Rješavanje problema

Samo kao referenca — instalacijska skripta sve ovo već obavlja umjesto vas. Pročitajte ove odjeljke ako ste na distribuciji koju nijedna skripta ne pokriva ili ako trebate ručno popraviti pojedini korak:

- Instalacija ovisnosti
- Instalacija Node.js-a i npm-a
- Instalacija OpenCL-a
- Ručno građenje
- Instalacija audiokodeka Opus

Sistemske zahtjevi

Podržani operacijski sustavi

Primarni (preporučeno):

- Ubuntu 22.04 LTS (Jammy), 24.04 LTS (Noble) i 26.04 LTS (Resolute) — preporučuje se 24.04
- Debian 12 (Bookworm) i Debian 13 (Trixie)

Alternativa:

- Fedora (najnovije stabilno izdanje)

- Arch Linux (kotrljajuće izdanje)
- openSUSE Tumbleweed (ne Leap — vidjeti napomenu niže)

Hardverski zahtjevi

Minimalna konfiguracija:

- Procesor: dvojezgreni (2+ GHz)
- RAM: 4 GB
- Pohrana: 10 GB slobodnog prostora
- Mreža: veza od 100 Mbit/s

Preporučena konfiguracija:

- Procesor: četverojezgreni ili bolji (Ryzen 5 2600, Intel i5-6500T ili bolji)
- RAM: 8 GB ili više
- Pohrana: SSD od 20 GB ili više
- GPU: AMD/NVIDIA s podrškom za OpenCL (vrlo preporučljivo)
- Mreža: veza od 1 Gbit/s

Konfiguracija visokih performansi:

- Procesor: 6+ jezgri (Ryzen 7, Intel i7 ili bolji)
- RAM: 16 GB ili više
- Pohrana: NVMe SSD
- GPU: namjenski GPU s podrškom za OpenCL/CUDA
- Mreža: 1 Gbit/s ili bolja

Priprema prije instalacije

1. Ažurirajte sustav

```
# Ubuntu/Debian
sudo apt update
sudo apt upgrade -y
sudo reboot

# Fedora
sudo dnf update -y
sudo reboot
```

2. Provjerite verziju Ubuntu

```
lsb_release -a
```

Očekivani ispis trebao bi pokazati: Ubuntu 24.04 LTS

3. Provjerite slobodan prostor na disku

```
df -h
```

Pobrinite se da u svom osobnom direktoriju imate najmanje 10 GB slobodno.

4. Provjerite podatke o procesoru

```
lscpu
```

Zabilježite broj jezgri/dretvi radi optimizacije konfiguracije.

Instalacija ovisnosti

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `./install.sh` — ili instalacijska skripta vaše distribucije — sve to obavlja umjesto vas; vidi [Što instalacijska skripta radi](#). Ovo je referenca za ručno postavljanje, za distribuciju koju nijedna skripta ne pokriva ili za ručni popravak pojedinog dijela.

Ubuntu 24.04 LTS

```
sudo apt install -y \
    build-essential \
    cmake \
    pkg-config \
    meson \
    libfftw3-dev \
    libwebsocketpp-dev \
    libflac++-dev \
    zlib1g-dev \
    libzstd-dev \
    libboost-all-dev \
    libopus-dev \
    libliquid-dev \
    git \
    psmisc \
    wget \
    curl
```

Fedora

```
sudo dnf install -y \
    g++ \
    meson \
    cmake \
    fftw3-devel \
    websocketpp-devel \
    flac-devel \
    zlib-devel \
    boost-devel \
    libzstd-devel \
    opus-devel \
    liquid-dsp-devel \
    git \
    psmisc \
    wget \
    curl
```

Provjerite instalaciju

```
# Check if key dependencies are installed
pkg-config --modversion fftw3
cmake --version
meson --version
```

Instalacija Node.js-a i npm-a

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `./install.sh` — ili instalacijska skripta vaše distribucije — sve to obavlja umjesto vas; vidi [Što instalacijska skripta radi](#). Ovo je referenca za ručno postavljanje, za distribuciju koju nijedna skripta ne pokriva ili za ručni popravak pojedinog dijela.

PhantomSDR-Plus zahtijeva Node.js za izgradnju sučelja. Za instalaciju ćemo koristiti NVM (Node Version Manager).

1. Instalirajte NVM

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.4/install.sh | bash
```

2. Učitajte NVM

VAŽNO: zatvorite i ponovno otvorite terminal ili pokrenite:

```
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
```

3. Provjerite instalaciju NVM-a

```
nvm --version
```

Očekivani ispis: `0.40.4` ili slično

4. Instalirajte Node.js

```
# Install Node.js 22 - the version every install script requires
nvm install 22

# Make it the default, so systemd units and cron see it too
nvm alias default 22

# Verify installation
node --version
npm --version
```

5. Neobavezno: instalirajte dodatne verzije Nodea

```
# Install latest version
nvm install node

# Install specific version (if needed)
nvm install 22.22.3

# List installed versions
nvm list

# Use specific version
nvm use 22
```

Instalacija OpenCL-a (neobavezno, ali preporučeno)

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `./install.sh` — ili instalacijska skripta vaše distribucije — sve to obavlja umjesto vas; vidi [Što instalacijska skripta radi](#). Ovo je referenca za ručno postavljanje, za distribuciju koju nijedna skripta ne pokriva ili za ručni popravak pojedinog dijela.

OpenCL dramatično poboljšava performanse prebacivanjem FFT izračuna na GPU. Ovaj odjeljak pokriva integriranu Intel grafiku. Za AMD/NVIDIA GPU pogledajte dokumentaciju proizvođača.

Intel procesor s integriranom grafikom

1. Instalirajte osnovne OpenCL komponente

```
sudo apt install -y \
  build-essential \
  cmake \
  pkg-config \
  meson \
  libfftw3-dev \
  libwebsocketpp-dev \
  libflac++-dev \
  zlib1g-dev \
  libzstd-dev \
  libboost-all-dev \
  libopus-dev \
  libliquid-dev \
  libclfft-dev \
  ocl-icd-openscl-dev \
  clinfo
```

2. Preuzmite Intel Compute Runtime

```
mkdir -p ~/neo
cd ~/neo

wget https://github.com/intel/compute-runtime/releases/download/19.07.12410/intel-
gmmlib_18.4.1_amd64.deb
wget https://github.com/intel/compute-runtime/releases/download/19.07.12410/intel-igc-
core_18.50.1270_amd64.deb
wget https://github.com/intel/compute-runtime/releases/download/19.07.12410/intel-igc-
opencl_18.50.1270_amd64.deb
wget https://github.com/intel/compute-runtime/releases/download/19.07.12410/intel-
opencl_19.07.12410_amd64.deb
wget https://github.com/intel/compute-runtime/releases/download/19.07.12410/intel-
ocloc_19.07.12410_amd64.deb
```

3. Instalirajte Intel OpenCL Runtime

```
sudo dpkg -i *.deb
```

Ako se pojave pogreške ovisnosti:

```
sudo apt --fix-broken install
```

4. Instalirajte OpenCL ICD učitavač

```
sudo apt update
sudo apt install intel-opencl-icd
```

Ako se pojave pogreške:

```
sudo apt --fix-broken install
sudo apt install intel-opencl-icd
```

5. Provjerite instalaciju OpenCL-a

```
sudo clinfo
```

Trebali biste vidjeti podatke o svojoj OpenCL platformi i uređajima. Potražite:

- Broj platformi: 1 (ili više)
- Naziv platforme: Intel(R) OpenCL (ili slično)
- Vrstu uređaja: GPU ili CPU

6. Ponovno pokrenite računalo

```
sudo reboot
```

OpenCL na AMD GPU-u

Za AMD GPU instalirajte ROCm:

```
# Add ROCm repository
wget -q -O - https://repo.radeon.com/rocm/rocm.gpg.key | sudo apt-key add -
echo 'deb [arch=amd64] https://repo.radeon.com/rocm/apt/debian/ ubuntu main' | sudo tee /etc/apt/sources.list.d/rocm.list

# Install ROCm
sudo apt update
sudo apt install rocm-opencl rocm-clinfo

# Add user to video group
sudo usermod -a -G video $USER

# Reboot
sudo reboot

# Verify
clinfo
```

OpenCL na NVIDIA GPU-u

Za NVIDIA GPU instalirajte CUDA-u:

```
# Install NVIDIA drivers
sudo apt install nvidia-driver-525

# Install CUDA toolkit
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/cuda-keyring_1.0-1_all.deb
sudo dpkg -i cuda-keyring_1.0-1_all.deb
sudo apt update
sudo apt install cuda

# Reboot
sudo reboot

# Verify
nvidia-smi
clinfo
```

Instalacija PhantomSDR-Plusa

Kloniranje repozitorija, izvršne skripte, pokretanje instalacije

```
cd ~
git clone --recursive https://github.com/svlbtl/PhantomSDR-Plus
cd PhantomSDR-Plus
chmod +x *.sh
./install.sh
```

⚠ **VAŽNO:** kada instalacija završi, **ponovno pokrenite terminal** — tek tada su novoinstalirani Node.js i Rust u vašem PATH-u.

Već vam radi PhantomSDR-Plus? Nemojte ga instalirati ponovno — ažurirajte ga. Dohvatite alat za ažuriranje jednom i pokrenite ga; vaša konfiguracija, oznake, administratorska lozinka, popis frekvencija i povijest razgovora nikada se ne diraju, a ono što ste sami mijenjali predočava vam se umjesto da bude prepisano:

```
bash cd ~/PhantomSDR-Plus curl -fLO https://raw.githubusercontent.com/sv1btl/PhantomSDR-Plus/main/update.sh chmod +x update.sh ./update.sh
```

Posljednji redak samo *prijavljuje* što bi se promijenilo i ništa ne zapisuje; `./update.sh --apply` to i obavi. Pojediniosti su u poglavlju *Ažuriranje PhantomSDR-Plusa*. Ponovno pokretanje instalacijske skripte na postaji koja radi potrebno je jedino ako izgradnja padne zbog nedostajućih sistemskih paketa.

Uzmite skriptu koja odgovara vašem sustavu. Sve rade isti posao i postavljaju ista pitanja; razlikuje se samo upravitelj paketa:

Sustav	Skripta
Ubuntu 22.04 / 24.04 / 26.04, Debian 12 / 13	<code>./install.sh</code>
Fedora	<code>./install_fedora.sh</code>
Arch	<code>./install_arch.sh</code>
openSUSE Tumbleweed	<code>./install_opensuse.sh</code>

Jedan instalacijski program za svako izdanje Debiana i Ubuntu. `install.sh` čita `/etc/os-release` i instaliranu inačicu Boosta te se sam prilagođava; stari `install_ubuntu22.sh`, `install-Deb12.sh` i `install_ubuntu26.sh` više ne postoje. Postavlja `DEBIAN_FRONTEND=noninteractive`, kako `tzdata` koji stiže uz `python3-matplotlib` ne bi mogao zaustaviti izvođenje pitanjem o vašoj vremenskoj zoni, a zatim progutati odgovor namijenjen sljedećem pitanju. Na Jammyju izravno poziva `install_rade_ubuntu22.sh`, jer je Jammyjev `python3-websockets` 10.1, a RADE traži 11.0 ili noviji. Zna gdje se `intel-ocl-icd` nalazi na svakom izdanju: u repozitorijima na 22.04 i 26.04, u `non-free` na Debianu 12 te u Intelovu vlastitom grafičkom repozitoriju na 24.04 i Debianu 13. A ondje gdje je Boost 1.87 ili noviji, zakrpa `websocketpp` zaglavlja prestaje biti neobavezna — instalacijski program provjerava da je primijenjena i bez nje odbija graditi. Noviji prevoditelj nikada nije potreban: tvornički GCC prihvaća `-std=c++23` na svih pet izdanja, pa zbog toga nemojte instalirati `gcc-12`.

Instalacija teče u 17 jasno numeriranih koraka, a svaka točka na kojoj se čeka na vas uokvirena je natpisom **POTREBAN JE VAŠ UNOS**, tako da se pitanje ne može zamijeniti s ispisom koji promiče. Sedam pitanja navedeno je odmah na početku, prije nego što se išta instalira. `PHANTOM_NONINTERACTIVE=1` odgovara na sva njihovim zadanim vrijednostima; pogledajte zaglavlje `install.sh` za varijable `PHANTOM_*`.

Svako pokretanje zapisuje `install.txt`. Kada instalacijski program završi — ili prekine na pola puta — zapisuje izvještaj u `install.txt` u mapi `PhantomSDR-Plus`: rezultat, svaki od 17 koraka kao OK / SKIPPED / PARTIAL / FAILED, što je otkrio (distribuciju, Boost, prevoditelj, Node.js), koje su komponente instalirane i svako upozorenje koje se pojavilo. Neuspjelo pokretanje ostavlja izvještaj koji završava na koraku koji je pao, s razlogom i napomenom da se instalacijski program smije sigurno ponovno pokrenuti. To je prva datoteka koju treba pročitati kada nešto nije uspjelo i prva koju treba priložiti prijavi greške. Svako pokretanje je prepisuje, pa sačuvajte kopiju ako želite usporediti dvije instalacije.

Ubuntu 26.04, Arch i openSUSE Tumbleweed se prevode, ali nisu isprobani u stvarnom radu. Sva tri donose Boost noviji od 1.87, koji je uklonio `io_service` API za koji je pisan ugrađeni `websocketpp` 0.8.2. Zakrpana zaglavlja koja instalacijski program kopira premošćuju taj jaz (`io_context`, `executor_work_guard`, `boost::asio::post`, moderni resolver), a ondje gdje je Boost 1.87 ili noviji program tu zakrpu smatra obveznom, a ne neobaveznom — provjerava da su kopije stigle i bez njih odbija graditi. Potpuna instalacija, uključujući upravljački program prijarnika i sve neobavezne komponente, provjerena je od početka do kraja u kontejnerima na Boostu 1.90 (Ubuntu 26.04), 1.91 (openSUSE Tumbleweed) i 1.92 (Arch). To dokazuje da se poslužitelj prevodi i pokreće — ne i da satima posluhuje prijarnik. Smatrajte sva tri neprovjerenima u produkciji dok netko ne javi.

openSUSE ovdje znači Tumbleweed. Ondje je `install_opensuse.sh` provjeren. Leap 15.6 ne radi: u njegovim repozitorijima uopće nema `liquid-dsp-devel` koji poslužitelj treba, a Boost postoji samo pod verzioniranim nazivima paketa. Podrška za Leap značila bi dodavanje vanjskih OBS repozitorija, pa je zasad izvan opsega.

Što instalacijska skripta radi

Ništa se ne mora pripremati ručno unaprijed — nema popisa paketa za kopiranje, nema Node.js-a za dohvaćanje, nema traženja OpenCL paketa. Teče u 19 numeriranih koraka i zaustavlja se uz najviše deset pitanja, svako uokvireno natpisom „POTREBAN JE VAŠ UNOS" — pa ili ostanite za tipkovnicom ili postavite `PHANTOM_NONINTERACTIVE=1` i pustite ga da na sve odgovori zadanim vrijednostima (vidi niže), a računajte na dvadesetak minuta do znatno više od sat vremena, ovisno o stroju i o tome koliko dodataka zadržite.

#	Korak	Što vas pita
1	Popisuje PhantomSDR-Plus servise koji su trenutačno pokrenuti — administratorska ploča, obrnuti proxy, poslužitelj statistike, prijamnik — i nudi da ih zaustavi prije nego što išta dirne. Kao prijamnik računa se samo pokrenuta skripta za pokretanje; ona koja je samo otvorena u uređivaču ostaje netaknuta.	potvrda, zadano da
2-6	Prepoznaje distribuciju i instalira sve ovisnosti za građenje (prevoditelj, meson/ninja, FFTW, Boost, FLAC, Opus, liquid-dsp, zlib/zstd, libcurl ...), uz Node.js 22 preko nvm-a ako ga sustav nema ili je prestar	ništa
7	Gradi backend pomoću mesona	ništa
8	Gradi upravljački program za vaš prijamnik — RX888 MkII / RX888, RTL-SDR (Blog V4 se pita zasebno), SDRplay RSP1A, RigExpert Fobos SDR, Airspy HF+, HackRF One ili nijedan. Uz RX888 instalira i udev pravila , pa poslužitelju nikad ne treba <code>sudo</code> za uređaj. RSP1A, Fobos i Airspy HF+ rade preko SoapySDR-a: njihov izbor pokreće <code>setup-rsp1a.sh</code> , <code>setup-fobos.sh</code> ili <code>setup-airspyhf.sh</code> , koji grade upravljački program i <code>rx_sdr</code> te također instaliraju udev pravilo (vidi Prijamnici preko SoapySDR-a (RSP1A, Fobos, Airspy HF+)) HackRF One ne treba SoapySDR: njegov izbor pokreće <code>setup-hackrf.sh</code> , koji instalira paket <code>hackrf</code> iz distribucije i udev pravilo.	koji SDR imate
9	Otvora <code>frontend/site_information.json</code> u vašem uređivaču	pozivni znak, lokator, oprema, antena — nemojte preskočiti
10-11	Instalira ovisnosti frontenda i gradi stranice za računalo i <code>/mobile</code>	ništa
12	Instalira OpenCL i bira pružatelja prema pronađenom hardveru (Intel / AMD / NVIDIA GPU ili x86 CPU runtime). Ako nema uređaja s podrškom, to kaže i nastavlja dalje	potvrda, zadano da
13-15	Instalira administratorsku ploču, FreeDV RADE V1 dekodera i poslužitelj statistike — sva tri prema zadanome	potvrda za svaki, zadano da; svaki ima svoja pitanja
16	Ponovno primjenjuje pet zakrpanih websocketpp zaglavlja preko meson podprojekta i provjerava da su stigla. Tri od njih su rad na kompatibilnosti s Boostom ≥ 1.87 , bez kojega se backend ne prevodi na Boostu 1.90; druga dva su vlastite izmjene projekta, od kojih je jedna ispravak koji treba registracija na websdr.org	ništa
17	Instalira emulaciju KiwiSDR klijenata pokretanjem <code>kiwi_install.sh</code> , kako bi se Kiwi klijenti poput AetherSDR-a mogli spojiti na ovaj prijamnik. Zakrpava izvore i dodaje <code>[kiwi_emulation]</code> u konfiguracijske datoteke u korijenu repozitorija — vidi Emulacija KiwiSDR klijenata	potvrda, zadano da
18	Pokreće <code>recompile.sh</code> , da se sve izgradi iz zakrpanih izvora	[3] Both backend and frontend → početna varijanta → [1] <code>build-all.sh</code>
19	Ispisuje sažetak: svaki korak s ishodom i svaku komponentu s time što je instalirano	ništa

Instalacija bez nadzora

Svako pitanje ima varijablu okoline koja ga nadjačava, a instalacijski program i sam prelazi na zadane vrijednosti kada `stdin` nije terminal (cijev, kontejner, CI posao). Jednako vrijedi za sva četiri: `install.sh`, `install_arch.sh`, `install_fedora.sh` i `install_opensuse.sh`. Postavite `PHANTOM_NONINTERACTIVE=1` i cijelo se izvođenje dovrši bez ijednog pitanja:

Varijabla	Učinak
PHANTOM_NONINTERACTIVE=1	answer every question with its default
PHANTOM_SDR=1...7	RX888 · RTL-SDR · SDRplay RSP1A · skip · Fobos · Aircspy HF+ · HackRF (default 4)
PHANTOM_RTLSDR_V4=y\ n	RTL-SDR Blog V4 driver (default n)
PHANTOM_SITE_EDIT=y\ n	open site_information.json in an editor
PHANTOM_OPENCL=y\ n	install OpenCL (default y)
PHANTOM_OPENCL_PROVIDER=1\ 2\ 3	Intel · Mesa/Rusticl · POCL (default: from the detected hardware)
PHANTOM_ADMIN=y\ n	admin panel (y interactive, n unattended)
PHANTOM_RADE=y\ n	RADE / FreeDV (y interactive, n unattended)
PHANTOM_STATS=y\ n	statistics server (y interactive, n unattended)
PHANTOM_KIWI=y\ n	KiwiSDR client emulation (default y)
PHANTOM_RECOMPILE=y\ n	final rebuild (default y)
PHANTOM_CURLPP=y\ n	nastavi bez curlpp-a — samo Arch i openSUSE (zadano y)
PHANTOM_FIX_CLOCK_SKEW=y\ n	poništi vremenske oznake izvornih datoteka datirane u budućnost, kako bi meson mogao graditi (zadano y)

Tri pod-instalacijska programa označena s *n bez nadzora* i sami su interaktivni, pa ih izvođenje bez nadzora preskače umjesto da zapne na njihovim pitanjima. Navedite ih izrijekom da bi bili uključeni:

```
# a complete unattended install for an RX888 MkII
PHANTOM_NONINTERACTIVE=1 PHANTOM_SDR=1 ./install.sh

# ... and with the admin panel, RADE and the statistics server too
PHANTOM_NONINTERACTIVE=1 PHANTOM_SDR=1 \
  PHANTOM_ADMIN=y PHANTOM_RADE=y PHANTOM_STATS=y ./install.sh
```

Nakon ponovnog pokretanja udev pravila za RX888 i eventualni OpenCL upravljački program potpuno su na snazi; skripta vam kaže kada je potrebno ponovno pokretanje ili ponovna prijava.

Pri postavljanju administratorske ploče instalater nudi i dvije systemd jedinice te **snažno preporučuje** da ih prihvatite: zaštita od pregrijavanja procesora radi unutar ploče, pa bez njih ponovno dizanje sustava ili pad ostavljaju računalo nezaštićenim. Korak se automatski preskače ondje gdje systemd ne radi (kontejneri, WSL1, OpenRC) — vidi [vodič za ploču](#).

Svaki je dodatak obična skripta koju možete pokrenuti i zasebno, kad god želite — `./setup_admin.sh` ([priručnik](#)), `./install_rade.sh` ([priručnik](#)), `./install-stats-server.sh` ([priručnik](#)), `./setup-rx888-udev.sh` i skripte prijmnika `./setup-rsp1a.sh`, `./setup-fobos.sh`, `./setup-airspyhf.sh` i `./setup-hackrf.sh` — upravo njih poziva instalacijska skripta.

Ručno građenje (referenca)

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `./install.sh` — ili instalacijska skripta vaše distribucije — sve to obavlja umjesto vas; vidi [Što instalacijska skripta radi](#). Ovo je referenca za ručno postavljanje, za distribuciju koju nijedna skripta ne pokriva ili za ručni popravak pojedinog dijela.

Ako ikad budete morali graditi bez instalacijske skripte — distribucija koju skripte ne pokrivaju ili popravak nedovršenog builda:

Izgradnja backenda

```
# Clean any previous builds, Configure build with optimization, Compile
rm -rf build
meson setup build --buildtype=release --optimization=3
meson compile -C build

# Verify binary was created
ls -lh build/spectrumserver
```

Izgradnja frontenda

```
cd frontend

# Install dependencies, Fix any vulnerabilities, Build the frontend
npm install
npm audit fix
npm run build

# Return to project root
cd ..
```

Provjerite instalaciju

```
# Check if binary exists
ls -lh build/spectrumserver

# Check if frontend was built
ls -lh frontend/dist/

# List important files
ls -lh *.sh *.toml
```

Instalacija audiokodeka Opus

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `./install.sh` — ili instalacijska skripta vaše distribucije — sve to obavlja umjesto vas; vidi [Što instalacijska skripta radi](#). Ovo je referenca za ručno postavljanje, za distribuciju koju nijedna skripta ne pokriva ili za ručni popravak pojedinog dijela.

Opus daje bolju kvalitetu zvuka i manju latenciju od FLAC-a.

1. Instalirajte sistemsku biblioteku libopus

```
sudo apt-get update
sudo apt-get install -y libopus0 libopus-dev
```

2. Instalirajte Opus dekodera za sučelje

```
cd PhantomSDR-Plus/frontend

# Install npm dependencies if not already done, Install Opus WASM decoder, Fix any vulnerabilities,
# Rebuild frontend
npm install
npm install @wasm-audio-decoders/opus-ml
npm audit fix
npm run build

# Return to project root
cd ..
```

3. Provjerite instalaciju Opusa

```
# Check if Opus system library is installed
pkg-config --modversion opus

# Check if Opus npm package is installed
cd frontend
npm list @wasm-audio-decoders/opus-ml
cd ..
```

Autorun Spot Reporter (FT8/FT4/WSPR)

PhantomSDR-Plus dolazi s neobaveznim **autorun spot reporterom** (u direktoriju `autorun/`). Dekodira FT8/FT4/WSPR na poslužiteljskoj strani izravno s prijmnika i šalje spotove u mreže za prijavu:

- **FT8 / FT4 → PSK Reporter** (IPFIX/UDP)
- **WSPR → wsprnet.org** (HTTP)

Upravlja se u cijelosti s kartice **administratorska ploča → „Spot Reporting“**, a **prijavljivanje je prema zadanom ISKLJUČENO** — ništa se ne odašilje ni ne šalje dok to ondje ne uključite. Dekoder radi kao zaseban Node.js daemon i lokalno preuzima zvuk prijmnika, pa ne dodaje nikakav dodatni RF hardver.

Preduvjeti

Preduvjet	Napomene
Node.js 22+	Isto radno okruženje koje već treba izgradnja sučelja — instalira se u koraku Node.js .
npm paketi ws + cbor-x	Razrješavaju se preko <code>autorun/node_modules</code> , simboličke poveznice na <code>node_modules</code> sučelja (oba su paketa navedena u <code>frontend/package.json</code>).
util-linux (taskset)	Gumb „Start“ u administraciji veže daemon uz E-jezgre pomoću naredbe <code>taskset</code> . Prisutan je u gotovo svakoj distribuciji; instalacijske skripte dodaju ga izričito.
Pozivni znak + lokator	Čitaju se iz <code>frontend/site_information.json</code> (<code>siteSysop</code> / <code>siteGridSquare</code>), osim ako se ne promijene na administratorskoj kartici. Spotovi se šalju pod tim pozivnim znakom.

⚠ **Prijavljujte samo ono što doista primete.** Spotovi se šalju u javne mreže pod vašim pozivnim znakom — uključite samo pojaseve/načine rada koje vaš prijmnik stvarno čuje i koristite ispravan lokator.

Automatska instalacija

`install.sh` (i inačice po distribucijama: `install_arch.sh`, `install_fedora.sh`, `install_opensuse.sh`) obavljaju sve umjesto vas: instaliraju Node.js 22 i `util-linux`, pokreću `npm install` za sučelje i zatim automatski stvaraju simboličku poveznicu `autorun/node_modules`. Nisu potrebni dodatni koraci — značajka je spremna čim `install.sh` završi.

Ručna instalacija

NAPOMENA

Instalacijska skripta to već radi umjesto vas: stvara simbolički link i instalira `util-linux`. Korake u nastavku koristite samo za ručni popravak.

Ako ste instalirali ručno (opcija B), sami stvorite simboličku poveznicu nakon `npm install` za sučelje kako bi daemon mogao razriješiti svoje ovisnosti:

```
cd ~/PhantomSDR-Plus

# 1. Frontend deps must be installed first (ws + cbor-x live there)
cd frontend && npm install && cd ..

# 2. Link the autorun daemon to the frontend's node_modules
ln -sf ../frontend/node_modules autorun/node_modules

# 3. Ensure taskset is available (Debian/Ubuntu shown; use your package manager)
sudo apt install -y util-linux
```

Napomena: `autorun/node_modules` git zanemaruje, pa ga svježi `git clone` nikad ne sadrži — poveznicu treba (ponovno) stvoriti nakon svakog čistog preuzimanja. Instalacijske skripte to rade umjesto vas; gornja naredba odnosi se samo na ručna postavljanja.

Provjera

```
# The symlink should point at ../frontend/node_modules
ls -l autorun/node_modules

# ws + cbor-x must resolve
ls -d frontend/node_modules/ws frontend/node_modules/cbor-x

# taskset must be on PATH
command -v taskset
```

Zatim otvorite administratorsku ploču, idite na karticu **Spot Reporting**, postavite svoj identitet, označite pojaseve/načine rada za dekodiranje, uključite odredišta i pritisnite **Start**. Oznaka „REPORTING“ pojavljuje se na glavnom slapu dok je prijavljivanje aktivno. (PSK Reporter šalje skupno svakih 5 minuta, a wsprnet svake 2 minute, pa svježe pokrenut daemon prvih nekoliko minuta pokazuje „0 sent“ — to je normalno.)

Zatim se prikazuju dva različita brojača koja je lako zamijeniti. Pločice **SPOTS UPLOADED PER DECODER** broje samo trenutačni rad, pa ih Stop/Start vraća na nulu; broj uz svaki potvrdni okvir pojasa/načina rada je **ukupan** zbroj tog mjesta od početka, spremljen u `autorun-totals.json` pa preživljava ponovna pokretanja. Oboje opisuje [vodič za administratorsku ploču](#), kao i stranicu **Grafikoni**, koja crta frekvenciju procesora, opterećenje, temperaturu i broj korisnika u rasponu od 15 minuta do 24 sata.

Ako vam se stroj grije, ploča nosi i **toplinsku zaštitu** koja zaustavlja poslužitelj kada procesor dosegne opasnu temperaturu i pokreće ga ponovno kad se ohladi. Pragove izvodi iz kritičnog praga vašeg vlastitog procesora, pa nema ničega za računati, i radi s bilo kojim načinom pokretanja/zaustavljanja. Instalira se s pločom, ali kreće u načinu samo zapisivanja: bilježi što bi *učinila* i ne mijenja ništa dok je ne uključite u Postavkama. Neprekidno dekodiranje drži procesor zaposlenim danonoćno, pa se isplati pročitati karticu CRASH nakon tjedan dana autoruna i vidjeti koliko vam se stroj zaista približi.

Port poslužitelja otkriva se automatski. Daemon se spaja izravno na `[server]` port samog spectrumservera (petlja + token, zaobilazeći proxy). Taj port čita iz konfiguracijske datoteke s kojom je pokrenut spectrumserver koji radi, pa funkcionira na bilo kojem portu bez konfiguracije — redak `[autorun] tap backend: ...` u `autorun.log` pokazuje što je razriješio. Ako vaš poslužitelj nije radio kada je daemon pokrenut ili imate neobičnu postavu, zadajte ga u `autorun.json`: `json "server": { "host": "127.0.0.1", "port": 9002 }` Pogrešan port očituje se kao `504 / tap closed 1006` u `autorun.log`, uz `decodes` zaglavljen na 0.

Emulacija KiwiSDR klijenata (neobavezno)

Od v4.1.0 PhantomSDR-Plus može odgovarati i na **KiwiSDR protokol**, pa se softver pisan za KiwiSDR — **AetherSDR**, `kiwiclient` i ostali — spaja izravno na vaš prijamnik, na istom računalu i portu koje već objavljujete. Isključeno je dok se u konfiguraciju s kojom radi vaš prijamnik ne doda `[kiwi_emulation] enabled = true`.

Instalacijski program to nudi kao korak 17; `./kiwi_install.sh` primjenjuje most na već instalirano stablo.

Potpuna dokumentacija: Emulacija KiwiSDR klijenata — što most radi, kako ga instalirati, svaki ključ iz `[kiwi_emulation]`, spajanje klijenta, razina zvuka, S-metar, brzina slapa i tablica simptoma.

Konfiguracija

1. Odaberite konfiguracijsku datoteku

Odaberite odgovarajuću konfiguracijsku datoteku za svoj SDR:

- `config-rtl.toml` – RTL-SDR uređaji
- `config-rspla.toml` – SDRplay RSP1A
- `config-airspyhf.toml` – Airspy HF+ Discovery
- `config-fobos-hf.toml` – RigExpert Fobos SDR, HF1/HF2 izravno uzorkovanje (0-25 MHz)
- `config-fobos.toml` – RigExpert Fobos SDR, RF grana (25-6000 MHz)
- `config-rx888mk2.toml` – RX888 MK2
- `config-hackrf.toml` – HackRF One
- `config.example.hackrf.toml` – HackRF One

U ovom primjeru koristit ćemo RTL-SDR.

2. Uredite konfiguracijsku datoteku

```
nano config-rtl.toml
```

Ključne postavke

```
[server]
port = 9002                # Web interface port (or everything else)
html_root = "frontend/dist/" # Frontend location
otherusers = 1              # Show other users (1=yes, 0=no)
threads = 2                 # Number of server threads

[websdr]
register_online = true       # Register on sdr-list.xyz (true/false)
name = "Your WebSDR Name"   # Display name
antenna = "Your Antenna Type" # e.g., "Vertical", "Loop", "Dipole"
grid_locator = "AB12cd"     # Your Maidenhead grid square
hostname = "your.domain.com" # Your domain or IP address

[input]
sps = 2048000               # Sample rate (adjust for your coverage)
fft_size = 131072           # FFT size (higher = better resolution)
brightness_offset = -10     # Waterfall brightness adjustment
frequency = 145000000        # Base frequency in Hz (145 MHz for 2m)
signal = "iq"               # "iq" for complex, "real" for real sampling
fft_threads = 2             # FFT processing threads
accelerator = "opencl"      # "none", "cuda", or "opencl"
audio_sps = 12000           # Audio sample rate (keep at 12000)
audio_compression = "opus"  # "flac" or "opus"
smeter_offset = -2          # S-meter calibration
waterfall_size = 1024       # Waterfall FFT size
waterfall_compression = "zstd" # Waterfall compression

[input.driver]
name = "stdin"              # Input driver
format = "u8"               # Sample format for RTL-SDR

[input.defaults]
frequency = 145500000        # Default tuning frequency
modulation = "FM"           # Default modulation mode
```

Smjernice za frekvenciju uzorkovanja

Pokrivenost	Frekvencija uzorkovanja	Veličina FFT-a
2 MHz	2048000	131072
3.2 MHz	3200000	131072
10 MHz	10000000	1048576
30 MHz	30000000	2097152
60 MHz	60000000	4194304

3. Postavite podatke o lokaciji

```
nano frontend/site_information.json
```

Uredite sljedeća polja:

```
{
  "siteSysop": "YourCallsign",
  "siteSysopEmailAddress": "your@email.com",
  "siteGridSquare": "AB12cd",
  "siteCity": "Your City, Country",
  "siteInformation": "https://github.com/sv1btl/PhantomSDR-Plus",
  "siteHardware": "Computer specifications",
  "siteSoftware": "PhantomSDR-Plus v5.0.0",
  "siteReceiver": "Your SDR model",
  "siteAntenna": "Antenna description",
  "siteNote": "Additional information",
  "siteIP": "http://your.domain.com:9002",
  "siteSDRBaseFrequency": 0,
  "siteSDRBandwidth": 2048000,
  "siteRegion": 1,
  "siteChatEnabled": true
}
```

IARU regije:

- **1:** Europa, Afrika, Bliski istok, sjeverna Azija
- **2:** Amerike (Sjeverna, Srednja, Južna), Karibi
- **3:** Azija i Pacifik, Oceanija

4. Prilagodite frekvencijske oznake (neobavezno)

```
nano markers.json
```

Dodajte svoje omiljene frekvencije, repetitore i radiodifuzijske postaje.

5. Uredite početnu skriptu

```
nano start-rtl.sh
```

Početna skripta samostalan je pokretač s watchdogom. Uredite samo blok **RECEIVER CONFIGURATION** pri vrhu kako bi argumenti prijmnika odgovarali vašoj postavi:

```
# == RECEIVER CONFIGURATION (the only receiver-specific part) ==
RX_LABEL="RTL-SDR"
RX_COMM="rtl_sdr"                # process name to monitor/kill
CONFIG="$PHANTOMDIR/config-rtl.toml" # your .toml
FIFO="$PHANTOMDIR/rtl.fifo"
RX_ARGS="${RX_ARGS:-f 145000000 -s 2048000 -}" # receiver args (edit these)
```

Parametri unutar `RX_ARGS`:

- `-f 145000000`: središnja frekvencija (145 MHz)
- `-s 2048000`: frekvencija uzorkovanja (2,048 MSPS)

`CONFIG` pokazuje na vašu `.toml` datoteku. Ništa drugo u skripti **ne** morate mijenjati — logika pokretanja/ponovnog pokretanja/watchdoga/zapisivanja je općenita. Argumente prijmnika možete zamijeniti i pri pokretanju, bez uređivanja datoteke: `RX_ARGS="-f 145000000 -s 2048000 -" ./start-rtl.sh` (`RX888_ARGS` za `start-rx888mk2.sh`).

Postavljanje ovisno o SDR uređaju

RTL-SDR

Instalirajte RTL-SDR alate

```
sudo apt install -y rtl-sdr
```

Testirajte RTL-SDR

```
rtl_test
```

Pritisnite Ctrl+C za prekid. Trebali biste vidjeti podatke o frekvenciji uzorkovanja.

Uredite konfiguraciju

```
nano config-rtl.toml
```

Uobičajene postavke za RTL-SDR:

```
sps = 2048000
frequency = 145000000
signal = "iq"
format = "u8"
```

Prijamnici preko SoapySDR-a (RSP1A, Fobos, Airspy HF+)

SDRplay RSP1A, RigExpert Fobos SDR i Airspy HF+ do poslužitelja stižu na isti način: SoapySDR upravljački program za uređaj i `rx_sdr` (iz `rx_tools`) koji njegove uzorke prosljeđuje `spectrumserver-u`. Jedna skripta po prijamniku instalira cijeli lanac:

Prijamnik	Skripta	Opcija instalacije	Skripta za pokretanje	Konfiguracija
SDRplay RSP1A	<code>./setup-rspla.sh</code>	3	<code>start-rspla.sh</code>	<code>config-rspla.toml</code>
RigExpert Fobos SDR	<code>./setup-fobos.sh</code>	5	<code>start-fobos-hf.sh</code> (HF) / <code>start-fobos.sh</code> (RF)	<code>config-fobos-hf.toml</code> / <code>config-fobos.toml</code>
Airspy HF+	<code>./setup-airspyhf.sh</code>	6	<code>start-airspyhf.sh</code>	<code>config-airspyhf.toml</code>

Svaka skripta:

- instalira `git`, `cmake`, prevoditelj, `libusb` i **SoapySDR** iz paketa vaše distribucije — prepozna je `apt`, `dnf`, `pacman` i `zypper`, pa jednako radi na Debianu/Ubuntuu, Fedori, Archu i openSUSE-u;
- gradi upravljački program prijamnika iz izvornog koda u `sdr_drivers/` (git ga ne prati);
- gradi **rx_tools** za `rx_sdr`, osim ako je `rx_sdr` već instaliran — jedna kopija služi svim prijamnicima;
- instalira **udev pravilo** i dodaje vas u grupu `plugdev`, pa skripti za pokretanje nikad ne treba `sudo` za uređaj. **Jednom se odjavite i ponovno prijavite** da grupa počne vrijediti;
- na kraju provjerava da SoapySDR stvarno navodi upravljački program, a ako ne, zaustavlja se s greškom.

Pokrenite ih kao običan korisnik — same pozivaju `sudo`. Idempotentne su: ponovno pokretanje ažurira izvorni kod upravljačkog programa s `git pull` i ponovno gradi. Instalacijska skripta poziva upravo te

skripte; na već instaliranoj stanici pokrenite zasebno onu za svoj prijamnik. Zajednički dio nalazi se u `setup-sdr-common.sh`, koja se ne pokreće sama. Dvije neobavezne varijable vrijede za sve tri: `SDR_USER=<ime>` stavlja drugi račun u `plugdev`, a `SDR_SKIP_DEPS=1` preskače instalaciju paketa.

S priključenim uređajem provjerite da je prijamnik pronađen:

```
SoapySDRUtil --find="driver=soapyMiri"    # RSP1A
SoapySDRUtil --find="driver=fobos"       # Fobos
SoapySDRUtil --find="driver=airspyhf"    # Airspy HF+
```

SDRplay RSP1A

`setup-rsp1a.sh` instalira **upravljački program otvorenog koda**: `libmirisdr-5`, upravljački program nastao obrnutim inženjeringom za Mirics čipove MSI2500/MSi001 u RSP1 i RSP1A, s njegovim SoapySDR modulom `SoapyMiri` (`driver=soapyMiri`). Ne treba mu pozadinska usluga ni root. Skripta također stavlja na crnu listu upravljačke programe jezgre `msi2500` i `msi001` u `/etc/modprobe.d/blacklist-msi2500.conf`, jer oni zauzmu RSP1A kao V4L2 radio prije nego što ga `rx_sdr` može otvoriti. Ako je uređaj bio priključen dok su bili učitani, jednom ga iskopčajte i ponovno ukopčajte (ili ponovno pokrenite računalo).

Zatvoreni SDRplay API (`driver=sdrplay`, sa `SoapySDRPlay` i uslugom `sdrplay`) i dalje radi — priložena datoteka `instructions-for-rsp1a` opisuje tu instalaciju. `start-rsp1a.sh` sam bira između njih: ako SoapySDR ima upravljački program `sdrplay`, koristi API i prije toga ponovno pokreće uslugu `sdrplay`, a inače koristi `libmirisdr-5`. Stanica postavljena na API tako nastavlja raditi bez promjena. Da nametnete jedan, pokrenite s `RX_DRIVER=miri` ili `RX_DRIVER=sdrplay`. Odabrani upravljački program upisuje se u `logwebsdr.txt` kao RSP1A driver: `miri` ili RSP1A driver: `sdrplay`.

RigExpert Fobos SDR

Fobos šalje **samo jednu granu odjednom**, pa ima dvije skripte za pokretanje koje dijele zaključavanje `watchdoga` kao svaka `start-*.sh` — pokretanje jedne zaustavlja drugu:

- **`start-fobos-hf.sh`** — ulazi HF1/HF2 u izravnom uzorkovanju, 0-25 MHz, na portu 9003. U tom načinu rada nema lokalnog oscilatora: ADC neprestano digitalizira 0-25 MHz (`-f 0` je namjerno, a upozorenje *Failed to set center freq* iz `rx_sdr` je očekivano). `rx_sdr` daje CF32 na 50 Msps; mali pretvarač `cf32_to_real` (gradi ga `setup-fobos.sh` iz `cf32_to_real.c`) zadržava kanal I i skalira ga na s16, pa `config-fobos-hf.toml` koristi `signal="real"` i `format="s16"`.
- **`start-fobos.sh`** — RF grana, 25-6000 MHz, IQ, na portu 9002. Odaberite prozor prije prvog pokretanja: `-f` (središte) i `-s` (brzina uzorkovanja) u retku `RX_ARGS` moraju odgovarati `frequency=` i `sps=` u `config-fobos.toml`. Isporučene vrijednosti (97 MHz, 20 Msps) samo su polazište za provjeru da radi.

VAŽNO

`config-fobos-hf.toml` dolazi s `accelerator="opencl"`: tok od 50 Msps kroz FFT od 1M binova težak je kao 60 Msps RX888, a na CPU-u gubi uzorke i grije računalo. Na računalu bez OpenCL uređaja `spectrumserver` se s tom postavkom ne pokreće — ondje postavite `accelerator="none"`.

`SoapyFobosSDR` za izgradnju treba zaglavlja i od `libfobos` i od `libfobos-sdr-agile`, pa `setup-fobos.sh` gradi oboje, čak i za standardni firmware. Podrška za Fobos potječe iz skripte koju je napisala druga stanica nakon što je pokrenula Fobos na PhantomSDR-Plus; HF vrijednosti u `config-fobos-hf.toml` su one koje su ondje radile.

Airspy HF+

`setup-airspyhf.sh` instalira `libairspyhf` iz distribucije gdje je zapakirana (Debian i Ubuntu, Fedora, openSUSE), a gradi je iz izvornog koda gdje nije (Arch je ima samo u AUR-u), zatim gradi `SoapyAirspyHF` (`driver=airspyhf`). `start-airspyhf.sh` daje 912 ksps IQ u CS16, što `config-airspyhf.toml` čita kao `format="s16"`. Priložena datoteka `instructions-for-airspy` opisuje ručni put.

RX888 MK2

Instalirajte RX888 alate

```
# Install rx_tools
git clone https://github.com/rxseger/rx_tools.git
cd rx_tools
mkdir build && cd build
cmake ..
make -j4
sudo make install
sudo ldconfig

# Install RX888 firmware and support
# Follow manufacturer instructions
```

Pokretanje rx888_stream bez sudo (udev pravila)

Prema zadanom, USB uređaj Cypress FX3 na RX-888 dostupan je samo rootu, pa bi `rx888_stream` trebao `sudo`. Pokrenite jednom priloženi pomoćni skript kako biste instalirali udev pravila i dodali se u grupu `plugdev`:

```
./setup-rx888-udev.sh          # re-runs itself with sudo automatically
```

Skripta zapisuje `/etc/udev/rules.d/99-rx888.rules` za sva tri identifikatora proizvoda Cypress FX3 (04b4:00f1/00f3 bootloader i 04b4:8613 s učitanim firmverom), ponovno učitava udev i stvara stabilnu poveznicu `/dev/rx888`. **Odjavite se i ponovno prijavite** (zbog promjene grupe) i jednom **ponovno priključite uređaj**; nakon toga `rx888_stream` — i pokretač `start-rx888mk2.sh` — rade bez `sudo`. Skripta je idempotentna pa ju je sigurno ponovno pokrenuti. Za drugi korisnički račun: `RX888_USER=<ime> ./setup-rx888-udev.sh`.

Uredite konfiguraciju

```
nano config-rx888mk2.toml
```

Uobičajene postavke za RX888:

```
sps = 60000000
frequency = 0
signal = "real"
format = "s16"
fft_size = 4194304
```

HackRF One

Za HackRF ništa se ne gradi: `hackrf_transfer` iz paketa `hackrf` vaše distribucije prosljeđuje uzorke izravno `spectrumserver-u`. `./setup-hackrf.sh` (opcija 7 instalacije) instalira taj paket s `apt`, `dnf`, `pacman` ili `zypper` te dodaje udev pravilo i grupu `plugdev`, pa `start-hackrf.sh` radi bez `sudo` — nakon toga se jednom odjavite i ponovno prijavite.

```
./setup-hackrf.sh    # installs the hackrf package and the udev rule
hackrf_info          # with the HackRF plugged in: must list the board
```

`start-hackrf.sh` šalje jedan prozor odjednom. Odaberite ga prije prvog pokretanja: `-f` (središte) i `-s` (brzina uzorkovanja) u njegovu retku `RX_ARGS` moraju odgovarati `frequency=` i `sps=` u `config-hackrf.toml`. `-l` i `-g` su pojačanja LNA (0-40 dB, koraci od 8 dB) i VGA (0-62 dB, koraci od 2 dB), a `-a 1` uključuje RF pojačalo. Isporučene vrijednosti (98 MHz, 20 Msps) samo su polazište za provjeru da radi.

hackrf_transfer piše 8-bitni IQ **s predznakom**, pa konfiguracija treba `format="s8"`.
`config.example.hackrf.toml` je do rujna 2026. imao `u8`; `s u8` svaki je uzorak pomaknut za 128 i vodopad pokazuje samo šum — provjerite svaku stariju kopiju.

Više prijemnika na jednom računalu

Jedno računalo može istodobno pokretati više gore navedenih prijemnika — na primjer RX-888 za kratki val i RTL-SDR za 2 m — svaki kao zaseban poslužitelj, s izbornikom prijemnika na stranici. Svaki dodatni prijemnik pokreće se s `INSTANCE=<ime> ./start-<radio>.sh` i podešava u `instances/<ime>/receivers.toml` ih navodi za proxy i za `start-all.sh`. Sve je opisano u [Više prijemnika](#). Najbrže se prijemnik dodaje s `./add-receiver.sh`, koji instalacijske skripte nude na kraju: instalira upravljački program (`setup-rtlshdr.sh` za RTL-SDR, odgovarajući `setup-*.sh` za ostale) i za vas zapisuje mapu instance i `receivers.toml`.

Testiranje i provjera

1. Probno pokretanje

```
# For RTL-SDR
./start-rtl.sh
```

Time se poslužitelj pokreće **u pozadini** (odvaja se i odmah vraća upravljanje) i počinje zapisivati u `logwebsdr.txt`.

2. Provjerite ima li pogrešaka

Skripta bilježi napredak u `logwebsdr.txt` — pratite ga uživo:

```
tail -f logwebsdr.txt
```

Potražite:

- `Starting the initialization script of the PhantomSDR Server`
- `<receiver> running (PID ...)`
- `Server started normally`
- `Server ... FAILED ...` ili `ERROR: receiver binary '...' not found` (ispravite argumente prijamnika ili `.toml`, ili instalirajte alat prijamnika, pa ponovno pokrenite skriptu)

3. Pristupite web-sučelju

Otvorite preglednik na:

```
http://localhost:9002
```

(Zamijenite broj porta svojim konfiguriranim portom)

4. Provjerite funkcionalnost

- Slap se mora prikazivati
- Zvuk se mora reproducirati pri kliku na signale
- S-metar mora reagirati na signale
- Brojač korisnika mora pokazivati „1“

5. Testirajte s drugog uređaja

S drugog računala u svojoj mreži:

```
http://YOUR_SERVER_IP:9002
```

6. Provjerite potrošnju resursa

```
# Monitor CPU usage
htop

# Monitor GPU usage (if OpenCL/CUDA enabled)
nvidia-smi # For NVIDIA
radeontop  # For AMD

# Monitor network usage
iftop
```

7. Zaustavite poslužitelj

Poslužitelj radi u pozadini pod watchdogom, pa ga **Ctrl+C neće zaustaviti** (a watchdog bi ga ionako ponovno pokrenuo). Koristite zajedničku skriptu za zaustavljanje, koja radi za svaki prijamnik — prvo zaustavlja watchdog, zatim prijamnik i `spectrumserver`:

```
./stop-websdr.sh
```

Postavljanje automatskog pokretanja

Pomoću systemd (preporučeno)

1. Stvorite datoteku usluge

```
sudo nano /etc/systemd/system/phantomsdr.service
```

Dodajte sljedeći sadržaj (prilagodite putanje i korisnika):

```
[Unit]
Description=PhantomSDR-Plus WebSDR Server
After=network.target

[Service]
Type=simple
User=youruser
WorkingDirectory=/home/youruser/PhantomSDR-Plus
# Run the watchdog in the FOREGROUND (note the --watchdog flag) so systemd can
# track it. Do NOT use the plain "./start-rtl.sh" here – that form detaches into
# the background and exits, which systemd would treat as the service stopping.
ExecStart=/home/youruser/PhantomSDR-Plus/start-rtl.sh --watchdog
ExecStop=/home/youruser/PhantomSDR-Plus/stop-websdr.sh
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Proces `--watchdog` već sam ponovno pokreće prijatelj `spectrumserver`; `Restart=always` tek je sigurnosna mreža za rijedak slučaj da sam watchdog završi. Budući da ovdje systemd nadzire poslužitelj, umjesto ručnog pokretanja skripti možete koristiti `systemctl start/stop/restart` i `journalctl -u phantomsdr -f`.

2. Omogućite i pokrenite uslugu

```
# Reload systemd
sudo systemctl daemon-reload

# Enable service to start on boot
sudo systemctl enable phantomsdr.service

# Start service now
sudo systemctl start phantomsdr.service

# Check status
sudo systemctl status phantomsdr.service
```

3. Upravljanje uslugom

```
# Start
sudo systemctl start phantomsdr

# Stop
sudo systemctl stop phantomsdr

# Restart
sudo systemctl restart phantomsdr

# View logs
sudo journalctl -u phantomsdr -f
```

Pomoću Screena (alternativa)

Obično nepotrebno: `./start-rtl.sh` već se odvaja u pozadinu (putem `setsid`) i nastavlja raditi nakon odjave, s vlastitim watchdogom. Screen je koristan samo ako izričito želite interaktivnu sesiju za pokretanje inačice u prednjem planu `./start-rtl.sh --watchdog`.

1. Instalirajte Screen

```
sudo apt install -y screen
```

2. Pokrenite u Screen sesiji

```
screen -S phantomsdr
./start-rtl.sh
```

Pritisnite `Ctrl+A`, zatim `D` za odvajanje.

3. Vratite se u sesiju

```
screen -r phantomsdr
```

Toplinska zaštita procesora

Cjeloviti priručnik: [THERMAL_GUARD.md](#) — četiri načina rada i što sysop mora učiniti u svakome, rad bez ploče, throttle faza bez roota, testiranje i rješavanje problema.

PhantomSDR-Plus donosi čuvara koji zaustavlja poslužitelj ako procesor dosegne opasnu temperaturu i pokreće ga ponovno kad se ohladi. Pragove izvodi iz kritičnog praga koji objavljuje vaš vlastiti procesor, pa nema ničega za računati, i radi s bilo kojim načinom pokretanja/zaustavljanja.

Ako koristite administratorsku ploču, već ga imate — radi unutar ploče i podešava se na njezinoj stranici Postavke. Vidi [vodič za administratorsku ploču](#). Ostatak ovog odjeljka odnosi se na instalacije **bez** ploče.

1. Provjerite što čuvar vidi na vašem stroju

```
cd ~/PhantomSDR-Plus
python3 thermal_guard.py --once
```

```
sensor      : hwmon:coretemp
temperature: 58.0 C
crit        : 100.0 C
mode        : log
thresholds  : warn 88.0  throttle 92.0  stop 95.0  resume 75.0 (C)
cpufreq     : not writable - throttle stage disabled
```

Koriste se samo pravi senzori na procesoru (`coretemp`, `k10temp`, `zenpower`, `cpu_thermal`); `acpitz` i neoznačene toplinske zone namjerno se zanemaruju jer često prijavljuju temperaturu kućišta ili ploče desetke stupnjeva nižu od procesora. Ako ispiše `sensor : NONE`, taj se stroj ne može zaštititi — često na VPS-u ili unutar kontejnera — i čuvar će ostati neaktivan umjesto da se pretvara.

Ništa se ne mora instalirati: `thermal_guard.py` koristi samo Pythonovu standardnu biblioteku.

2. Podesite ga

Čuvar čita `admin_config.json` uz `thermal_guard.py`. Napravite ga ako ga nemate — bez ploče ga nećete imati. Najmanja korisna datoteka:

```
{
  "sdr_process_name": "spectrumserver",
  "stop_script": "stop-websdr.sh",
  "start_script": "start-rx888mk2.sh",
  "thermal_mode": "stop+restart"
}
```

- `thermal_mode` — `log` (samo zapisuje, zadano), `throttle`, `stop` ili `stop+restart`. **To je ono što morate postaviti**, jer se čuvar isporučuje neaktivan: ostavljen takav, samo zapisuje što bi učinio. Možete ga zadati i u naredbenom retku kao `--mode stop+restart`, što nadjačava datoteku — ako vam odgovaraju sve ostale zadane vrijednosti, konfiguracijska datoteka uopće vam ne treba.
- `stop_script` / `start_script` — kako zaustavlja i pokreće vaš poslužitelj. Bez skripte za zaustavljanje pribjegava `SIGTERM`-u, pa nakon 10 sekundi `SIGKILL`-u nad `sdr_process_name`. Bez skripte za pokretanje zaustavlja, ali nikad ne pokreće ponovno.
- Pragovi i vremena mogu se postaviti i ovdje, s istim nazivima ključeva koje koristi ploča — cijela je tablica u [vodiču za administratorsku ploču](#).

Ako vaš SDR poslužitelj nadzire systemd, neka `stop_script` pokazuje na malu skriptu koja izvodi `systemctl stop vasa-jedinica`, umjesto da čuvar signalizira proces izravno. Čuvar će pobijediti u svakom slučaju — dok je procesor prevruć, ponavlja zaustavljanje svake 2 sekunde, pa je sve što oživi poslužitelj poništeno — ali čisto zaustavljanje bolje je od borbe svake dvije sekunde.

3. Pokrenite ga kao uslugu

Repozitorij sadrži gotovu jedinicu, `thermal-guard.service`. U njoj uredite `User=` i dvije putanje, a zatim:

```
sudo cp thermal-guard.service /etc/systemd/system/
sudo systemctl daemon-reload
sudo systemctl enable --now thermal-guard
systemctl status thermal-guard
```

`User=` mora biti račun kojem je dopušteno pokretati vaše skripte za pokretanje/zaustavljanje — obično isti korisnik koji pokreće SDR poslužitelj. **Čuvar koji radi kao korisnik koji ne može zaustaviti poslužitelj ostavlja vas nezaštićenima, a izgleda kao da vas štiti.**

Da čuvara naoružate iz jedinice umjesto iz konfiguracijske datoteke, dodajte `--mode` u `ExecStart`:

```
ExecStart=/usr/bin/python3 -u /home/sysop/PhantomSDR-Plus/thermal_guard.py --mode stop+restart
```

Bez `systemd-a`:

```
cd ~/PhantomSDR-Plus
nohup python3 -u thermal_guard.py --mode stop+restart >> thermal.log 2>&1 &
```

`python3 thermal_guard.py --help` navodi sve opcije, uključujući `--config` za konfiguracijsku datoteku koja nije uz skriptu.

4. Promatrajte pa vjerujte

Sve što čuvar učini zapisuje se u `crash.log` u direktoriju `PhantomSDR-Plus`, jedan događaj po retku:

```
[THERMAL] warn: 89.0C (warn=88.0 crit=100.0) sustained 31s
[THERMAL] STOPPED server: 96.0C sustained 62s (stop=95.0 crit=100.0) - Started stop-websdr.sh
[THERMAL] lockout: process reappeared at 94.0C - re-stopped [3 time(s) so far]
[THERMAL] recovered: 74.0C held below 75.0 - lockout cleared
```

```
grep THERMAL crash.log
```

Ostavite `thermal_mode` na `"log"` tjedan dana i pročitajte tu datoteku nakon teških trenutaka — potpune ponovne izgradnje, vrućeg poslijepodneva. Ako se ništa ne pojavi, vaš stroj nikada nije ni prišao. Zatim postavite način na `stop`.

Prije nego se pouzdate u njega, dokažite cijeli put jednom lažnim očitanjem iznad svojeg praga zaustavljanja:

```
"thermal_test_temp": 97
```

Čuvar ga tretira kao stvarnog, pa se `warn` → `stop` → zaključavanje → oporavak → ponovno pokretanje odvijaju na zahtjev. Uklonite redak (ili ga postavite na `null`) za povratak na stvarni senzor. S načinom `stop` ovo doista zaustavlja poslužitelj i odspaja vaše slušatelje, pa to učinite kad nikoga nema — ili testirajte u načinu `log`, gdje se vidi što bi se *dogodilo*, bez diranja ičega.

Rješavanje problema

Pogreške pri izgradnji

Problemi s ovisnostima

```
# Ubuntu: Install missing dependencies
sudo apt install --fix-broken

# Fedora: Install missing dependencies
sudo dnf install <package-name>
```

Neuspjela konfiguracija Mesona

```
# Clean and reconfigure
rm -rf build
meson setup build --buildtype=release
```

meson setup puca uz ModuleNotFoundError: No module named 'mesonbuild'

```
Traceback (most recent call last):
  File "/home/<user>/local/bin/meson", line 3, in <module>
    from mesonbuild.mesonmain import main
ModuleNotFoundError: No module named 'mesonbuild'
```

Izvorni kod nije kriv. Zaostala instalacija preko `pip install --user meson` ostavila je skriptu u `~/.local/bin`, koja na PATH-u dolazi prije `/usr/bin` i zato zaklanja ispravnu kopiju koju je instalirao upravitelj paketa. Nadogradnja distribucije (primjerice Ubuntu 24.04 → 26.04) prebacuje Python na novu verziju, stari `site-packages` u kojem je bio `mesonbuild` više nije na putanji uvoza, pa skripta pada prije nego išta napravi. Isto se može dogoditi i `ninji`.

Instalacijska skripta to prepoznaje i zaobilazi za vrijeme trajanja instalacije, uz upozorenje, ali sustav ipak popravite:

```
rm -f ~/.local/bin/meson
hash -r
meson --version          # mora ispisati verziju
```

Ako radije želite zadržati meson instaliran preko pipa, ponovno ga instalirajte za Python koji sustav sada ima:

```
python3 -m pip install --user --force-reinstall --break-system-packages meson
```

meson setup staje uz Clock skew detected

```
ERROR: Clock skew detected. File /home/pi/PhantomSDR-Plus/build/./meson.build has a time stamp
10392.2144s in the future.
```

S izvornim stablom nije ništa u redu: sistemski sat kasni za datotekama. meson i ninja odbijaju graditi kada je ulazna datoteka novija od trenutnog vremena jer ne mogu znati što je zastarjelo. Događa se na Raspberry Piju bez baterije u RTC ležištu — svako pokretanje kreće od zadnjeg poznatog vremena, pa gradnja pokrenuta prije nego se NTP uskladi vidi cijelo stablo datirano u budućnost — kao i na stablu raspakiranom ili kopiranom sa stroja koji žuri.

Najprije ispravite sat:

```
timedatectl                                # je li vrijeme točno? je li NTP usklađen?
sudo timedatectl set-ntp true
```

Pričekajte nekoliko sekundi pa ponovno pokrenite instalater. On to provjerava prije poziva mesaona i nudi poništavanje spornih vremenskih oznaka (`PHANTOM_FIX_CLOCK_SKEW=y|n`). Ručno, iz izvornog stabla:

```
find . -path ./build -prune -o -path ./git -prune -o -newermt now -print0 | xargs -0r touch
```

Pozadinski dio je izgrađen, ali web-stranice nema

Simptom je instalacija koja izgleda kao da je uglavnom uspjela: `build/spectrumserver` postoji, poslužitelj se pokreće, a preglednik ne dobiva ništa — jer `frontend/dist/` nikada nije proizveden. Potražite u ispisu instalacijskog programa:

```
[vite-plugin-top-level-await] missing field `type`
  at Compiler.printSync (node_modules/@swc/core/index.js:257:29)
error during build:
  Failed: Default
```

`vite-plugin-top-level-await` traži `@swc/core ^1.12.14`, pa čisti `npm install` razriješi na 1.16.0, čiji `printSync()` odbija sintaksno stablo koje mu dodatak predaje. To nema veze s vašom distribucijom — svaki instalacijski program sada pribada verziju koja radi, kao i `frontend/package.json`. Stroj postavljen prije tog pribadanja nastavlja raditi dok se njegov `node_modules` ne obriše, zbog čega se ovo pri ponovnoj instalaciji pojavi niotkuda.

Ako starije preuzimanje popravljate ručno:

```
cd frontend
npm pkg set 'overrides.@swc/core=1.15.33'
npm install
./build-all.sh
```

Pogreške pri izvođenju

Port je već zauzet

```
# Find process using port
sudo lsof -i :9002

# Kill process
sudo kill -9 <PID>
```

Pristup SDR-u odbijen

```
# Add user to plugdev group
sudo usermod -a -G plugdev $USER

# Reload groups (or log out and back in)
newgrp plugdev
```

Za **RX-888 MkII** to samo po sebi nije dovoljno — uređaj Cypress FX3 traži i udev pravila. Pokrenite priloženi pomoćni skript (instalira pravila i dodaje vas u `plugdev`), zatim se odjavite i prijavite te ponovno priključite uređaj:

```
./setup-rx888-udev.sh
```

Problemi sa zvukom

```
# Try different codec
# Edit config file and change:
audio_compression = "flac" # or "opus"

# Rebuild
cd frontend && npm run build && cd ..
```

Problemi s OpenCL-om

clinfo ne prikazuje uređaje

```
# Verify drivers are loaded
sudo clinfo

# Check OpenCL ICD files
ls /etc/OpenCL/vendors/

# Reinstall drivers (Intel example)
sudo apt remove --purge intel-opencl-icd
sudo apt install intel-opencl-icd
```

Performanse se nisu poboljšale

```
# Verify OpenCL is enabled in config
accelerator = "opencl"

# Try different device
# Add to config file:
[input.opencl]
device_id = 0 # Try 0, 1, 2, etc.
```

Mrežni problemi

Nema pristupa s drugih uređaja

```
# Check firewall
sudo ufw status
sudo ufw allow 9002/tcp

# Or disable firewall temporarily for testing
sudo ufw disable
```

Velika latencija

```
# Reduce waterfall size in config
waterfall_size = 512

# Increase buffer size
buffer_size = 16384

# Use Opus instead of FLAC
audio_compression = "opus"
```

Problemi sa SDR uređajem

RTL-SDR nije pronađen

```
# Check if device is recognized
lsusb | grep Realtek

# Test with rtl_test
rtl_test

# Try blacklisting DVB-T drivers
echo "blacklist dvb_usb_rtl28xxu" | sudo tee /etc/modprobe.d/blacklist-rtl.conf
sudo rmmod dvb_usb_rtl28xxu
```

SDRplay nije pronađen

`grep "RSP1A driver" logwebsdr.txt` pokazuje koji je upravljački program odabrao `start-rsp1a.sh`.

S otvorenim upravljačkim programom (libmirisdr-5, RSP1A driver: miri):

```
# The kernel's msi2500 driver must not hold the device
lsmod | grep msi2500 && sudo modprobe -r msi2500 msi001

# Test with SoapySDR
SoapySDRUtil --find="driver=soapyMiri"
```

Sa SDRplay API-jem (RSP1A driver: sdrplay):

```
# Check API installation
systemctl status sdrplay

# Restart API
sudo systemctl restart sdrplay

# Test with SoapySDR
SoapySDRUtil --find="driver=sdrplay"
```

Fobos ili Airspy HF+ nije pronađen

```
SoapySDRUtil --info | grep -i factories # must list fobos / airspyhf
SoapySDRUtil --find="driver=fobos"      # or driver=airspyhf
id | grep plugdev                       # log out and back in if missing
```

Ako upravljačkog programa nema na popisu, ponovno pokrenite `./setup-fobos.sh` ili `./setup-airspyhf.sh`: zaustavlja se i navodi razlog ako izgradnja ili provjera SoapySDR-a ne uspije.

Optimizacija performansi

Optimizacija procesora

```
# Adjust thread counts based on CPU cores
[server]
threads = 4 # Set to number of cores

[input]
fft_threads = 4 # Set to number of cores
```

Optimizacija memorije

```
# Reduce memory usage
[input]
waterfall_size = 512 # Lower value = less memory
fft_size = 65536      # Lower value = less memory
```

Optimizacija mreže

```
# Optimize for limited bandwidth
[input]
audio_compression = "opus" # More efficient than FLAC
waterfall_compression = "zstd" # Efficient compression
```

Ažuriranje PhantomSDR-Plusa

Od verzije 4.1.0 repozitorij donosi **update.sh**, alat koji instalirani prijamnik dovodi u skladu s objavljenim stablom **bez diranja datoteka koje ga čine vašom postajom**. Zamjenjuje ručno pisanu `git pull` skriptu koju su ranija izdanja ovog priručnika tražila da napišete i git mu uopće nije potreban: objavljeno stablo preuzima se kao tarball i uspoređuje s vašim datoteku po datoteku, pa radi jednako bilo da ste repozitorij klonirali, raspakirali `update.zip` ili stablo prekopirali s USB stika.

Ako vaša instalacija još nema update.sh

Starije stablo ne sadrži skriptu. Dohvatite je jednom — to je jedini korak cijelog ovog postupka koji ćete ikada obaviti ručno:

```
cd ~/PhantomSDR-Plus
curl -fLO https://raw.githubusercontent.com/svlbtl/PhantomSDR-Plus/main/update.sh
chmod +x update.sh
```

Od tada sve — izvorni kod, frontend, dokumentacija, instalacijske skripte i sam `update.sh` — dolazi preko alata.

Korak 1 — pogledajte što bi se promijenilo (ništa se ne zapisuje)

```
cd ~/PhantomSDR-Plus
./update.sh
```

Preuzima objavljeno stablo, uspoređuje ga s vašim i ispisuje izvještaj. Ne zapisuje baš ništa, pa ga je sigurno pokrenuti u bilo kojem trenutku, i dok je prijamnik u eteru. Izlazni je kod `0` kada ste ažurni i `10` kada ažuriranje čeka, pa vas cron posao može obavijestiti kada ima posla.

Korak 2 — primijenite ga

```
./update.sh --apply
```

Tri se vrste datoteka tretiraju različito, i upravo je u toj razlici cijela poanta:

Datoteke	Što se događa
config.toml, config-<prijamnik>.toml, markers.json, admin_config.json, autorun.json, frequencylist/, chat_history.txt, frontend/variant.json, frontend/site_information.json, zapisnici, build/, frontend/dist/	Nikada se ne diraju niti se ikada pojavljuju u pitanju. Upravo one od stroja čine vaš prijamnik.
start-*.sh, stop-websdr.sh, *.service jedinice, install*.sh, recompile.sh, setup_admin.sh, smeter_theme.sh, proxy.py, admin_server.py, thermal_guard.py, websdr_relay.py, frontend/src/bands-config.js, ogledne konfiguracije config.example*.toml	Uvijek se pita , jer su to datoteke koje operator s razlogom može biti izmijenio.
Sve ostalo	Ažurira se, nakon što se kopija stare datoteke spremi u arhivu sigurnosne kopije (vidi <i>Poništavanje ažuriranja</i>).

Za svaku datoteku iz srednje skupine prikazuju se razlike i nude tri izbora:

```
start-rx888mk2.sh [K]eep mine / [u]pstream / [b]oth (ENTER = Keep mine)
```

- **Keep mine** — vaša datoteka ostaje točno onakva kakva jest.
- **upstream** — instalira se nova inačica, a vaša se prethodno sigurnosno pohranjuje.
- **both** — nova se inačica zapisuje uz vašu kao `start-rx888mk2.sh.new`, da svoje izmjene prenesete kada vam odgovara.

Kako izgleda prvo pokretanje. Prvi put ne postoji zapis o tome iz koje inačice potječu vaše datoteke, pa vam se predočava svaka datoteka iz srednje skupine — desetak pitanja. Odgovarajte ovako:

Vaša situacija	Odgovor
Tu datoteku nikada niste mijenjali	u — uzmite novu inačicu. Uobičajen slučaj.
Mijenjali ste je (vlastiti RX888_ARGS, prikvačivanje na jezgre, prilagođena jedinica)	b — vaša ostaje, a nova dolazi uz nju kao <code><datoteka>.new</code> .
Niste sigurni	ENTER — vaša ostaje, ništa se ne gubi, usporedit ćete poslije.

Vaša konfiguracija u tome nikada ne sudjeluje: pitanja se uvijek tiču samo skripti i uslužnih jedinica.

`update.sh` u `.update-state/` bilježi inačicu svake datoteke koju instalira. Već od drugog pokretanja zato razlikuje datoteku koju ste **vi** mijenjali od one koja je naprosto stara, i zaustavlja se samo kod onih kojih ste se doista dotakli.

Prije nego išta zapiše zaustavlja prijamnik, administratorsku ploču, obrnuti proxy i WebSDR diversity relej **instalacije koju ažurira** — ono što posluhuje drugi direktorij navodi se i ostavlja raditi, pa se drugi klon može ažurirati dok prvi ostaje u eteru — a na kraju pokreće natrag točno ono što je zaustavio. Ako su se promijenile izvorne ili frontend datoteke, ponudit će da za vas pokrene `recompile.sh`. Ništa se nikada ne briše: datoteke kojih više nema u repozitoriju samo se prijavljuju, a uklanjaju se jedino ako to zatražite s `--prune`.

Poništavanje ažuriranja

```
./update.sh --restore LAST
```

Svaka datoteka koju ažuriranje prepíše najprije se sprema, u jednu arhivu s datumom po pokretanju, u običnu, vidljivu mapu `update-backups/`:

```
update-backups/phantomsdr-backup-20260923-164530.zip
```

Arhiva sadrži samo datoteke koje je to pokretanje zamijenilo, svaku na putanji koju ima u vašem stablu, pa se može kopirati na drugo računalo kao jedna datoteka, otvoriti bilo gdje ili raspakirati izravno preko instalacije. Gdje `zip` nije instaliran, zapisuje se kao `.tar.gz`. Čuvaju se tri najnovije. Bilo koja od ovih naredbi vraća ono što je ažuriranje zamijenilo:

```
./update.sh --restore LAST
./update.sh --restore phantomsdr-backup-20260923-164530.zip
```

Svaka arhiva nosi i vlastiti `restore.sh`, za slučaj da je problem sam `update.sh`: raspakirajte arhivu i dajte mu direktorij svoje instalacije, na primjer `./restore.sh /home/pi/PhantomSDR-Plus`. Tijekom vraćanja `update.sh` radi iz kopije izvan stabla, jer je obično i sam jedna od datoteka koje se vraćaju.

Sigurnosne kopije od prije 23. rujna 2026. nalaze se u skrivenoj mapi `.update-backups/<vremenska oznaka>/`; vraćaju se na isti način i ubrajaju se u tri koje se čuvaju.

Ostale mogućnosti

```
./update.sh --apply --yes      # nikada ne pita; svaka vaša izmijenjena datoteka OSTAJE
./update.sh --ref v5.0.0      # oznaka, grana ili commit umjesto trenutnog stabla
./update.sh --list-excludes   # ispisuje pravila "ne diraj" kako vrijede ovdje
./update.sh --verbose         # nabraja sve datoteke, ne samo prvih 40
```

Vlastita pravila "ne diraj" dodajete tako da u `update-exclude.txt` u korijenskoj mapi instalacije upišete jedan uzorak po retku.

Ako izgradnja padne na vrlo staroj instalaciji

`update.sh` ažurira datoteke, a ne sistemske pakete. Ako je vaše stablo toliko staro da izgradnja sada traži biblioteke kojih nemate, `recompile.sh` će stati s greškom prevoditelja ili mesona. To nije pokvareno ažuriranje — nedostaju ovisnosti:

```
./install.sh
```

Instalacijska skripta i sama se ažurira istim pokretanjem, a vaša konfiguracija preživljava i nju.

Ručno ažuriranje

Ako paket datoteka radije primjenjujete sami:

```
cd ~
unzip -o update.zip
cd PhantomSDR-Plus
./stop-websdr.sh
cp -a ~/update/. .
chmod +x *.sh
./recompile.sh
./start-rx888mk2.sh          # ili skripta za pokretanje vašeg prijamnika
```

Prvo sigurnosno pohranite svoju konfiguraciju — `config-*.toml`, `frontend/site_information.json`, `admin_config.json` i `markers.json` — jer paket datoteka ne može razlikovati vaše izmjene od onih iz izdanja. Upravo taj problem `update.sh` rješava.

Sigurnosna kopija i vraćanje

Datoteke koje treba sigurnosno kopirati

- Konfiguracijske datoteke: *.toml
- Podaci o lokaciji: frontend/site_information.json
- Oznake: markers.json
- Vlastite skripte: start-*.sh, stop-*.sh
- Povijest razgovora: chat_history.txt
- Pozadinska slika: frontend/src/assets/background.jpg

Naredba za sigurnosnu kopiju

```
cd ~/PhantomSDR-Plus
tar -czf phantomsdr-backup-$(date +%Y%m%d).tar.gz \
*.toml \
*.sh \
markers.json \
chat_history.txt \
frontend/site_information.json \
frontend/src/assets/background.jpg
```

Naredba za vraćanje

```
tar -xzf phantomsdr-backup-YYYYMMDD.tar.gz
```

Sigurnosna razmatranja

Konfiguracija vatrozida

```
# Allow only necessary ports
sudo ufw allow 9002/tcp
sudo ufw enable
```

Obratni proxy (neobavezno)

Razmislite o korištenju nginxa ili Apachea kao obratnog proxyja za:

- SSL/TLS enkripciju
- pridruživanje naziva domene
- raspodjelu opterećenja
- kontrolu pristupa

Ograničenja korisnika

Uredite config.toml:

```
[server]
max_users = 100 # Limit concurrent users
```

Kako doći do pomoći

Izvori

- **Dokumentacija:** ovaj vodič, README.md, USER_GUIDE.md
- **GitHub Issues:** <https://github.com/sv1btl/PhantomSDR-Plus/issues>
- **Demonstracija uživo:** <http://phantomsdr.no-ip.org:8900/>

Prijava problema

Kada prijavljujete problem, navedite:

1. operacijski sustav i verziju
2. model SDR uređaja
3. sadržaj konfiguracijske datoteke
4. poruke o pogreškama
5. potrošnju sistemskih resursa (CPU, RAM, GPU)

Podrška zajednice

- Prije otvaranja novih, provjerite postojeće GitHub issue prijave
 - Navedite detaljne podatke o svojoj postavi
 - Priložite zapise i poruke o pogreškama
 - Budite strpljivi i pristojni
-

Dodatak A: potpuni popis ovisnosti

Popis paketa za Ubuntu 24.04

```
build-essential
cmake
pkg-config
meson
libfftw3-dev
libwebsocketpp-dev
libflac++-dev
zlib1g-dev
libzstd-dev
libboost-all-dev
libopus-dev
libliquid-dev
git
util-linux (taskset – for the autorun spot reporter)
psmisc
wget
curl
rtl-sdr (for RTL-SDR)
airspy (for Airspy)
hackrf (for HackRF)
libclfft-dev (for OpenCL)
ocl-icd-opencl-dev (for OpenCL)
clinfo (for OpenCL)
```

Dodatak B: primjeri konfiguracije

Primjer 1: RTL-SDR za VHF/UHF

```
[input]
sps = 2048000
frequency = 145000000
signal = "iq"

[input.driver]
format = "u8"

[input.defaults]
frequency = 145500000
modulation = "FM"
```

Primjer 2: RX-888 mk2 za KV (0-30 MHz)

Prijamnik koji koristi većina sysopa. Ovo je cijeli [input] odjeljak, a ne isječak, s vrijednostima iz config-rx888mk2.toml koji dolazi uz repozitorij.

```
[input]
sps = 60000000          # 0-30 MHz izravnim uzorkovanjem
fft_size = 4194304      # vidjeti napomenu ispod
fft_threads = 8
brightness_offset = -9  # negativnije ako slap pokazuje crne mrlje
frequency = 0           # osnovni pojas: RX-888 uzorkuje od DC-a
signal = "real"         # ne "iq" - izravno uzorkovanje daje realni tok
accelerator = "opencl"  # "none" ako nema OpenCL okruženja
audio_sps = 12000
audio_compression = "flac"
waterfall_size = 1024
waterfall_compression = "zstd"
smeter_offset = 5
analog_smeter_offset = 5

[input.driver]
name = "stdin"
format = "s16"

[input.defaults]
frequency = 7120000
modulation = "LSB"
```

Napaja ga rx888_stream, što za vas obavlja start-rx888mk2.sh:

```
rx888_stream -f /path/to/SDDC_FX3.img -s 60000000 -g 50 -a 0 -m low --pga -d -r -o - \
| build/spectrumserver --config config-rx888mk2.toml
```

O fft_size: pri 60 MSPS ispravna je veličina 4194304. 8388608 udvostručuje razlučivost slapa, a s njom i memoriju te procesorsku cijenu svake transformacije; na većini strojeva oštrije binove plaćate izgubljenim okvirima. Počnite s 4194304 i povećavajte samo ako poslužitelj ima obilje rezerve.

Primjer 3: HackRF za širokopojasni FM

```
[input]
sps = 10000000
frequency = 100900000
signal = "iq"

[input.driver]
format = "s8"

[input.defaults]
frequency = 100900000
modulation = "WBFM"
```

Instalacija je gotova! Sada biste trebali imati potpuno funkcionalan PhantomSDR-Plus poslužitelj.

73 de SV1BTL & SV2AMK

POGLAVLJE 3

Postavljanje administratorske ploče

ADMIN_PANEL_SETUP.md

Administratorska ploča PhantomSDR-Plusa — Vodič za postavljanje

⚠ **Sigurnosna napomena:** držite ovu administratorsku ploču na kućnoj mreži ili iza VPN-a. Javno izlaganje administratorskog porta ne preporučuje se bez odgovarajućeg pojačanja autentifikacije.

Pregled

Administratorska ploča sastoji se od dvije Python usluge:

Usluga	Datoteka	Uloga
Administratorska ploča	<code>admin_server.py</code>	Flask web-aplikacija, veže se na 127.0.0.1 (samo interno)
Obratni proxy	<code>proxy.py</code>	Izlaže i SDR i administratorsku ploču na jednom javnom portu

Objе usluge čitaju konfiguraciju iz `admin_config.json`, koju zapisuje `setup_admin.sh`. **Nemojte izravno mijenjati konstante portova u Python datotekama** — svi portovi i IP adresa SDR poslužitelja pohranjeni su u `admin_config.json`.

Što vam administratorska ploča donosi

- **Nadzorna ploča** — stanje poslužitelja uživo, CPU/RAM, najzahtjevniji procesi, nedavni ispis zapisnika, terminal
- **Uređivač konfiguracije** — pregled, uređivanje i spremanje bilo koje `.toml`, `.sh`, `.json`, `.h` ili `.cpp` datoteke u vašoj instalaciji
- **Preglednik zapisnika** — praćenje bilo koje datoteke zapisnika u stvarnom vremenu (SDR poslužitelj, administratorska ploča, RADE, rušenja, autorun, proxy)
- **Oznake** — pregled i uređivanje frekvencijskih oznaka
- **Povijest razgovora** — pregled zapisnika WebSDR razgovora, potpuno brisanje ili uklanjanje pojedinačnih poruka bez ponovnog pokretanja poslužitelja
- **Poruka na slapu** — emitiranje trajne poruke svim spojenim korisnicima, vidljive u stvarnom vremenu na prikazu slapa
- **Korisnici** — popis spojenih slušatelja uživo s njihovom frekvencijom, načinom rada, trajanjem veze i gumbom Kick
- **Brisanje poruka razgovora** — gumb Delete koji odmah uklanja tu jednu poruku iz zapisnika
- **Emitirane poruke na slapu** — omogućuje slanje trajnog tekstualnog natpisa na slap svakog spojenog korisnika
- **Spot Reporting** — pokretanje/zaustavljanje autorun dekodera FT8/FT4/JS8/WSPR, odabir pojaseva/ načina rada i prijava spotova na PSK Reporter / wsprnet (prema zadanom isključeno), s dva brojača:

pločice po dekoderu za tekući rad i ukupan zbroj od početka uz svaki potvrdni okvir pojasa/načina rada

- **Grafikoni** — frekvencija procesora, opterećenje procesora, temperatura procesora i povezani korisnici, prikazani kroz zadnjih 15 minuta / 1 sat / 4 sata / 12 sati / 24 sata
- **Toplinska zaštita (Thermal Guard)** — zaustavlja poslužitelj ako se procesor pregrije i ponovno ga pokreće kad se ohladi, s pragovima izvedenima iz vašeg vlastitog procesora; radi s bilo kojim načinom pokretanja/zaustavljanja i isporučuje se u načinu samo zapisivanja, pa ne čini ništa dok ga ne uključite
- **Postavke** — promjena administratorske lozinke, osnovnog direktorija SDR-a, naziva procesa i javnog porta

Preduvjeti

- PhantomSDR-Plus već instaliran i pokrenut
- Python 3.8+

NAPOMENA

Uobičajeno to za vas obavlja instalacijska skripta. `./install.sh` — kao i svaka od četiri distribucijske skripte, obje RADE skripte i skripta poslužitelja statistike — postavlja ploču prema zadanim, uz prethodnu instalaciju `pipa` upraviteljem paketa sustava ako nedostaje. Ova stranica opisuje što se u tom koraku događa i što učiniti ako ste ga preskočili ili želite nešto naknadno promijeniti.

Python biblioteke ploče (`flask`, `psutil`, `aiohttp`, `tomli-w`) instalira `setup_admin.sh`; ako to zakaže, ispisuje točnu naredbu koju treba pokrenuti.

Korak 1 — Datoteke

Postavite ove četiri datoteke u svoj PhantomSDR-Plus direktorij:

```
admin_server.py
manage_admin.sh
setup_admin.sh
proxy.py
```

Postavite skripte ljske kao izvršne:

```
chmod +x setup_admin.sh manage_admin.sh
```

Korak 2 — Pokrenite skriptu za postavljanje

```
./setup_admin.sh
```

Skripta će:

1. Provjeriti je li instaliran Python 3
2. Provjeriti postoje li `admin_server.py` i `manage_admin.sh`
3. Zabilježiti `127.0.0.1` kao `sdr_host` u konfiguraciji (vidi [Postavka sdr_host](#))

4. Zatražiti tri broja porta. Svaki upit nudi zadanu vrijednost u uglatim zagrada koje prihvaća običan Enter, pa je uobičajeno postavljanje pitanje triju pritisaka tipke:
5. **Port spectrumservera** — port na kojem sluša vaš SDR poslužitelj (zadano 8900, upravo ono što donosi svaki `config-*.toml` u repozitoriju)
6. **Interni port administratorske ploče** — gdje se `admin_server.py` lokalno veže (zadano 3000)
7. **Javni port proxyja** — jedini vanjski port koji objedinjuje SDR + administraciju (zadano 8902)

Neispravni odgovori odbijaju se i pitanje se ponavlja, ali samo pet puta — nakon toga uzima se zadana vrijednost. Izvođenje čiji ulaz nije terminal (kroz cijev, cron, bez nadzora) odmah uzima zadane vrijednosti umjesto da čeka unos koji nikad neće stići.

1. Instalirati `flask`, `psutil`, `aiohttp` i `tomli-w` putem `pipa`
2. Dodijeliti naredbi `ss` mogućnost `cap_net_admin` (pomoćni put značajke Kick Users, potreban samo ako ploča radi bez proxyja)
3. Pitati koja skripta pokreće, a koja zaustavlja prijamnik — preko njih ploča upravlja poslužiteljem, a jednako tako i toplinska zaštita
4. Pitati koliko daleko toplinska zaštita procesora smije ići sama — `log`, `throttle`, `stop` ili `stop+restart` — i ponuditi pokretanje `setup-cpufreq-perms.sh` kako bi `throttle` faza radila bez roota (vidi [THERMAL_GUARD.md](#))
5. Zapisati `admin_config.json` sa svim postavkama
6. Ponuditi instalaciju dviju `systemd` jedinica (ploča + proxy) — pokretanje pri dizanju sustava i ponovno pokretanje nakon pada. **Preporučeno i zadano** (dovoljan je Enter): zaštita od pregrijavanja procesora radi samo dok radi ploča, pa bez jedinica ponovno dizanje sustava ostavlja računalo nezaštićenim. Korak se automatski preskače ondje gdje `systemd` nije PID 1 (kontejneri, WSL1, OpenRC) ili gdje nemate `sudo`.

Nakon postavljanja otvorite preglednik putem proxyja:

```
http://YOUR_SERVER_IP:<proxy_port>/admin
```

Zadana lozinka: **admin**

⚠ **Odmah promijenite lozinku** — pri prvoj prijavi idite na Settings. Čarobnjak prvog pokretanja vodit će vas kroz postavljanje SDR direktorija, naziva procesa, javnog porta i nove lozinke.

Korak 3 — Pokretanje / zaustavljanje / ponovno pokretanje

Postoje dva načina za pokretanje ploče i proxyja. **Odaberite jedan i držite ga se** — upozorenje na kraju ovog odjeljka objašnjava što se događa ako ih miješate.

Način B (systemd) snažno se preporučuje svugdje gdje je `systemd` dostupan. Zaštita od pregrijavanja procesora radi unutar ploče, pa ploča koja umre u 03:00 ili računalo koje se ponovno digno ostavlja procesor nezaštićenim dok ga netko ručno ne pokrene. `setup_admin.sh` sada nudi jedinice po zadanome. Način A koristite kad `systemd`a nema — kontejneri, WSL1, OpenRC/sysvinit — ili kad namjerno želite sami nadzirati stvari.

Način A — `manage_admin.sh` (bez roota, ništa se ne instalira)

```
./manage_admin.sh start      # start admin panel and proxy
./manage_admin.sh stop      # stop both
./manage_admin.sh status    # show running status and PIDs
```

Namjerno **nema restart**. Prije je ubijao ploču i odmah pokretao vlastitu kopiju, što se sudara sa systemd-om na svakom računalu koje koristi Način B. Za ponovno pokretanje po Načinu A izvršite `stop` pa `start` — vidi [Ponovno pokretanje ploče](#) niže.

Jedna skripta upravlja **objema** procesima. Prije pokretanja proxyja provjerava je li instaliran `aiohttp` i ispisuje jasnu pogrešku s naredbom za ispravak ako ga nema.

Ništa se ne pokreće samo od sebe: nakon ponovnog pokretanja sustava, ili ako se ploča sruši, pokrećete je ručno. To je u redu za prijamnik koji nadzirete, ali imajte na umu da toplinska zaštita procesora radi unutar ploče: dok je ploča ugašena, ništa ne štiti stroj (vidi [THERMAL_GUARD.md](#)).

Način B — systemd jedinice (pokreću se pri dizanju sustava, ponovno se pokreću nakon pada)

Repozitorij donosi dvije gotove jedinice, `phantomsdr-admin.service` i `phantomsdr-proxy.service`. U obje prilagodite `User=` i putanje, zatim:

```
sudo cp phantomsdr-admin.service phantomsdr-proxy.service /etc/systemd/system/
sudo systemctl daemon-reload
./manage_admin.sh stop      # free the ports first
sudo systemctl enable --now phantomsdr-admin
sudo systemctl enable --now phantomsdr-proxy
```

Svakodnevne naredbe tada postaju:

```
sudo systemctl start    phantomsdr-admin
sudo systemctl stop     phantomsdr-admin
sudo systemctl restart  phantomsdr-admin
systemctl status        phantomsdr-admin
sudo journalctl -u phantomsdr-admin -f
```

Iste četiri s `phantomsdr-proxy`, ili obje odjednom: `sudo systemctl restart phantomsdr-admin phantomsdr-proxy`.

`admin.log` i `proxy.log` rade točno kao i prije — jedinice pišu u iste datoteke.

Jedno napravite jednom: instalirajte `tmpfiles` pravilo iz `tmpfiles/phantomsdr-logs.conf` (prvo u njemu izmijenite putanje i korisnika). `systemd` te dvije datoteke stvara kao `root` prije nego se spusti na `User=`, pa gumb **Clear Logs** na ploči na njima pada uz `Permission denied` — vidi niže „Clear Logs javlja `Permission denied`“. `setup_admin.sh` pravilo instalira umjesto vas.

Jedinica ploče nosi `SupplementaryGroups=cpufreq`, što `throttle` fazi toplinske zaštite omogućuje snižavanje takta procesora bez da ploča radi kao `root`. Tu grupu prvo stvorite s `./setup-cpufreq-perms.sh`, inače `systemd` neće pokrenuti jedinicu; obrišite taj redak ako ne koristite `throttle` fazu.

Povratak na način A:

```
sudo systemctl disable --now phantomsdr-admin
sudo systemctl disable --now phantomsdr-proxy
```

⚠ **Nemojte ih miješati.** Uz uključene jedinice, `./manage_admin.sh stop` systemd poništi pet sekundi kasnije, a `./manage_admin.sh start` daje vam drugu, neupravljanu kopiju koja se sa systemd-ovom otima za isti port. (`restart` je upravo zbog toga uklonjen iz skripte; sada ispisuje pogrešku i upućuje vas na `systemctl`.) `./manage_admin.sh status` samo čita pa ostaje koristan u oba slučaja.

Ponovno pokretanje ploče

Ostatak ovog dokumenta na više mjesta kaže „ponovno pokrenite ploču“. To znači ono što odgovara vašem načinu:

```
# Način B – systemd jedinice (uobičajena postava)
sudo systemctl restart phantomsdr-admin phantomsdr-proxy

# Način A – jedinice nisu instalirane
./manage_admin.sh stop && ./manage_admin.sh start
```

Jedinica dolazi s `KillMode=process` i ta je linija ključna. Ako SDR pokrećete iz ploče, prijatelj je dijete administratorske jedinice i nasljeđuje njezinu cgroup. Uz systemdovu zadanu vrijednost `KillMode=control-group`, ponovno pokretanje ploče povuklo bi sa sobom `spectrumserver`, njegov `watchdog` i autorun demon — nakon što bi prethodno stalo punih 90 sekundi vremena čekanja na zaustavljanje. Uz `KillMode=process` systemd signalizira samo ploči, pa `sudo systemctl restart phantomsdr-admin` ostavlja zauzet prijatelj na eteru.

Nadogradnja jedinice instalirane prije v4.1.0. Starije datoteke jedinice nemaju tu liniju, a ponovno pokretanje ploče je neće dodati: `systemctl restart` ponovno pokreće program, ali ne mijenja njegovu konfiguraciju. Uredite *instaliranu* kopiju — ona u repozitoriju samo je predložak koji systemd nikada ne čita:

```
sudo nano /etc/systemd/system/phantomsdr-admin.service
```

Dodajte `KillMode=process` u odjeljak `[Service]`, uz `Restart=always`. Zatim neka systemd ponovno pročita datoteku i provjerite:

```
sudo systemctl daemon-reload
systemctl show phantomsdr-admin -p KillMode
```

Druga naredba mora ispisati `KillMode=process`. Ako i dalje ispisuje `control-group`, systemd radi sa spremljenom kopijom — `systemctl show phantomsdr-admin -p NeedDaemonReload` javlja `yes` dok je ponovno učitavanje na čekanju. Vodeći razmaci u datotekama jedinica se zanemaruju, pa uvlaka nije važna.

Korak 4 — Otvorite port na vatrozidu

Na vatrozidu otvorite samo **javni port proxyja**. Interni port administratorske ploče nije potrebno izlagati prema van:

```
sudo ufw allow <proxy_port>
```

Korak 5 — Značajka Kick Users

Stranica Users prikazuje svakog trenutno spojenog slušatelja. Svaki redak prikazuje korisnikovu IP adresu, ugođenu frekvenciju, način rada i trajanje veze. Da biste odspojili jednog korisnika, kliknite gumb

Kick u njegovu retku — prekida se samo ta TCP veza. Svi ostali slušatelji ostaju spojeni i ništa ne primjećuju.

Prekid je trenutčan i precizan: poslužitelj se ne pokreće ponovno, nikome drugom se zvuk ne prekida, a odspojeni korisnik može se odmah ponovno spojiti (značajka služi uklanjanju problematičnih ili zaglavljenih veza, a ne zabrani pristupa).

U pozadini prekid koristi `ss -K dst <IP> dport <port>` za rušenje te konkretne TCP utičnice. To zahtijeva Linux mogućnost koju skripta za postavljanje dodjeljuje automatski. Ako ste preskočili postavljanje ili gumb javlja pogrešku, primijenite je ručno:

```
sudo setcap cap_net_admin+ep $(which ss)
# Verify:
getcap $(which ss)    # should show: cap_net_admin=ep
```

Konfiguracija — admin_config.json

Sva konfiguracija izvođenja nalazi se u `admin_config.json` u vašem PhantomSDR-Plus direktoriju. `setup_admin.sh` zapisuje početnu datoteku; kasnije promjene mogu se raditi putem stranice Settings ili izravnim uređivanjem datoteke.

Ključna polja koja zapisuje `setup_admin.sh`:

Ključ	Opis
<code>port</code>	Interni port administratorske ploče (ondje se veže <code>admin_server.py</code>)
<code>public_port</code>	Port spectrumservera (koristi ga stranica Users za upit na <code>/users</code>)
<code>proxy_port</code>	Javni port na kojem sluša proxy
<code>sdr_host</code>	Računalo preko kojeg proxy doseže spectrumserver — <code>127.0.0.1</code> (vidi dolje)
<code>password_hash</code>	SHA-256 sažetak administratorske lozinke
<code>sdr_base_dir</code>	Putanja do vaše PhantomSDR-Plus instalacije
<code>sdr_process_name</code>	Naziv procesa za nadzor (zadano: <code>spectrumserver</code>)
<code>start_script</code> / <code>stop_script</code>	Skripte kojima ploča pokreće i zaustavlja SDR poslužitelj — preko njih djeluje i toplinska zaštita

Više prijemnika na jednom računalu? Postavite `start_script` na `start-all.sh`. Skripta za zaustavljanje već zaustavlja sve prijemnike, a Restart i toplinska zaštita koriste skriptu za pokretanje; sa `start-all.sh` vraćaju sve prijemnike iz `receivers.toml` umjesto samo glavnoga. Stranica Users, broj slušatelja i izbacivanje tada također obuhvaćaju svaki prijemnik. Vidi [Više prijemnika](#).

Ključevi toplinske zaštite (svi neobavezni — ploča ih zapisuje kad spremite postavke Thermal Guarda, a oni koji nedostaju koriste ove zadane vrijednosti):

Ključ	Zadano	Opis
<code>thermal_enabled</code>	<code>true</code>	Glavni prekidač zaštite
<code>thermal_mode</code>	<code>"log"</code>	<code>log</code> (samo zapisuje, nikad ne djeluje) · <code>throttle</code> · <code>stop</code> · <code>stop+restart</code>
<code>thermal_warn</code>	<code>null</code>	Temperatura upozorenja °C; <code>null</code> = izvesti iz kritičnog praga vašeg procesora
<code>thermal_throttle</code>	<code>null</code>	Temperatura na kojoj se snižava gornja granica takta; <code>null</code> = automatski
<code>thermal_stop</code>	<code>null</code>	Temperatura na kojoj se poslužitelj zaustavlja; <code>null</code> = automatski
<code>thermal_resume</code>	<code>null</code>	Temperatura ispod koje procesor mora pasti za oporavak; <code>null</code> = automatski
<code>thermal_sustain_s</code>	<code>60</code>	Sekunde koliko se temperatura zaustavljanja mora zadržati prije djelovanja
<code>thermal_warn_sustain_s</code>	<code>30</code>	Sekunde koliko se moraju zadržati temperature upozorenja/throttlea
<code>thermal_resume_s</code>	<code>300</code>	Sekunde ispod temperature oporavka prije otpuštanja zaključavanja
<code>thermal_max_stops_hour</code>	<code>2</code>	Toplinskih zaustavljanja na sat prije isključivanja automatskog ponovnog pokretanja
<code>thermal_test_temp</code>	<code>null</code>	Lažna temperatura za testiranje; <code>null</code> = koristi stvarni senzor

Promjene stupaju na snagu pri sljedećem uzorku — u nekoliko sekundi, bez ponovnog pokretanja.

Za promjenu portova nakon početnog postavljanja uredite `admin_config.json` i ponovno pokrenite:

```
sudo systemctl restart phantomsdr-admin phantomsdr-proxy # B
./manage_admin.sh stop && ./manage_admin.sh start # A
```

Postavka `sdr_host`

`proxy.py` se na `spectrumserver` spaja preko povratne (loopback) adrese — `sdr_host` je `127.0.0.1` u `admin_config.json`, gdje ga upisuje `setup_admin.sh`. Ne biste to trebali morati mijenjati.

Postavite stvarnu adresu samo ako `spectrumserver` radi na **drugom računalu** od proxyja. Zatim [Ponovno pokretanje ploče](#).

Promjena: starije verzije zahtijevale su da `sdr_host` bude LAN IP računala, jer je `spectrumserver` zatvarao WebSocket veze koje dolaze s povratne adrese. Taj je filter uklonjen, pa lokalne veze rade normalno. Dvije posljedice: otvaranje `http://localhost:<port>` na samom poslužiteljskom računalu sada prikazuje cijelo sučelje (prije bi se stranica učitala, ali ostala prazna), a proxy se više ne kvari kad DHCP dodijeli računalu novu IP adresu. Ako nadograđujete, a vaš `admin_config.json` još sadrži LAN IP, promijenite ga u `127.0.0.1` ili ponovno pokrenite `setup_admin.sh`.

proxy.py — što je i kada vam treba

`proxy.py` smješta i SDR poslužitelj i administratorsku ploču na **jedan javni port**, usmjeravajući prema prefiksu putanje:

Putanja	Usmjerava se na
/admin*	Administratorsku ploču (localhost:<port>)
Sve ostalo	Spectrumserver (<sdr_host>:<public_port>)

Bez proxyja morate izložiti dva porta zasebno. S proxyjem izlažete samo jedan.

Proxy također uklanja ograničenje veličine WebSocket poruke od 4 MB (važno za velike FFT okvire s RX-888 pri 60 MSPS), prosljeđuje stvarne IP adrese klijenata putem X-Forwarded-* zaglavlja i održava dugotrajne veze kroz NAT pomoću otkućaja svakih 30 sekundi.

Promjena portova nakon postavljanja

Uredite `admin_config.json` izravno:

```
{
  "port":      3000,
  "public_port": 9001,
  "proxy_port": 9002,
  "sdr_host":   "127.0.0.1"
}
```

Zatim ponovno pokrenite obje usluge:

```
sudo systemctl restart phantomsdr-admin phantomsdr-proxy # B
./manage_admin.sh stop && ./manage_admin.sh start      # A
```

Ako uz to trebate da se `admin_server.py` pri pokretanju veže na drugi port (npr. u systemd usluzi), to možete nadjačati varijablom okruženja:

```
ADMIN_PORT=3000 python3 admin_server.py
```

Prema zadanom, `admin_server.py` veže se samo na `127.0.0.1`. Za samostalno pokretanje bez proxyja (razvoj/testiranje) vežite ga na sva sučelja:

```
ADMIN_BIND=0.0.0.0 python3 admin_server.py
```

Korisne ručne naredbe

```
# Start admin panel manually (foreground)
python3 ~/PhantomSDR-Plus/admin_server.py

# Kill the admin panel
pkill -f admin_server.py

# Kill the proxy
pkill -f proxy.py

# Free a port that is stuck in use
sudo fuser -k 3000/tcp

# Check what is listening on a port
ss -tlnp | grep 3000
```

Datoteke zapisnika

Datoteka	Sadržaj
admin.log	Ispis administratorske ploče
proxy.log	Natpis pri pokretanju proxyja + jedan redak pristupa po proslijeđenom zahtjevu
logwebsdr.txt	Zapisnik pokretača i nadzornika — pokretanje/zaustavljanje, upravljački program prijavnika, registracija na websdr.org
spectrumserver.log	Izlaz samog procesa prijavnika — postavljanje FFT-a i OpenCL-a, veze pojedinih klijenata, pogreške. Najveća od ovih datoteka; rotira se, pa uz nju može stajati spectrumserver.log.1
rade.log	Zapisnik RADE/FreeDV sidecara
autorun.log	Zapisnik dekoderskog daemona — spotovi, SNR, drift
crash.log	Izveštaji o rušenju i toplinski događaji: [CRASH] / [TERMINATE] backtraceovi, [EXIT] bilješke i [THERMAL] retci zaštite. Uobičajeno prazan — sve što se u njemu nađe vrijedi pročitati

Sve se to može čitati iz kartica **Log Viewer** na ploči (LOGWEBSDR, SPECTRUMSERVER, ADMIN, RADE, CRASH, AUTORUN, PROXY). Preglednik prikazuje zadnjih 150 redaka odabrane kartice i osvježava se svake 3 sekunde kad je uključeno automatsko osvježavanje; čita samo kraj datoteke, pa njezina veličina nije važna.

Rotiranje proxy.log i admin.log

proxy.log bilježi svaki proslijeđeni zahtjev, a admin.log svaki zahtjev koji stigne do same ploče. Nijedan se ne rotira prema zadanom. Ispitivanja nadzorne ploče nekoć su dominirala objema datotekama — /admin/api/status okida svakih ~3 sekunde dok je kartica otvorena — ali sada se filtriraju iz oba zapisnika (vidi niže), pa rotacija ograničava spor rast, a ne poplavu.

Repozitorij donosi logrotate konfiguraciju u logrotate/phantomsdr (dnevno, ili ranije ako zapisnik prijeđe 10 MB; čuva 7 gzip arhiva u logproxy/ kako ne bi zatrpavale korijen projekta). **Ovisi o instalaciji — otvorite je i prije instalacije promijenite tri putanje zapisnika, putanju olddir i korisnika su da odgovaraju vašem računalu, zatim:**

```
sudo cp logrotate/phantomsdr /etc/logrotate.d/phantomsdr
sudo logrotate -d /etc/logrotate.d/phantomsdr # dry run, should list all three logs
```

logrotate/phantomsdr u repozitoriju i /etc/logrotate.d/phantomsdr dvije su **neovisne kopije** — cp ih ne povezuje. Uređivanje datoteke u repozitoriju ne mijenja ništa na pokrenutom sustavu dok ponovno ne izvršite gornji sudo cp. Logrotate zapravo čita samo kopiju u /etc.

Konfiguracija koristi copytruncate, što je nužno: manage_admin.sh pokreće oba procesa s >> <log> i nijedan ponovno ne otvara svoj standardni izlaz, pa bi ih rotacija temeljena na preimenovanju ostavila da pišu u rotiranu datoteku dok bi novi zapisnik zauvijek ostao prazan.

Arhive završavaju u logproxy/ (stvara se automatski zahvaljujući createolddir) i nose nazive proxy.log.1.gz ... proxy.log.7.gz, najnovija prva. Jednu pročitajte naredbom zcat logproxy/proxy.log.1.gz | less, ili pretražite sve odjednom pomoću zgrep "pattern" logproxy/*.gz.

Gumb **Clear Logs** na ploči prazni **svaku** datoteku iz gornje tablice, uključujući autorun.log i proxy.log (CLEAR_EXCLUDE u admin_server.py je prazan). Imajte na umu što to stoji kod autorun.log: njegova povijest dekodiranja jedini je zapis o tome što je i kada primljeno i nije je moguće rekonstruirati —

pražnjenje preživi samo ono što je logrotate već premjestio u `logproxy/`, pa se gubi sve dekodirano od zadnje rotacije. Pražnjenje uz dekodirano u radu je sigurno: daemon datoteku drži otvorenu s `O_APPEND` i nastavlja pisati u isti inode. Obavijest navodi što je ispražnjeno, a ako se u neku datoteku nije moglo pisati — recimo `proxy.log` koji na instalaciji s odvojenim korisnicima pripada drugom korisniku — navodi tu datoteku i pogrešku umjesto da javi uspjeh te ostavlja prikaz tog zapisnika na zaslonu umjesto da ga isprazni. Datoteke se prazne na mjestu, nikada se ne brišu: i `manage_admin.sh` i `systemd` jedinice otvaraju ih s `O_APPEND`, pa bi obrisana datoteka nastavila puniti nevidljiv inode sve do ponovnog pokretanja usluge.

Uspješna ispitivanja filtriraju se iz **oba** zapisnika pristupa, `QuietPollFilterom` u `proxy.py` i onim u `admin_server.py`. Nadzorna ploča poziva `/admin/api/status` svake 3 sekunde, a `/admin/api/logs`, `/admin/api/autorun/status`, `/admin/api/users`, `/admin/api/graph-stats` i `/admin/api/chat` svakih 5 sekundi; nefiltrirani bi činili ~95 % obiju datoteka. Izbacuju se samo obični 200 na tim putanjama — svaki drugi status, putanja ili metoda bilježi se kao i prije, pa neuspjelo ispitivanje ostaje vidljivo, a radnje koje mijenjaju stanje (`kick`, `autorun/start`, ...) idu zasebnim putanjama i uvijek se zapisuju. Jedina namjerna iznimka je `/admin/api/logs/clear`, filtrirana u obje datoteke: njezin vlastiti redak pristupa piše se *nakon* pražnjenja, pa je bez filtra svako uspješno pražnjenje ostavljalo jedan svjež redak u zapisniku koji je upravo ispraznilo, a gumb je izgledao neispravno.

Budući da je `proxy.log` zapisnik pristupa, sadrži IP adrese posjetitelja i nizove `user-agenta` — imajte to na umu prije nego što ga podijelite tražeći pomoć.

„Clear Logs“ javlja `Permission denied`

Na stroju gdje su `systemd` jedinice instalirane na čistom stablu, gumb može vratiti:

```
x Could not clear admin.log ([Errno 13] Permission denied: /home/you/PhantomSDR-Plus/admin.log),
proxy.log ([Errno 13] Permission denied: /home/you/PhantomSDR-Plus/proxy.log)
```

Uvijek su to te dvije datoteke i nikada ostalih pet, jer ih jedine stvara sam `systemd`.

`StandardOutput=append`: otvara PID 1 **prije** nego se spusti na `User=`, pa se zapisnik koji još ne postoji stvara kao `root:root 0644`. Ploča radi kao vaš korisnik i ne može ga isprazniti. Ostalih pet zapisnika stvaraju skripte za pokretanje, koje ionako rade kao vaš korisnik, pa se brišu normalno. Ondje gdje su te dvije datoteke starije od jedinica — stvorio ih je `manage_admin.sh` s `>>` — vlasnik je već ispravan i gumb radi; zato se problem pojavljuje samo na svježoj instalaciji koja je odmah krenula sa `systemd` jedinicama.

Provjerite s `ls -l admin.log proxy.log`: one koje ne uspijevaju pokazuju `root root`. Popravite ih:

```
sudo chown "$(id -un):$(id -gn)" admin.log proxy.log
sudo chmod 664 admin.log proxy.log
```

Ponovno pokretanje nije potrebno — `systemd` i dalje piše kroz već otvoreni opisnik datoteke, a dopisivanje ne mijenja vlasništvo. Pritisnite **Clear Logs** ponovno i obje se prazne.

Nemojte umjesto toga obrisati datoteke. Obje su otvorene s `O_APPEND`; brisanjem jedne `systemd` nastavlja puniti nevidljivi inode sve do ponovnog pokretanja usluge, a datoteka koja se ponovno pojavi opet pripada rootu.

Da se ne bi vraćalo, instalirajte `tmpfiles` pravilo koje repozitorij isporučuje u `tmpfiles/phantomsdr-logs.conf`. Ono pri svakom dizanju sustava, prije pokretanja jedinica, ponovno stvara oba zapisnika s ispravnim vlasnikom, pa se uklonjeni zapisnik — ručno čišćenje, ponovna instalacija, premješten instalacijski direktorij — nikada ne vrati u vlasništvo roota. **Pravilo ovisi o stroju: otvorite ga i najprije promijenite dvije putanje i `user:group` da odgovaraju vašem sustavu**, a zatim:

```
sudo cp tmpfiles/phantomsdr-logs.conf /etc/tmpfiles.d/99-phantomsdr-logs.conf
sudo systemd-tmpfiles --create /etc/tmpfiles.d/99-phantomsdr-logs.conf
```

Kao i kod `logrotate/phantomcdr`, datoteka u repozitoriju i kopija u `/etc` neovisne su — uređivanje one u repozitoriju ne mijenja ništa dok ponovno ne pokrenete `cp`. Na `systemd 254` i novijem pravilo popravljiva i vlasništvo i prava već postojećeg zapisnika, pa rješava i trenutni kvar i buduće; na starijim verzijama gumb odblokira upravo `chown` iznad.

`setup_admin.sh` sve ovo obavlja umjesto vas pri instalaciji jedinica, pa postava napravljena skriptom nikada ne naiđe na ovo. Gornji koraci su za ručno instalirani par ili za instalaciju stariju od koraka sa vlasništvom zapisnika u skripti.

Logrotate ne vraća problem: isporučena konfiguracija koristi `copytruncate`, koji zadržava isti inode, a time i istog vlasnika.

Sažetak pristupa

Postava	SDR	Administratorska ploča
Bez proxyja	<code>http://YOUR_IP:<public_port></code>	<code>http://YOUR_IP:<port>/admin</code>
S proxyjem	<code>http://YOUR_IP:<proxy_port></code>	<code>http://YOUR_IP:<proxy_port>/admin</code>

Brisanje poruka razgovora

Stranica Chat History navodi svaku poruku u trenutnom zapisniku razgovora. Svaki unos ima gumb **Delete** koji odmah uklanja tu jednu poruku iz zapisnika — bez ponovnog pokretanja poslužitelja.

Kako to radi:

- Administratorska ploča prepisuje datoteku zapisnika razgovora na mjestu, uklanjajući samo redak odabrane poruke.
- Promjena vrijedi za svakog korisnika koji ponovno učitava razgovor; već učitana povijest u otvorenim karticama preglednika ne ažurira se retroaktivno.
- Brisanje cijelog zapisnika (gumb **Clear All**) prazni datoteku, što također djeluje bez ponovnog pokretanja.
- **Odgovor u chatu** sprema se kao obična poruka koja počinje oznakom što imenuje poruku na koju odgovara, na primjer `[RE:2026-09-25 12:00:01|SV1BTL]`, pa se u ovom popisu prikazuje s tom oznakom ispred. Brisanje odgovora uklanja samo taj odgovor. Brisanje poruke na koju odgovara ostavlja njezine odgovore na mjestu, a oni tada prikazuju samo ime i vrijeme na koje odgovaraju.

Gumb za brisanje dosljedno se prikazuje u svih pet Svelte inačica aplikacije. Ako se čini da brisanje nema učinka, provjerite ima li administratorska ploča pravo pisanja u datoteku zapisnika razgovora:

```
ls -l ~/PhantomSDR-Plus/chat.jsonl # path depends on your config
```

Emitirane poruke na slapu

Ploča **Waterfall Message** omogućuje slanje trajnog tekstualnog natpisa na prikaz slapa svakog spojenog korisnika bez diranja procesa poslužitelja.

Slanje poruke

1. Otvorite administratorsku ploču i idite na **Waterfall Message**.

2. Upišite tekst poruke i odaberite boju (heksadecimalno, npr. `#ffdd00`).
3. Kliknite **Send** — poruka se odmah pojavljuje u svim aktivnim prikazima slapa.

Uklanjanje poruke

Kliknite **Clear** da uklonite natpis sa svih slapova. Stanje poruke administratorska ploča drži u memoriji; briše se automatski ako se proces ploče ponovno pokrene.

Kako to radi

Administratorska ploča izlaže dvije interne krajnje točke koje proxy prosljeđuje:

Krajnja točka	Metoda	Svrha
<code>/admin/api/waterfall-message</code>	POST	Postavljanje ili brisanje trenutnog teksta i boje natpisa
<code>/admin/api/waterfall-message</code>	GET	Vraćanje trenutnog stanja poruke kao JSON

Sučelje ispituje stanje poruke i iscrtava je kao preklop na platnu slapa. Na strani klijenta nije potrebno ni ponovno spajanje WebSocketeta ni ponovno učitavanje stranice.

Uobičajene primjene

- Najava planiranog održavanja: "Server restart in 10 minutes"
- Naznaka uvjeta na pojasu: "Solar flux 180 – 10m wide open"
- Poruka dobrodošlice: "Welcome to SV1BTL WebSDR – Athens, KM17"

Grafikoni sustava

Stranica **Grafikoni** crta četiri veličine na zajedničkoj vremenskoj osi:

- **Frekvencija procesora** — prosječni takt svih jezgri, u GHz
- **Opterećenje procesora** — ukupna iskorištenost, u postotcima
- **Temperatura procesora** — u °C, s crtanim oznakama na 70 °C (toplo) i 80 °C (kritično)
- **Korisnici na vezi** — slušatelji spojeni na WebSDR u tom trenutku

Raspon birate s **15 MIN / 1 HOUR / 4 HOURS / 12 HOURS / 24 HOURS**. Prelaskom miša preko grafikona pojavljuje se nitni križ s sve četiri vrijednosti tog trenutka. Četiri kartice iznad grafikona uvijek prikazuju najnoviji uzorak.

Kako se uzorkuje

Pozadinska dretva u `admin_server.py` uzima jedan uzorak svake **2 sekunde**. Uzorci se čuvaju u dvije razine: zadnji **1 sat** u punoj razlučivosti od 2 sekunde (1800 točaka) i **24 sata** tridesetsekundnih prosjeka (2880 točaka) za duge raspone. Stranica bira razinu koja pokriva odabrani raspon i u zaglavlju piše `30s averages` kada gledate prosječenu razinu.

Cijeli dan u razlučivosti od 2 sekunde bio bi ~43000 točaka — deseci MB memorije, prvi prijenos od nekoliko megabajta i oko 40 točaka po pikselu, što nijedan zaslon ne može prikazati. Prosjek na 30 sekundi drži dan u 2880 točaka.

Uzorkovanje kreće zajedno s administracijskom pločom, a ne pri prvom otvaranju stranice, pa se stranica otvara na povijesti koja već postoji.

Međuspremnik je **samo u memoriji**: ništa se ne zapisuje na disk, a povijest se gubi pri ponovnom pokretanju ploče. To je namjerno — funkciju drži izvan rotacije zapisnika i s diska.

2 sekunde daju gotovo živu sliku i dovoljno su kratke da ne izobliče frekvenciju procesora, koja na suvremenom procesoru od uzorka do uzorka skače između takta mirovanja i turba. Jedan uzorak košta otprilike jednu milisekundu, pa je uzorkivač zanemariv uz SDR poslužitelj. Stranica pita u ritmu svoje razine — svake 2 sekunde na živim rasponima, svakih 30 sekundi na prosječenicima, jer češće pitanje vraća samo prazne odgovore.

Za promjenu ritma ili trajanja pohrane uredite ove konstante na početku bloka uzorkivača u `admin_server.py` i ponovno pokrenite:

```
GRAPH_INTERVAL_S    = 2          # sekundi između finih uzoraka
GRAPH_FINE_S        = 3600       # dokle seže fina razina (1 h)
GRAPH_COARSE_EVERY  = 15         # finih uzoraka po skupnoj točki (15 x 2 s = 30 s)
GRAPH_COARSE_S      = 24 * 3600 # dokle seže skupna razina (24 h)
```

Odakle dolaze brojke

Veličina	Izvor
Frekvencija procesora	<code>psutil.cpu_freq()</code> , uz zamjenu <code>/sys/devices/system/cpu/cpu*/cpufreq/</code> . To su isti podaci jezgre koje oblikuje <code>cpufreq-info</code> , pa <code>cpufrequtils</code> nije potreban .
Opterećenje procesora	<code>psutil.cpu_percent()</code>
Temperatura procesora	<code>psutil.sensors_temperatures()</code> , uz zamjenu <code>/sys/class/thermal</code> , zatim naredba <code>sensors</code>
Korisnici na vezi	Vlastita krajnja točka <code>/users</code> poslužitelja <code>spectrumserver</code> — mjerodavan popis sesija. Brojanje priključaka s <code>ss</code> ne vidi stvarne IP adrese klijenata iza <code>proxy.py</code> .

Ako neka veličina na vašem računalu nije dostupna, taj to okvir kaže umjesto da crta ravnu nulu. Neuspjelo očitavanje prekida liniju, da praznina nikada ne izgleda kao izmjerena vrijednost.

Zašto četiri okvira, a ne jedan grafikon

GHz, postotak, stupnjevi i broj ljudi nemaju zajedničku skalu. Crtanje u jednom grafikonu tražilo bi više y-osi, a sjecišta koja pritom nastaju artefakt su skaliranja, a ne činjenica o poslužitelju. Četiri složena okvira na jednoj vremenskoj osi drže usporedbu poštenom. Svaki okvir počinje od nule, pa visina krivulje ostaje razmjerna vrijednosti.

Boja je pridržana za temperaturu, gdje jantarna i crvena označavaju gornje pragove. Ostali okviri dijele jednu boju jer svaki sadrži jednu jedinu seriju koju njegov naslov već imenuje.

Krajnja točka

`GET /admin/api/graph-stats?since=<epoch>` (potrebna prijava) vraća uzorke novije od `since`, pa stranica pita postupno umjesto da svaki ciklus ponovno preuzima cijeli međuspremnik.

Thermal Guard

Cjeloviti priručnik: [THERMAL_GUARD.md](#) — četiri načina rada i što sysop mora učiniti u svakome, rad bez ploče, throttle faza bez roota, testiranje i rješavanje problema.

Zaustavlja SDR poslužitelj kada procesor postane opasno vruć i ponovno ga pokreće kad se ohladi. Dio je ploče — nema ničega dodatnog za instalirati — i pojavljuje se kao kartica **THERMAL GUARD** na Nadzornoj ploči, a postavke su pod **Postavke**.

Isporučuje se **neaktivna**: način rada počinje na `log`, pa u početku samo bilježi što bi učinila. Ništa ne dira vaš poslužitelj dok ne promijenite način rada.

Radi s bilo kojim načinom pokretanja/zaustavljanja

Zaštita nikada ne pokušava utvrditi što nadzire vaš poslužitelj. Djeluje preko skripti koje ste postavili kao **Default start script** i **Default stop script** u Postavkama, a dok je procesor pregrijan ponavlja zaustavljanje pri **svakoj provjeri** (svake 2 s). Sve što vrati poslužitelj — watchdog unutar `start-*.sh`, systemd jedinica, cron zadatak, `tmux` sesija — poništeno je u nekoliko sekundi, sve dok se stroj ne ohladi. To je *zaključavanje* (lockout).

Ako skripta za zaustavljanje nije postavljena, šalje `SIGTERM` procesu navedenom u `sdr_process_name`, a nakon 10 sekundi počeka `SIGKILL`.

Pragovi dolaze iz vašeg vlastitog procesora

Umjesto fiksnog broja koji bi na većini strojeva bio pogrešan, zaštita čita kritični prag koji jezgra objavljuje za vaš procesor i računa unatrag od njega:

Vaš procesor prijavljuje	upozorenje	throttle	zaustavljanje	oporavak
Intel, granica 100 °C	88	92	95	75
AMD Ryzen, granica 95 °C	83	87	90	70
Raspberry Pi, granica 85 °C	73	77	80	60

To je važno: fiksnih 95 °C gotovo je normalno na Intelu, ali na Ryzenu — čiji Tctl po dizajnu stoji na 95 °C pod boostom — zaustavilo bi poslužitelj na posve zdravom stroju, a na Piju, koji smanjuje takt već na 80 °C, ne bi se nikada aktiviralo. Svaki se prag može postaviti i ručno u Postavkama; ostavite polje prazno za *automatski*.

Da vidite kako zaštita vidi vaš stroj:

```
python3 thermal_guard.py --once
```

```
sensor      : hwmon:coretemp
temperature: 58.0 C
crit        : 100.0 C
mode        : log
thresholds  : warn 88.0 throttle 92.0 stop 95.0 resume 75.0 (C)
cpufreq     : not writable – throttle stage disabled
```

Četiri stupnja

Zaustavljanje poslužitelja posljednje je sredstvo, a ne jedini alat:

1. **warn** — samo se zapisuje, ništa više.
2. **throttle** — snižava gornju granicu takta procesora za 20 %. Zahtijeva `cpufreq` upravljački program i pravo pisanja u `scaling_max_freq`, dakle obično root; gdje toga nema (uobičajeno za ploču bez povlastica i na većini VPS-ova) stupanj se **sam isključuje** i zaštita ide s warn izravno na stop. `./setup-cpufreq-perms.sh` daje to pravo pisanja grupi `cpufreq`, pa stupanj radi bez roota — vidi [THERMAL_GUARD.md](#).
3. **stop** — pokreće vašu skriptu za zaustavljanje i drži poslužitelj dolje (vidi zaključavanje).
4. **resume** — nakon što procesor ostane ispod temperature oporavka `thermal_resume_s` (zadano 5 minuta), granica takta se vraća i, u načinu `stop+restart`, ponovno se pokreće vaša skripta za pokretanje.

Svaki stupanj mora se **zadržati** svoje vrijeme — 30 s za warn i throttle, 60 s za stop — pa kratkotrajni skok tijekom prevođenja nikada ne okine zaštitu.

Ponovljena okidanja ograničena su s `thermal_max_stops_hour` (zadano 2). Kad se to dosegne, isključuje se *automatsko ponovno pokretanje* dok zaštita i dalje radi, tako da stroj koji se pregrijava ne titra između rada i zaustavljenosti.

Koji se senzori koriste

Samo pravi senzori na samom procesoru: `coretemp` (Intel), `k10temp` / `zenpower` (AMD), `cpu_thermal` / `soc_thermal` (ARM, Raspberry Pi). Uzima se najviše očitavanje na čipu, a ne prosjek.

`acpitz` i neoznačene toplinske zone **namjerno se zanemaruju** — često prijavljuju temperaturu kućišta ili ploče desetke stupnjeva nižu od stvarnog procesora, pa se zaštita koja bi im vjerovala nikada ne bi aktivirala kad je važno. Ako vaš stroj nema upotrebljiv senzor (često na VPS-u ili unutar kontejnera), kartica to jasno kaže (`no trusted CPU sensor`) i zaštita ostaje neaktivna umjesto da se pretvara da vas štiti.

Što zapisuje

Sve ide u `crash.log`, odnosno u karticu **CRASH** preglednika zapisa, jedan događaj po retku:

```
[THERMAL] warn: 89.0C (warn=88.0 crit=100.0) sustained 31s
[THERMAL] throttle: 93.0C sustained 30s - cpufreq max lowered 20%
[THERMAL] STOPPED server: 96.0C sustained 62s (stop=95.0 crit=100.0) - Started stop-websdr.sh
[THERMAL] lockout: process reappeared at 94.0C - re-stopped [3 time(s) so far]
[THERMAL] recovered: 74.0C held below 75.0 - lockout cleared
[THERMAL] auto-restart: Started start-rx888mk2.sh
```

Redak zaključavanja ograničen je na jedan u minuti, kako nadzornik koji se stalno opire ne bi preplavio zapis.

Testirajte prije nego joj povjerite

Postavite **TEST TEMPERATURE** u Postavkama na vrijednost iznad svojeg praga zaustavljanja. Zaštita to tretira kao stvarno očitavanje, pa se cijeli put — warn, throttle, stop, zaključavanje, oporavak, ponovno pokretanje — odvija na zahtjev. Ispraznite polje za povratak na stvarni senzor.

Napravite li taj test s načinom rada `stop`, poslužitelj se doista zaustavlja i vaši se slušatelji odspajaju. Učinite to kad nikoga nema ili ostavite način na `log`, gdje test pokazuje što bi se *dogodilo*, bez diranja ičega.

Preporučeni način uvođenja

1. Ostavite način na `log` tjedan dana i nastavite normalno.
2. Pročitajte karticu CRASH nakon teških trenutaka — potpunog prevođenja, vrućeg poslijepodneva. Ako se ništa ne pojavi, vaš stroj nikada nije ni prišao.
3. Ako zabilježene temperature izgledaju točno, postavite način na `stop` (ili `stop+restart` da se sam vrati).
4. Potvrdite to jednom poljem testne temperature pa ga pustite na miru.

Izlazak iz zaključavanja

CLEAR LOCKOUT na kartici Nadzorne ploče briše zaključavanje i ograničenje učestalosti. Ne razoružava zaštitu: ako je procesor još prevruć, sljedeća provjera jednostavno će ponovno zaustaviti poslužitelj. Koristite ga nakon što ste popravili hlađenje.

Endpoint

GET /admin/api/thermal (potrebna prijava) vraća cijelo stanje zaštite — senzor, pragove, stupanj, zaključavanje, posljednji događaj. POST s {"action": "reset"} čini isto što i gumb CLEAR LOCKOUT.

Bez administratorske ploče

thermal_guard.py koristi samo standardnu biblioteku i radi samostalno, čitajući isti admin_config.json:

```
python3 thermal_guard.py --once           # ispis senzora, kritičnog praga i pragova
python3 thermal_guard.py                 # pokretanje kao usluga (samo zapisuje dok se ne
naoruža)
python3 thermal_guard.py --mode stop+restart # naoružavanje bez ijedne konfiguracijske datoteke
python3 thermal_guard.py --config /path/to/other.json
python3 thermal_guard.py --help
```

--mode nadjačava thermal_mode iz konfiguracijske datoteke, pa se stroj bez ploče može naoružati izravno iz svoje systemd jedinice, bez ikakvog JSON-a.

Spot Reporting (autorun FT8/FT4/JS8/WSPR)

Kartica **Spot Reporting** upravlja autorun daemonom (autorun/index.js) — Node.js procesom koji dekodira FT8/FT4/JS8/WSPR na poslužiteljskoj strani izravno s prijmnika i šalje spotove u mreže za prijavu:

- **FT8 / FT4 / JS8 → PSK Reporter** (IPFIX/UDP)
- **WSPR → wsprnet.org** (HTTP)

JS8 se spotta samo pri brzini Normal — pozivnom ciklusu od 15 s, na kojem su heartbeatovi i CQ pozivi. Spotovi dolaze iz heartbeatova, compound okvira i usmjerenih poruka; grupna odredišta poput @ALLCALL i pozivni znakovi koje dekodirer nije uspio razriješiti (<...>) nikada se ne prijavljuju. JS8 dijeli PSK Reporter red s FT8 i FT4 te se broji zasebno, jednako kao i oni.

Prijavljivanje je prema zadanom ISKLJUČENO. Ništa se ne šalje dok ne uključite odredište i pritisnete **Start**. Dekodiranje i slanje neovisni su: daemon može dekodirati i zapisivati s isključenim prijavljivanjem, pa možete provjeriti aktivnost prije nego išta postane javno.

△Spotovi se šalju u javne mreže pod **vašim pozivnim znakom**. Uključite samo pojaseve i načine rada koje vaš prijmnik doista čuje i postavite ispravan lokator.

Preduvjeti

Autorun daemon treba Node.js 22+, npm pakete ws + cbor-x (razrješavaju se preko simboličke poveznice autorun/node_modules → frontend/node_modules) i util-linux (taskset). Instalacijske skripte sve to postavljaju automatski — vidi [odjeljak Autorun Spot Reporter u INSTALLATION.md](#). Ako **Start** ne uspije, uobičajeni je uzrok ta poveznica ili taskset (vidi Rješavanje problema niže).

Korištenje kartice

1. **Identitet** — pozivni znak i lokator. Unaprijed popunjeni iz frontend/site_information.json (siteSysop / siteGridSquare, skraćeno na 6 znakova); ovdje ih po potrebi zamijenite.
2. **Matrica pojas x način rada** — označite kombinacije za dekodiranje. Retci su pojasevi, stupci FT8 / FT4 / JS8 / WSPR; nepodržane ćelije su onemogućene. WSPR dodatno pokriva LF/MF pojaseve (2200 m, 630 m) i dodatni europski kanal na 80 m.

3. **Odredišta** — uključite **PSK Reporter** i/ili **wspnret** (oboje prema zadanom isključeno).
4. **Max slots** — sigurnosno ograničenje broja mjesta pojas/način rada koja mogu raditi istodobno.
5. **Start / Stop** — pokreće/gasi daemon (vezan uz gornje CPU jezgre pomoću `taskset`, koje se biraju automatski prema računalu — vidi *Zahtjevi za resursima i ograničenja* niže). **Save / Reload** spremaju i ponovno čitaju konfiguraciju; **Free All** prazni sva mjesta.

Kartica stanja osvježava se svakih 5 s i prikazuje stanje rada, brojače dekodiranja/slanja i vrijeme posljednjeg slanja. Oznaka **REPORTING** pojavljuje se na glavnom slapu dok je prijavljivanje aktivno.

„0 sent“ prvih nekoliko minuta je normalno. PSK Reporter šalje skupno svakih **5 minuta**, a wspnret svake **2 minute** — spotovi čekaju u redu do sljedećeg slanja. Kartica stanja prikazuje broj u čekanju i ritam slanja.

Promjena pojaseva ili načina rada dok radi

Promjene ne vrijede dok se daemon ponovno ne pokrene. `autorun/index.js` čita `autorun.json` samo jednom, pri pokretanju. Nema nadzora datoteke ni signala za ponovno učitavanje — jedini signali koje obrađuje su SIGINT i SIGTERM, a oba znače gašenje. Daemon koji radi zato i dalje dekodira slotove s kojima je pokrenut, što god poslije spremili.

Za primjenu promjene:

1. Označite / odznačite pojaseve i načine rada
2. **SAVE CONFIG**
3. **STOP**
4. **START**

Ili jednostavno **STOP** → promijenite kućice → **START**, jer START sprema konfiguraciju prije pokretanja.

⚠ **Pritisak na START bez prethodnog STOP-a ne primjenjuje promjenu.** START sprema konfiguraciju, a zatim zahtjev za pokretanje odgovara `already running` i ništa se ne pokreće ponovno. Vidjet ćete poruku „spremljeno“ uz pogrešku „already running“ dok daemon nastavlja sa **starim** pojasevima — lako se čita kao uspjeh. Uvijek prvo STOP.

Što ponovno pokretanje čini brojačima:

- **Ukupni brojevi uz svaku kućicu ostaju** — nalaze se u `autorun-totals.json`, koji se pri gašenju ne briše.
- **Pločice po dekoderu vraćaju se na nulu**, vrijede po pokretanju.
- **Ništa iz reda se ne gubi**: gašenje prvo pošalje spotove na čekanju.
- **Novouključeni** pojas kreće od 0; pojas koji ste **već koristili** nastavlja svoj prijašnji zbroj.
- Pojas koji **odznačite zadržava svoj broj** uz sada praznu kućicu; povijest se ne briše.

Dekodiranje staje samo na nekoliko sekundi između STOP-a i START-a.

Brojači spotova

Prikazana su dva različita brojanja koja odgovaraju na različita pitanja.

SPOTS UPLOADED PER DECODER — pločica po dekoderu (FT8 / FT4 / JS8 / WSPR) sa spotovima poslanima **od zadnjeg pokretanja daemona**, a ispod broj dekodiranja i koliko ih još čeka u redu. Stop/Start ih vraća na nulu. Dekoder čije je odredište isključeno prikazuje svoja dekodiranja i `reporting off` umjesto golog 0, jer je ondje nula slanja postavka, a ne kvar.

Broj uz svaku kućicu u BANDS & MODES je broj **ukupno** poslanih spotova za tu kombinaciju pojasa i načina rada. Čuva se u `autorun-totals.json` i preživljava ponovna pokretanja. **Jantarno s točkom**

ispred (·123) znači da je taj slot dekodirao, ali još ništa nije poslao. To je normalno između slanja — PSK Reporter šalje svakih 5 minuta, a wsprnet svake 2 — pa FT8/FT4 brojači prvih nekoliko minuta nakon pokretanja stoje jantarno. Jantarno ostaje i ako je odredište tog dekodera isključeno. Obična 0 bez točke znači da je slot uključen, ali još ništa nije dekodirao — WSPR stoji na nuli nekoliko minuta nakon pokretanja jer radi u ciklusu od 2 minute, dok FT8 već broji u stotinama. Slot koji daemon nikada nije pokrenuo ne prikazuje ništa. Prelaskom miša vide se slanja, dekodiranja i vrijeme zadnjeg slanja.

FT8 i FT4 dijele jedan red za slanje prema PSK Reporteru, pa daemon svaki spot pripisuje vlastitom načinu rada i pojasu u trenutku slanja; podjela se ne procjenjuje naknadno iz ukupnih brojeva.

Poništavanje brojača. FREE ALL SLOTS odznacuje sve pojaseve i načine rada, isključuje oba odredišta, zaustavlja daemon i **brise ukupne brojače** — nakon toga uz nijednu kućicu nema broja. To je jedini način da ih poništite i ne može se poništiti. Gumb zaustavlja daemon i čeka da izađe prije brisanja `autorun-totals.json`, jer daemon tu datoteku ponovno zapisuje pri gašenju; brisanje dok daemon radi samo bi vratilo brojeve.

Zahtjevi za resursima i ograničenja

Daemon je namjerno lagan i radi na hardveru skromnih resursa (četverojezgreni i5 je dovoljan), ali postoje stvarna ograničenja kojih treba biti svjestan.

Vezanje autoruna uz jezgre svjesno je konfiguracije i automatsko. Autorun daemon uvijek pokreće administratorska ploča (`admin_server.py`), pa se njegovo vezanje postavlja pri svakoj instalaciji bez ikakve konfiguracije: iz broja procesora izvodi raspon jezgri za `taskset`, rezervirajući nekoliko gornjih jezgri za dekodiranje, ili radi bez vezanja na ≤ 4 jezgre.

Vezanje spectrumservera ovisi o vašem načinu pokretanja. Način na koji pokrećete spectrumserver razlikuje se među instalacijama (vrsta SDR-a, alati, osobne skripte), pa ovaj dokument ne pretpostavlja nikakav određeni pokretač. Administratorska ploča ne pokreće niti veže spectrumserver — to u potpunosti ovisi o naredbi ili usluzi kojom ga izvodite. Ako želite spectrumserver držati podalje od jezgri koje koristi autorun daemon, vežite ga sami pomoću `taskset` u vlastitoj naredbi za pokretanje (vidi odjeljak o zamjeni niže). Ako to ne učinite, jednostavno radi bez vezanja, a raspoređivač operacijskog sustava ga uravnotežuje — ispravno i sigurno, samo gubite namjerno razdvajanje jezgri.

Za orijentaciju, autorun daemon rezervira ove gornje jezgre (pa ako vežete spectrumserver, zadržite ga na nižima kako biste izbjegli preklapanje):

Logičkih CPU-a	Koristi autorun daemon	Ostaviti za spectrumserver
≤ 4	<i>bez vezanja</i>	<i>bez vezanja</i>
6	5	0-4
8	6-7	0-5
12 (npr. hibrid 8P+4E)	8-11	0-7
16	12-15	0-11

Na računalu s **4 jezgre (ili manje)** nema se što razdvajati, pa i autorun daemon radi *bez vezanja* i dijeli sve jezgre — ispravno i sigurno, ali FFT SDR-a i naleti dekodiranja natječu se za iste jezgre.

Poznata ograničenja:

- Pravo usko grlo su spectrumserver i širina pojasa SDR-a, a ne daemon.** RX888 na 30 MHz traži OpenCL/GPU; procesor s malo jezgri bez sposobnog GPU-a ne može održati taj FFT. Uparite skromni hardver s užim SDR-om (RSP1A $\approx$ 10 MHz, RTL-SDR $\approx$ 2.4 MHz).
- WSPR je najzahtjevniji za CPU.** Njegov Fano dekodirer je JS port ($\sim$ 30 s po pojasu, jednodretveni). Skup od 4 workera izvodi 4 dekodiranja paralelno; uključivanje **više od $\sim$ 4 WSPR pojasa** odjednom

može produljiti red preko intervala od 120 s, osobito dok se SDR natječe za jezgre. Dekodiranja FT8/FT4 u usporedbi su jeftina.

3. **Skaliranje broja mjesta.** Podatak od ~2–2,5 jezgre / 600–700 MB odnosi se na punih ~28 mjesta na osmojezgrenom računalu. Na 4 dijeljene jezgre ograničite se na nekoliko pojaseva (smjernica: ≤ 6 FT8/FT4 + ≤ 3 WSPR) i pratite statistiku `queued` skupa.
4. **Vezanje pretpostavlja hibridnu Intel topologiju** (niže jezgre = brže P-jezgre). Na AMD-u ili procesorima s isprepletenim SMT numeriranjem automatska podjela i dalje vrijedi (bez preklapanja, bez rušenja), ali „niže jezgre su brže” možda ne vrijedi doslovno; na čipu 4C/8T s hyperthreadingom gornje su jezgre SMT blizanci — ograničene, ali ne i potpuno izolirane. Ondje gdje automatski raspon nije idealan, upotrijebite **ručnu zamjenu** niže.
5. **Pokrivenost pojaseva ograničava prijammnik.** Konfiguracija RX888 (`sps=60000000` → Nyquist na 30 MHz) ne doseže 6 m i više; tablica pojaseva ide od 160 m do 10 m. Drugi SDR-ovi pokrivaju ono što dopušta njihov ugođeni prozor.
6. **Ritam prijavljivanja.** PSK Reporter šalje svakih 5 min, wsprnet svake 2 min — „0 sent” prvih nekoliko minuta normalno je, a ne kvar.

Ručna zamjena vezanja uz jezgre (napredno)

Automatski izvedena podjela jezgri prikladna je za većinu računala, no možete nametnuti određeno vezanje ondje gdje nije (AMD CCX/CCD, ARM big.LITTLE, isprepleteni SMT). Dvije su neovisne zamjene, a svaka prihvaća popis jezgri u obliku `taskset -c (2-3, 0,2,4, 0-2,5)` ili `none/off/unpinned` za isključivanje vezanja. Neispravna vrijednost se zanemaruje i primjenjuje se automatsko izvođenje — tipfeler nikada ne može spriječiti pokretanje.

Autorun daemon — varijabla okruženja `AUTORUN_CORES` ili polje `"cores"` u `autorun.json` (okruženje ima prednost). Polje `"cores"` čuva se pri **Save** u administratorskoj ploči, pa ručna izmjena ostaje. Dva načina da ga postavite:

Mogućnost A — `autorun.json` (trajno, preporučeno). Dodajte redak `"cores"` u konfiguraciju u korijenu repozitorija, zatim na kartici Spot Reporting učinite **Stop → Start**:

```
// autorun.json
{ "identity": { "callsign": "SV1BTL", "grid": "KM17VX" },
  "reporting": { "pskreporter": true, "wsprnet": false },
  "slots": [ /* ... */ ],
  "cores": "2-3" }           // or "none" to run unpinned
```

Mogućnost B — varijabla okruženja (jednokratno). Daemon pokreće administratorska ploča, pa varijabla mora biti u okruženju **ploče** — postavite je i ponovno pokrenite ploču (okruženje ima prednost pred vrijednošću u `autorun.json`):

```
# Način A – varijabla mora biti na procesu koji se pokreće:
./manage_admin.sh stop && AUTORUN_CORES=2-3 ./manage_admin.sh start

# Način B – systemd ignorira varijablu zadanu u naredbenom retku
# systemctl, pa je stavite u jedinicu:
sudo systemctl set-environment AUTORUN_CORES=2-3
sudo systemctl restart phantomsdr-admin
```

spectrumserver — vežite ga sami tako da naredbi kojom pokrećete poslužitelj dodate prefiks `taskset -c <jezgre>`. To radi sa svakim načinom pokretanja (izravno pokretanje, SDR cjevovod, jedinica usluge itd.):

```
# pin the server to cores 0-3, direct launch:
taskset -c 0-3 ./build/spectrumserver --config <your-config>.toml &lt; <your-input>
# or in an SDR pipeline:
<your-sdr-source> | taskset -c 0-3 ./build/spectrumserver --config <your-config>.toml
```

⚠ **Pobrinite se da preživi ponovna pokretanja.** Ugrađeni `taskset` vrijedi samo za to jedno pokretanje. Ako nešto automatski ponovno pokreće poslužitelj (`watchdog`, `systemd` usluga, `cron`/`@reboot` unos ...), dodajte prefiks `taskset` unutar skripte ili jedinice koja ga zapravo pokreće — inače ponovno pokretanje gubi vezanje.

Provjerite je li vezanje primijenjeno — ispišite stvarni afinitet CPU-a pokrenutih procesa:

```
taskset -cp "$(pgrep -f 'autorun/index.js')" # autorun daemon
taskset -cp "$(pgrep -x spectrumserver)" # spectrumserver
```

Prihvaćene vrijednosti (za obje zamjene): popis u obliku `taskset -c (2-3, 0,2,4, 0-2,5)` ili `none/off/unpinned` za rad bez vezanja. Sve neispravno zanemaruje se i primjenjuje se automatsko izvođenje, pa tipfeler nikada ne može blokirati pokretanje.

Na Raspberry Piju / bilo kojem računalu s ≤ 4 jezgre obično ne trebate nijednu — automatski put već pokreće oba procesa bez vezanja, što je ispravan izbor na malom, homogenom procesoru. Zamjena je namijenjena većim računalima koja nisu Intel hibridi.

Datoteke i krajnje točke

Stavka	Svrha
<code>autorun.json</code>	Spremljena konfiguracija (identitet, mjesta, odredišta, najveći broj mjesta, neobavezna zamjena <code>cores</code>). Zapisuje je kartica; git je zanemaruje.
<code>autorun-status.json</code>	Trenutačno stanje koje čita kartica stanja (pid, brojači, posljednje slanje). Zapisuje se svakih 15 s; uklanja se pri zaustavljanju.
<code>frontend/dist/autorun-active.json</code>	Javni podaci oznake, posluženi na <code>/autorun-active.json</code> . Prazno kada je prijavljivanje isključeno.
<code>autorun.log</code>	Standardni izlaz i pogreške daemona.
<code>GET/POST /admin/api/autorun/config</code>	Čitanje/pisanje <code>autorun.json</code> (provjerava pozivni znak, lokator, kombinacije pojas/način rada, ograničenje mjesta).
<code>GET /admin/api/autorun/status</code>	Stanje rada (<code>pgrep</code>) + <code>autorun-status.json</code> .
<code>POST /admin/api/autorun/start / stop</code>	Pokretanje (<code>taskset -c <auto> node autorun/index.js</code>) / <code>SIGTERM</code> . Raspon jezgri izvodi se iz procesora (vidi gore).

Napomena: nakon nadogradnje `admin_server.py` morate **Ponovno pokretanje ploče** kako bi se učitale nove autorun rute ili ažurirana matrica pojas/način rada. Daemon radi kao zaseban proces, pa preživljava ponovna pokretanja administratorske ploče.

Rješavanje problema

Problem	Rješenje
Administratorska ploča se ne pokreće	Provjerite <code>admin.log</code> — obično nedostaje Python paket
Proxy se ne pokreće	Izvršite <code>python3 -c "import aiohttp"</code> — ako ne uspije: <code>pip3 install aiohttp --break-system-packages</code>
<code>proxy.log</code> je prazan	Prvo provjerite radi li proxy uopće (<code>./manage_admin.sh status</code>). Ako radi, imate staru verziju: pokretač mora koristiti <code>python3 -u</code> (bez međuspremnik — inače natpis pri pokretanju nikad ne napusti međuspremnik od 8 KB), a <code>main()</code> u <code>proxy.py</code> mora pozvati <code>logging.basicConfig()</code> (inače aiohttpov zapisivač pristupa nema rukovatelja i svaki se redak zahtjeva odbacuje, jer <code>web.AppRunner</code> — za razliku od <code>web.run_app</code> — ne konfigurira zapisivanje).
Proxy se pokreće, ali je slap prazan	Provjerite <code>sdr_host</code> u <code>admin_config.json</code> — uobičajeno <code>127.0.0.1</code> . Ako još sadrži stari LAN IP iz prethodne verzije, promijenite ga i Ponovno pokretanje ploče
Gumb Kick javlja „no connections“	Izvršite <code>getcap \$(which ss)</code> — ako nema <code>cap_net_admin</code> , izvršite <code>sudo setcap cap_net_admin+ep \$(which ss)</code>
Stanje uvijek prikazuje OFFLINE	Idite na Settings → SDR Process Name i unesite točan naziv koji prikazuje <code>ps -eo comm,args grep -v grep</code>
Stranica Users ne prikazuje podatke	Provjerite radi li krajnja točka: <code>curl http://127.0.0.1:<public_port>/users</code>
Vanjski preglednik javlja NetworkError	Provjerite radi li proxy: <code>./manage_admin.sh status</code>
Proxy se pokvario nakon promjene mreže	Odnosi se samo na slučaj kad <code>sdr_host</code> još sadrži LAN IP — postavite ga na <code>127.0.0.1</code> i Ponovno pokretanje ploče
Gumb za brisanje razgovora nema učinka	Provjerite pravo pisanja u datoteku zapisnika razgovora: <code>ls -l ~/PhantomSDR-Plus/chat.jsonl</code>
Poruka na slapu se ne pojavljuje	Potvrdite da proxy radi (<code>./manage_admin.sh status</code>) i da je sučelje iz novije verzije koja uključuje iscrtavanje preklopa
Poruka na slapu izgubljena nakon ponovnog pokretanja	Očekivano — stanje poruke je samo u memoriji; pošaljite je ponovno nakon ponovnog pokretanja ploče
„Start“ u Spot Reportingu ne uspijeva / daemon se odmah gasi	Provjerite <code>autorun.log</code> . Obično nedostaje simbolička poveznica <code>autorun/node_modules (ln -sf ../frontend/node_modules autorun/node_modules)</code> ili nije instaliran <code>taskset (sudo apt install -y util-linux)</code> .
Spot Reporting: daemon radi, ali decodes ostaje 0, a <code>autorun.log</code> pokazuje 504 / tap closed 1006	Audio odvod ne može doseći <code>spectrumserver</code> . Daemon automatski otkriva port iz konfiguracije pokrenutog poslužitelja; redak zapisnika <code>[autorun] tap backend: HOST:PORT</code> mora odgovarati vašem <code>[server] port</code> . Ako nije točan (ili poslužitelj nije radio pri pokretanju), zadajte ga u <code>autorun.json</code> : <code>"server": { "host": "127.0.0.1", "port": 9002 }</code> , zatim Stop→Start.
Spot Reporting prikazuje „0 sent“	Normalno prvih nekoliko minuta — PSK Reporter šalje svakih 5 min, wsrnet svake 2 min. Provjerite je li uključeno odredište i radi li daemon.
Novi pojasevi/načini rada ne prikazuju se u matrici	Ponovno pokretanje ploče — matrica se učitava pri pokretanju ploče.
<code>autorun.log</code> prikazuje <code>MODULE_TYPELESS_PACKAGE_JSON /</code>	Kozmetičko upozorenje (dekodiranje i dalje radi). U <code>frontend/src/modules/package.json</code> nedostaje <code>"type": "module"</code> ; dodajte taj

Problem	Rješenje
„Reparsing as ES module ... performance overhead“	redak pri vrhu, zatim Stop→Start. Ponovna izgradnja sučelja nije potrebna — daemon izravno uvozi izvornu datoteku.
„Start“ u Spot Reportingu ne uspijeva uz <code>taskset: ... Invalid argument</code> na starom/malom računalu	Ne bi se trebalo događati s pokretačem svjesnim konfiguracije — raspon jezgri izvodi se iz broja procesora i na ≤ 4 jezgre radi bez vezanja. Ako se dogodi, vaš <code>admin_server.py</code> stariji je od te promjene; ažurirajte ga (<code>_autorun_taskset_prefix</code>) i Ponovno pokretanje ploče .
Oznaka REPORTING ne pojavljuje se na slapu	Prijavljivanje mora biti uključeno s barem jednim mjestom; oznaka čita <code>/autorun-active.json</code> . Ponovno izgradite sučelje ako je <code>dist/index.html</code> stariji od oznake.

POGLAVLJE 4

Thermal Guard

THERMAL_GUARD.md

Thermal Guard — Priručnik za sysopa

Zaštita od pregrijavanja procesora za PhantomSDR-Plus.

WebSDR je jedna od rijetkih stvari koje drže svaku jezgru zaposlenom, cijeli dan, zauvijek. Ako se ventilator začepi, pumpa stane ili se prostorija u kolovozu pregrije, nitko vam to neće reći — poslužitelj jednostavno nastavlja dekodirati dok se procesor ne skuha. Thermal guard je dio koji to primijeti i izvuče utikač umjesto vas.

Isporučuje se **neaktivan**: iz kutije samo zapisuje u dnevnik. Ništa nije zaštićeno dok ne odaberete način rada. Ovaj priručnik govori o tome kako donijeti tu odluku i živjeti s njom.

Žurite se? Pokrenite `python3 thermal_guard.py --once`, pročitajte pragove koje ispiše i postavite `thermal_mode` na `stop+restart` u administratorskoj ploči. To je cijeli posao. Sve ostalo niže je *zašto*.

Sadržaj

1. Što radi
2. Brzi početak
3. Kako odlučuje
4. Četiri načina rada — što svaki radi i što vi morate učiniti
5. Odabir načina rada
6. Referenca postavki
7. Rad bez administratorske ploče
8. Uključivanje throttle faze bez roota
9. Rad bez nadzora — kad nitko ne gleda
10. Isprobajte ga prije nego mu povjerite stroj
11. Čitanje dnevnika
12. Lockout i ograničenje broja zaustavljanja
13. Rješavanje problema
14. Isključivanje ili uklanjanje

1. Što radi

Svake 2 sekunde čuvar čita temperaturu jezgre procesora i uspoređuje je s četiri praga. Kad temperatura dovoljno dugo ostane iznad nekog od njih, penje se ljestvama:

```

normalno → warn → throttle → stop → (hlađenje) → restart
          redak u   snižavanje  pokretanje  poslužitelj  pokretanje
          dnevniku takta        stop skripte  ostaje ugašen start skripte
  
```

Dvije projektne odluke čine ga upotrebljivim na stroju *bilo kojeg* sysopa, i vrijedi ih razumjeti prije nego se na njega oslonite:

Nikad ne pita kako pokrećete poslužitelj. systemd, watchdog petlja, cron, tmux, obična ljuska — čuvar to ne zna i nije ga briga. Umjesto toga, dok je stroj pregrijan, drži **lockout**: ponavlja naredbu zaustavljanja u *svakom* ciklusu. Ako vaš watchdog ponovno pokrene poslužitelj, čuvar ga zaustavi unutar dvije sekunde. Što god ga pokušava održati na životu, gubi — dok se procesor ne ohladi. Upravo to ga čini univerzalnim, i testirano je protiv stvarnog watchdoga.

Nikad ne ugrađuje fiksnu temperaturu. „Zaustavi na 95 °C“ loše je pravilo: na Intelu s Tjmax 100 to je gotovo normalno opterećenje, na Ryzenu je *ispod* mjesta na kojem Tctl po dizajnu stoji pod boostom, a na Raspberry Piju je nedostižno jer se SoC sam ograničava na 80. Zato čuvar čita kritični prag koji objavljuje vaša vlastita jezgra sustava (`tempN_crit`) i računa unatrag. Isti kod, razumne brojke posvuda.

2. Brzi početak

1. Pogledajte kako vaš stroj izgleda čuvaru. Uvijek prvo ovo:

```
cd ~/PhantomSDR-Plus
python3 thermal_guard.py --once
```

```
sensor      : hwmon:coretemp
temperature: 65.0 C
crit        : 100.0 C
mode        : log
thresholds  : warn 88.0  throttle 92.0  stop 95.0  resume 75.0 (C)
cpufreq     : not writable – throttle stage disabled
```

Ako `sensor` kaže `NONE`, stanite ovdje i pročitajte [Rješavanje problema](#) — čuvar ne može zaštititi stroj koji ne može izmjeriti, i to će vam reći umjesto da se pretvara.

2. Ostavite ga nekoliko dana u načinu `log`. Već radi unutar administratorske ploče. Neka promatra jedan normalan tjedan — ljetna popodnevna, veliko prevođenje, prometnu subotu s natjecanjem — i vidite bi li se ikad oglasio.

3. Naoružajte ga. Administratorska ploča → **Settings** → **THERMAL GUARD** → **MODE** → stop+restart → **SAVE THERMAL SETTINGS**. Vrijedi za otprilike 2 sekunde; ponovno pokretanje nije potrebno.

3. Kako odlučuje

Senzor

Čuvar vjeruje samo stvarnim senzorima na jezgri procesora:

Izvor	Tipičan hardver
<code>coretemp</code>	Intel
<code>k10temp</code> , <code>zenpower</code> , <code>zenpower3</code>	AMD
<code>cpu_thermal</code> , <code>soc_thermal</code>	Raspberry Pi, ARM pločice
Toplinska zona <code>x86_pkg_temp</code>	rezerva na x86

`acpitz` i neoznačene toplinske zone **namjerno** su isključeni. Na tipičnom stolnom računalu `acpitz` pokazuje oko 28 °C dok je paket procesora na 59 °C — čuvar koji bi mu vjerovao jednostavno se nikad ne bi oglasio. Prijaviti „isključeno, nema pouzdanog senzora“ poštenije je nego glumiti zaštitu.

`psutil` se koristi ako je slučajno instaliran, ali nikad nije nužan. Čuvar koristi isključivo standardnu biblioteku.

Pragovi

Svaki je odmak ispod kritičnog praga koji stroj sam objavljuje:

Prag	Odmak ispod crit	Značenje
warn	−12 °C	Nešto nije u redu, zabilježi
throttle	−8 °C	Pokušaj hlađenja usporavanjem procesora
stop	−5 °C	Preblizu rubu — zaustavi poslužitelj
resume	−25 °C	Doista opet hladno, povratak je siguran

U praksi to izgleda ovako:

Vaš procesor prijavljuje	warn	throttle	stop	resume
Intel, crit 100 °C	88	92	95	75
AMD Ryzen, crit 95 °C	83	87	90	70
Raspberry Pi, crit 85 °C	73	77	80	60

Ako stroj uopće ne objavljuje kritičnu točku, čuvar se vraća na konzervativan fiksni skup i to izrijekom kaže. Svaki prag možete nadjačati apsolutnom vrijednošću (vidi [Referencu postavki](#)), ali ne biste trebali morati.

Vremena zadržavanja — zašto vršak nikad ne djeluje

Ništa se ne događa na temelju jednog vrućeg očitavanja. Prag mora biti zadržan **neprekidno**:

- **warn** i **throttle**: 30 s (`thermal_warn_sustain_s`)
- **stop**: 60 s (`thermal_sustain_s`)
- **resume**: 300 s ispod praga povratka prije ponovnog pokretanja (`thermal_resume_s`)

Vršak pri prevođenju, nalet dekodiranja, ventilator koji ubrzava — sve je predugo kratko da bi se računalo. Jedno očitavanje ispod praga vraća sat na nulu. Uz to, nakon pokretanja čuvar traži 3 uzastopna valjana očitavanja prije nego uopće nešto poduzme, i odbacuje svaku vrijednost izvan 20–125 °C kao grešku senzora, a ne kao temperaturu.

4. Četiri načina rada

`thermal_mode` jedini je prekidač koji određuje koliko se visoko čuvar smije popeti ljestvama. Svaki način **uključuje sve što rade prethodni**.

Način	Upozorava	Usporava	Zaustavlja	Ponovno pokreće	Može li poslužitelj pasti?
log	✓	—	—	—	Ne — ništa ne dira
throttle	✓	✓	—	—	Ne
stop	✓	✓	✓	—	Da, ostaje ugašen dok ne reagirate
stop+restart	✓	✓	✓	✓	Da, vraća se sam

log — promatraj i izvijesti (tvornička postavka)

Što radi: sve što rade i ostali načini, samo bez djelovanja. Prolazi cijelim ljestvama i u `crash.log` zapisuje što bi *bio* učinio. Nikad ne pokreće vašu stop skriptu, ne dira takt procesora, ne prekida nijednog slušatelja.

Što morate učiniti: ništa — već radi. Nakon tjedan dana otvorite karticu **CRASH** u ploči ili pokrenite:

```
grep THERMAL crash.log
```

- **Nema ničega?** Vaše hlađenje je u redu i čuvara možete mirno naoružati.
- **STOP level reached ... NOT stopping?** Čuvar bi vam bio ugasio poslužitelj. To je stvaran toplinski problem s kojim ste radili — popravite hlađenje i naoružajte čuvara.

Koristite ga kad: ste tek instalirali čuvara i još ne znate kako se vaš stroj ponaša. To je način za uhodavanje, a ne odredište — čuvar u načinu `log` ne štiti ništa.

throttle — uspori procesor, nikad ne prekida uslugu

Što radi: na throttle pragu snižava najveću frekvenciju procesora za 20 %, a vraća je čim se temperatura vrati ispod linije upozorenja. Vaš WebSDR cijelo vrijeme ostaje na mreži. Slušatelji pod velikim opterećenjem možda primijete nešto slabije performanse; većina neće primijetiti ništa.

Ako temperatura ipak nastavi rasti preko praga zaustavljanja, ovaj način to **samo zabilježi i ne učini ništa** — neće zaustaviti poslužitelj.

Što morate učiniti:

1. Dati čuvaru pravo pisanja na ograničenje frekvencije procesora — treba mu, a po zadatome ga ima samo root. Jednom pokrenite `./setup-cpufreq-perms.sh`; vidi [odjeljak 8](#).
2. Provjeriti s `python3 thermal_guard.py --once` da ispisuje `cpufreq : writable, throttle stage available`. Ako piše *not writable*, ovaj način ne radi ništa više od `log`, i to će svaki put reći u dnevniku.

Koristite ga kad: tražite granični problem hlađenja i želite braniti stroj bez ijednog izgubljenog slušatelja. **Imajte na umu da to nije potpuna zaštita** — ako usporavanje nije dovoljno, ništa se drugo ne događa. Dobar kao prvi korak, loš kao konačan odgovor.

stop — ugasi poslužitelj i ostavi ga ugašenim

Što radi: na pragu zaustavljanja pokreće vašu stop skriptu, a zatim drži **lockout** — ponovno zaustavlja poslužitelj svake 2 sekunde dok je stroj pregrijan, tako da ga ništa ne oživi iza vaših leđa. Kad procesor 5 minuta bude ispod praga povratka, lockout se otpušta. **Poslužitelj se ne vraća sam**. Vi ga pokrećete kad se uvjerite da je uzrok otklonjen.

Što morate učiniti:

1. Osigurati da čuvar zna kako zaustaviti vaš poslužitelj. U ploči je to podešena stop skripta (`./stop-websdr.sh` uz isporučene pokretače). Ako nije postavljena, čuvar se vraća na SIGTERM prema imenu procesa, pa SIGKILL nakon 10 sekundi počeka — radi, ali prava stop skripta je čišća.
2. Prihvatiti da će **vaš WebSDR biti izvan mreže dok to ne primijetite**. Osigurajte da doznate: nadzor dostupnosti, kartica CRASH ili jednostavno svakodnevna provjera.

Koristite ga kad: stroj vam je važniji od usluge, obično ste u blizini, ili je toplinski događaj dovoljno ozbiljan da želite pregledati stroj prije nego nastavi. Također pravi izbor ako ne vjerujete automatskom ponovnom pokretanju na svom hardveru.

stop+restart — ugasi, ohladi se, vrati se sam

Što radi: sve što i `stop`, uz dodatak: kad procesor 5 minuta ostane ispod praga povratka, pokreće vašu start skriptu i WebSDR se vraća sam.

Lepršanje sprječava `thermal_max_stops_hour` (zadano 2). Nakon dva toplinska zaustavljanja unutar sat vremena čuvar **isključuje automatsko ponovno pokretanje** i ostavlja poslužitelj ugašenim da ga pogledate. Uočite važan detalj: ograničenje gasi *ponovno pokretanje*, nikad *zaštitu* — *lockout* i dalje zaustavlja poslužitelj dok je vruć, koliko god se puta već oglašio.

Što morate učiniti:

1. Osigurati da su **obje** skripte podešene i da doista rade samostalno — i start i stop. Isprobajte ih ručno: `./stop-websdr.sh`, pa `./start-rx888mk2.sh`.
2. Ništa više. Ovo je način „postavi i zaboravi“.

Koristite ga kad: je WebSDR javan, bez nadzora ili udaljen — dakle u većini slučajeva. To je preporučeni način za normalnu instalaciju.

5. Odabir načina rada

Vaša situacija	Način
Tek ste instalirali čuvara, stroj još ne poznajete	log tjedan dana
Javni WebSDR, bez nadzora, treba se brinuti sam o sebi	stop+restart
Udaljena lokacija do koje teško dolazite	stop+restart
Želite pregledati stroj nakon svakog toplinskog događaja	stop
Hardver kojem ne vjerujete automatsko ponovno pokretanje	stop
Granično hlađenje, ne smijete gubiti slušatelje	throttle, pa dalje na stop+restart
Stroj bez pouzdanog senzora procesora	nijedan neće raditi — prvo riješite senzor

Za većinu sysopa odgovor je `stop+restart`, a pošten sažetak ostalih glasi: `log` ne štiti ništa, a `throttle` štiti malo.

6. Referenca postavki

Svi ključevi nalaze se u `admin_config.json` pokraj `thermal_guard.py`, a svaki od njih može se urediti i u **Settings → THERMAL GUARD**. Čuvar **ponovno čita datoteku u svakom ciklusu**, pa promjene vrijede unutar ~2 sekunde — bez ponovnog pokretanja.

Ključ	Zadano	Značenje
<code>thermal_enabled</code>	<code>true</code>	Glavni prekidač. <code>false</code> potpuno isključuje čuvara.
<code>thermal_mode</code>	<code>"log"</code>	<code>log</code> <code>throttle</code> <code>stop</code> <code>stop+restart</code> — vidi odjeljak 4 .
<code>thermal_warn</code>	<code>null</code>	Apsolutni prag upozorenja u °C. <code>null</code> = izvedi iz kritičnog praga.
<code>thermal_throttle</code>	<code>null</code>	Apsolutni throttle prag. <code>null</code> = izveden.
<code>thermal_stop</code>	<code>null</code>	Apsolutni prag zaustavljanja. <code>null</code> = izveden.
<code>thermal_resume</code>	<code>null</code>	Apsolutni prag povratka. <code>null</code> = izveden.
<code>thermal_sustain_s</code>	<code>60</code>	Sekunde zadržavanja praga zaustavljanja prije gašenja. Najmanje 5.
<code>thermal_warn_sustain_s</code>	<code>30</code>	Sekunde za faze upozorenja i throttlea. Najmanje 5.
<code>thermal_resume_s</code>	<code>300</code>	Sekunde ispod praga povratka prije otpuštanja lockouta. Najmanje 30.
<code>thermal_max_stops_hour</code>	<code>2</code>	Toplinska zaustavljanja na sat nakon kojih se gasi automatsko pokretanje. Zaštita se nastavlja.
<code>thermal_test_temp</code>	<code>null</code>	Pretvara se da je procesor na ovoj temperaturi. Samo za testiranje — vidi odjeljak 10 .

Samostalni čuvar dodatno koristi `start_script`, `stop_script`, `sdr_process_name` i `sdr_base_dir` iz iste datoteke.

Nadjačavanje praga. Učinite to samo ako imate razlog. Čest je ovaj: vaš se stroj opravdano grije pri prevođenju i ne želite da se to broji. Podizanje `thermal_stop` kupuje vam prostor po cijenu odmaka od kritične točke — nikad ga ne postavljajte iznad svoje crit vrijednosti, inače će procesor prvi doći do vlastitog nužnog gašenja, a čuvar postaje ukras.

7. Rad bez administratorske ploče

Mnogi sysopi nikad ne instaliraju ploču. Čuvar sasvim dobro radi i sam — ista datoteka, isto ponašanje.

Prvo ga isprobajte u prvom planu:

```
cd ~/PhantomSDR-Plus
python3 thermal_guard.py --once          # što bi se ovdje dogodilo?
python3 thermal_guard.py --mode log      # gledajte uživo, Ctrl-C za izlaz
```

`--mode` nadjačava `thermal_mode` iz naredbenog retka, pa nikad ne morate ručno pisati JSON:

```
python3 thermal_guard.py --mode stop+restart
python3 thermal_guard.py --config /etc/phantomsdr/thermal.json
```

Zatim ga učinite trajnim pomoću ogledne jedinice isporučene u projektu:

```
sudo cp thermal-guard.service /etc/systemd/system/
sudo nano /etc/systemd/system/thermal-guard.service # postavite User= i putanje
sudo systemctl daemon-reload
sudo systemctl enable --now thermal-guard
systemctl status thermal-guard
```

Pročitajte komentare na vrhu `thermal-guard.service` prije nego je omogućite. Dvije stvari su najvažnije:

- **User= mora biti račun koji doista može pokrenuti vaše start i stop skripte.** Pokretanje čuvara kao root obično je pogrešan odgovor: tada bi i te skripte pokretao kao root.
- Ili ga uputite na `admin_config.json` koji sadrži `start_script` i `stop_script`, ili proslijedite `--mode` u retku `ExecStart`. Dovoljna je minimalna konfiguracija:

```
{
  "sdr_base_dir": "/home/vaskorisnik/PhantomSDR-Plus",
  "sdr_process_name": "spectrumserver",
  "start_script": "start-rx888mk2.sh",
  "stop_script": "stop-websdr.sh",
  "thermal_mode": "stop+restart"
}
```

Sve što čuvar radi i dalje ide u `crash.log` pokraj te konfiguracijske datoteke, uz `systemd journal` (`journalctl -u thermal-guard`).

8. Uključivanje throttle faze bez roota

Po zadanome `python3 thermal_guard.py --once` javlja:

```
cpufreq : not writable – throttle stage disabled
```

To je normalno i nije greška. Jezgra sustava stvara

```
/sys/devices/system/cpu/cpuN/cpufreq/scaling_max_freq
```

u vlasništvu `root:root`, s pravima `0644`. Administratorska ploča koja radi kao običan korisnik tamo ne može pisati, pa čuvar preskače throttle fazu i to kaže u dnevniku umjesto da tiho zakaže.

Pogrešno rješenje je pokretati čuvara kao root. On pokreće i vaše start i stop skripte, a one trebaju ostati bez povlastica.

Ispravno rješenje je dati pravo pisanja upravo tim datotekama posebnoj grupi. Jedna skripta to obavlja:

```
cd ~/PhantomSDR-Plus
./setup-cpufreq-perms.sh
```

Traži vašu lozinku jednom — i samo jednom, pri postavljanju — a zatim:

1. stvara sistemsku grupu `cpufreq` i dodaje vas u nju;
2. instalira `/etc/tmpfiles.d/99-phantomsdr-cpufreq.conf` kako bi se grupa i prava `0664` ponovno primijenili **pri svakom pokretanju sustava** — prava u `sysfs`-u sama ne preživljavaju ponovno pokretanje;
3. primjenjuje promjenu odmah, da možete testirati bez ponovnog pokretanja.

Zatim, ovim redom:

```
# 1. odjavite se i ponovno prijavite (ili ponovno pokrenite sustav) – nova
# grupa dolazi do vaših procesa samo kroz novu prijavu
id | grep cpufreq

# 2. ponovno pokrenite ploču; čuvar pravo pisanja provjerava jednom, pri pokretanju
sudo systemctl restart phantomsdr-admin
# (ovako odjava nije potrebna – systemd iznova gradi popis grupa pri
# svakom pokretanju. Samo ako jedinice NISU instalirane:
# ./manage_admin.sh stop && ./manage_admin.sh start – nikad oboje, bore se za port 3000)

# 3. potvrdite
python3 thermal_guard.py --once
# cpufreq : writable, throttle stage available
```

Za potpuno poništavanje: `sudo ./setup-cpufreq-perms.sh --revoke`.

Napomene

- Throttle faza *djeluje* samo ako je `thermal_mode throttle` ili viši. Sama dozvola ne mijenja ništa.
- Čuvar snižava za 20 % ono ograničenje koje je *tada* na snazi i vraća točno tu vrijednost. Ako već ograničavate procesor (primjerice `MAX_SPEED` u `/etc/init.d/cpufrequtils`), vraća se vaše ograničenje — čuvar vam neće potihom vratiti viši takt od onoga koji ste tražili.
- Neki strojevi uopće ne izlažu `cpufreq` upravljački program — mnogi VPS-ovi, većina kontejnera. Skripta to prepoznaje i ne mijenja ništa. Tamo koristite `stop` ili `stop+restart`; oni ne ovise o `cpufreq-u`.

9. Rad bez nadzora

Najčešće pitanje o čuvaru: *što se događa kad nitko ne gleda?* Kratak odgovor: upravo je za taj slučaj i napravljen.

Ne treba mu otvoren preglednik. Čuvar je pozadinska nit unutar `admin_server.py`, taktirana uzorkivačem grafova svake 2 sekunde. Radi bez obzira je li itko prijavljen u ploču, je li preglednik otvoren i jeste li budni. U samostalnom načinu on je systemd usluga, dakle još neovisniji. Kartica na nadzornoj ploči *prozor* je prema čuvaru, a ne sam čuvar.

Djeluje bez pitanja. Nema dijaloga za potvrdu ni obavijesti koju treba čekati. Na pragu zaustavljanja zaustavlja poslužitelj, i točka. Upravo je to smisao.

Ali samo ako ste mu dali dopuštenje. U načinu `log` vjerno će zabilježiti stroj koji se kuha i neće poduzeti ništa. Ako ovaj odjeljak čitate zato što „*nisam tu da to nadzirem*“, način koji želite je `stop+restart` — jedini koji i štiti stroj i vraća uslugu bez vas.

Pomaže samo dok je ploča živa. Čuvar je dio `admin_server.py`: ploča koja je pala u 02:00 odnosi zaštitu sa sobom, a nakon ponovnog pokretanja sustava stroj ostaje nezaštićen dok se ne prijavite i ponovno je pokrenete. Ako nitko ne gleda, neka gleda systemd: repozitorij donosi `phantomsdr-admin.service` i `phantomsdr-proxy.service`, koji se pokreću pri dizanju sustava i ponovno pokreću ploču u roku od pet sekundi nakon pada. Vidi [ADMIN_PANEL_SETUP.md](#). Za prijamnik bez nadzora ovo se **snažno preporučuje** — uključiti `stop+restart` bez toga štiti vas samo do sljedećeg dizanja sustava.

Kako ćete saznati poslije. Otprilike po korisnosti:

Gdje	Što dobivate
<code>crash.log</code> / kartica CRASH	Svaki postupak, s vremenskom oznakom. Trajni zapis.
Kartica THERMAL GUARD na nadzornoj ploči	Stanje uživo: temperatura, faza, lockout, zaustavljanja u zadnjem satu.
Stranica Graphs (temperatura procesora, 24 h)	Oblik događaja — koliko brzo je rasla, koliko je trajalo hlađenje.
<code>journalctl -u thermal-guard</code>	Samo u samostalnom načinu.

Razumna rutina za lokaciju bez nadzora: naoružan `stop+restart` i `grep THERMAL crash.log` kad se ionako prijavite. Ako je prazno, nema se što doznati.

Primjer do kraja. Kvar ventilatora u 03:00 na stroju u načinu `stop+restart`:

03:14 temperatura prelazi 88 °C, drži 30 s	→ zabilježeno upozorenje
03:16 prelazi 92 °C, drži 30 s	→ primijenjen throttle (ako je dopušten)
03:19 prelazi 95 °C, drži 60 s	→ STOP – stop skripta, lockout uključen
03:19 watchdog ponovno pokreće poslužitelj	→ čuvar ga opet zaustavlja i bilježi
03:26 procesor je 5 minuta ispod 75 °C	→ lockout otpušten, start skripta
03:41 pregrije se i stane drugi put	→ ograničenje dosegnuto, nema više pokretanja
	poslužitelj ostaje ugašen, zaštita ostaje

Budite se uz ugašen WebSDR, `crash.log` koji točno kaže zašto i — što je najvažnije — procesor koji se nikad nije približio svojoj kritičnoj točki.

10. Isprobajte ga prije nego mu povjerite stroj

Nemojte čekati pravi toplinski val da biste doznali radi li vaša stop skripta.

`thermal_test_temp` navodi čuvara da vjeruje kako je procesor na temperaturi koju vi odaberete. Sve ostalo ponaša se posve normalno — vremena zadržavanja, ljestve, lockout, vaše stvarne skripte.

1. Ploča → **Settings** → **THERMAL GUARD** → **TEST TEMPERATURE**.
2. Unesite vrijednost iznad svog praga zaustavljanja (primjerice 97 na Intelu).
3. Spremite i pratite karticu na nadzornoj ploči te karticu CRASH.
4. **Ispraznite polje kad završite.** Zaostala testna temperatura znači da čuvar više ne čita stvarni senzor.

U načinu `log` ovo vam pokazuje cijele ljestve bez ikakvog rizika — siguran prvi test i onaj koji treba obaviti prije naoružavanja bilo čega.

U načinu `stop+restart` ovo je **test uživo**: vaš će se poslužitelj doista zaustaviti i doista vratiti otprilike pet minuta nakon što ispraznite polje. Učinite to kad nitko ne sluša. Vrijedi to obaviti jednom, jer je to jedini način da znate da vaše stop i start skripte rade kad ih pozove nešto drugo, a ne vi.

Istovrijednice u samostalnom načinu:

<code>python3 thermal_guard.py --once</code>	# samo pragovi, bez djelovanja
<code>python3 thermal_guard.py --mode log</code>	# gledanje ljestava uživo

11. Čitanje dnevnika

Sve se dodaje u `crash.log` pokraj konfiguracijske datoteke, jedan događaj po retku, svaki s prefiksom `[THERMAL]`:

```
grep THERMAL crash.log
```

Reci su namjerno jednoreččani i prikladni za `grep`. Što ćete vidjeti:

Redak	Značenje
warn: 88.4C (warn=88.0 crit=100) sustained 30s	Prva prečka. Ništa nije učinjeno.
throttle: 92.1C sustained 30s – cpufreq max lowered 20%	Takt procesora snižen.
throttle level reached: ... (no cpufreq write access – stage skipped)	Vidi odjeljak 8.
STOP level reached: ... mode is 'log', NOT stopping	Bio bi zaustavio poslužitelj. Naoružajte ga.
STOPPED server: 95.2C sustained 60s (stop=95.0 crit=100) – ran stop-websdr.sh	Pravi slučaj.
lockout: process reappeared at 96.0C – re-stopped (...) [12 time(s) so far]	Nešto ponovno pokreće vaš poslužitelj; čuvar pobjeđuje. Ograničeno na jedan redak u minuti.
rate limit: 2 thermal stops within the hour – automatic restart is now DISABLED	Ostaje ugašen da istražite. Zaštita se nastavlja.
recovered: 54.0C held below 75.0 – lockout cleared	Opet hladno.
auto-restart: ran start-rx888mk2.sh	Ponovno na mreži.
back to normal: 59.0C (warn=88.0)	Sišao s ljestava a da se nikad nije zaustavio.

12. Lockout i ograničenje broja zaustavljanja

Ova dva ponašanja najviše iznenađuju, pa ih vrijedi jasno izreći.

Lockout je ono što čini da čuvar djeluje upravo protiv vaše postave. Kad zaustavi poslužitelj, ne šalje samo jednu naredbu i nada se. Označi se zaključanim i u svakom ciklusu od 2 sekunde, dok je temperatura iznad praga povratka, provjerava je li se proces vratio — i ako jest, opet ga zaustavlja. Vaš `watchdog`, `systemdov Restart=always`, `cron` posao, nestrpljiv `sysop`: svi gube tu raspravu dok se procesor ne ohladi. Otpušta se sam čim temperatura ostane ispod praga povratka `thermal_resume_s` (zadano 5 minuta).

Gumb CLEAR LOCKOUT na kartici nadzorne ploče otpušta lockout i brojač zaustavljanja po satu, za slučaj kad ste popravili hlađenje i ne želite čekati. **Ne** razoružava čuvara: ako je stroj još vruć, sljedeća provjera opet će zaustaviti poslužitelj. To je namjerno.

Ograničenje (`thermal_max_stops_hour`, zadano 2) sprječava da neispravan stroj cijelu noć ide gore-dolje. Kad se ograničenje dosegne, čuvar **isključuje automatsko ponovno pokretanje** i ostavlja poslužitelj ugašenim. Pročitajte pažljivo: isključuje ponovno pokretanje, a ne zaštitu. Lockout i dalje zaustavlja poslužitelj dok je pregrijan, koliko god se puta već oglasio. Čuvar koji bi zašutio točno kad je stroju najgore bio bi gori od nikakvog čuvara.

13. Rješavanje problema

sensor : NONE – no trusted CPU sensor on this machine Čuvar nije našao nijedan dopušteni senzor i odbija nagađati. Pokušajte `sensors` (iz `lm-sensors`; `sudo apt install lm-sensors && sudo sensors-detect`). U virtualnom stroju ili kontejneru često doista nema izloženog senzora jezgre — čuvar takav stroj ne može zaštititi i ispravno se proglašava isključenim umjesto da se pretvara.

cpufreq : not writable – throttle stage disabled Očekivano dok ne pokrenete `./setup-cpufreq-perms.sh` — vidi [odjeljak 8](#). Utječe samo na throttle fazu; `stop` i `stop+restart` nisu pogođeni.

Pokrenuo sam `setup-cpufreq-perms.sh`, i dalje piše `not writable` Dva vjerojatna uzroka: niste se otad odjavili i ponovno prijavili (nova grupa dolazi do procesa samo kroz novu prijavu — provjerite s `id`), ili ploča otad nije ponovno pokrenuta (čuvar pravo pisanja provjerava jednom, pri pokretanju). Učinite oboje, tim redom.

Pragovi mi izgledaju pogrešno za moj procesor Provjerite što stroj objavljuje:

`cat /sys/class/hwmon/hwmon*/temp*_crit`. Ako je vaš `crit` neuobičajen ili ga nema, postavite apsolutne vrijednosti u `thermal_stop` i srodne.

Zaustavio mi je poslužitelj, a mislim da nije bio vruć Provjerite je li `thermal_test_temp` ostao postavljen od nekog testa. To je daleko najčešći uzrok.

Nikad se ne oglasi iako se stroj grije Potvrdite da način nije `log`, da je `thermal_enabled` postavljen na `true`, i usporedite temperaturu koju vidite s pragovima iz `--once`. Zapamtite da prag mora biti prijeđen *neprekidno* kroz cijelo vrijeme zadržavanja.

Poslužitelj se stalno ponovno pokreće tijekom toplinskog događaja To je `lockout` na djelu, a reci `lockout`: njegov su izvještaj o uspjehu — vaš nadzornik nastavlja pokušavati, čuvar nastavlja poništavati. Nema se što popravljati.

Ploča ne radi, radi li onda čuvar? Ne — u načinu s pločom čuvar živi unutar `admin_server.py`. Želite li zaštitu neovisnu o ploči, koristite samostalnu uslugu iz [odjeljka 7](#).

14. Isključivanje ili uklanjanje

- **Pauza:** postavite `thermal_mode` na `log`. Nastavlja promatrati i izvještavati, ali nikad ne djeluje.
- **Potpuno isključivanje:** postavite `thermal_enabled` na `false`.
- **Samostalno:** `sudo systemctl disable --now thermal-guard`.
- **Poništavanje `cpufreq` prava:** `sudo ./setup-cpufreq-perms.sh --revoke`.

Brisanje `thermal_guard.py` također je sigurno — `admin_server.py` uvozi ga unutar `try` i jednostavno prijavi čuvara kao nedostupnog ako ga nema.

Vidi također

- [Postavljanje administratorske ploče](#) — kartica Thermal Guard u kontekstu
- [Instalacija](#) — toplinska zaštita pri svježoj instalaciji
- [Struktura projekta](#) — gdje se nalazi `thermal_guard.py`

POGLAVLJE 5

Ograničenja veza

CONNECTION_LIMITS.md

Ograničenja veza — Priručnik za sysopa

Zaštita javnog prijemnika od poplave veza.

WebSDR je javna usluga koja svakome tko zatraži predaje neprekidan tok zvuka i spektra. To mu je i svrha, i upravo je u tome problem: poslužitelju ništa kod običnog slušatelja ne izgleda drukčije od nekoga tko otvara sesije u petlji. Dana 16. rujna 2026. ovaj je prijemnik doživio točno to — jedna je adresa otvorila 134 slušateljske sesije unutar jedne minute i držala ih 181 istodobno, dok je stvarna publika stanice bila oko desetak ljudi.

Ovaj priručnik govori o tri sloja koja to zaustavljaju, o tome što svaki od njih može a što ne može, i kako provjeriti jesu li vaši razumno postavljeni.

U žurbi? Primjerni `config.toml` već dolazi s razumnim vrijednostima, pa je svježa instalacija zaštićena bez da išta učinite. Ako je vaš vlastiti `.toml` stariji i nema ključeva `[limits]` osim `audio/waterfall/events`, ograničenja po adresi jednostavno su **isključena** — prekopirajte blok iz primjera da ih uključite. Jedina naredba koju poslije vrijedi znati jest `grep Refused spectrumserver.log`, koja vam kaže odbija li se koga.

Sadržaj

1. Od čega se zapravo branite
2. Tri sloja
3. Ograničenja po adresi — politika slušatelja
4. Neaktivne veze — tihi napad
5. Zaštita u jezgri — `setup-firewall.sh`
6. Referenca postavki
7. Što vidi odbijeni slušatelj
8. Čitanje zapisa i brojača
9. Odabir vaših brojeva
10. Zamke koje vrijedi poznavati
11. Isključivanje svega

1. Od čega se zapravo branite

Vrijedi biti precizan, jer riječ „DDoS“ pokriva dvije vrlo različite stvari, a samo se jedna od njih može riješiti na vašem stroju.

Poplava iz jednog izvora. Jedno računalo, ili njih nekoliko, otvara veze najbrže što može. To se ovdje dogodilo i u potpunosti se da popraviti: veze dolaze s adrese koju možete prebrojati, a brojanje je dovoljno.

Pravi raspodijeljeni napad. Tisuće računala, često s lažiranim izvorišnim adresama, pune vašu vezu. Dok ti paketi stignu do vaše mrežne kartice, propusnost je već potrošena. **Ništa opisano u ovom priručniku ne pomaže**, niti to može išta što instalirate na poslužitelj — jedini je odgovor usluga ispred vas, što je drugi razgovor koji uključuje pravu domenu i pružatelja poput Cloudflarea ispred prijemnika. Sve što slijedi bavi se prvim slučajem. To nije ograničenje zbog kojeg se treba ispričavati: prvi je slučaj ono što se amaterskim stanicama doista događa.

2. Tri sloja



Namjerno nisu suvišni jedan drugome. Sloj jezgre je brz, ali glup — broji pakete s adrese i ne zna ništa o slušateljima. Sloj poslužitelja je precizan, ali vezu vidi tek kad je prihvaćena i raščlanjena, što nešto košta. Sloj preglednika postoji da odbijeni slušatelj sazna što se dogodilo, umjesto da gleda u stranicu koja se nikad ne učitava.

3. Ograničenja po adresi

Što se broji kao jedan slušatelj

Slušatelj je **jedna /audio veza** — ili, gdje je [kiwi_emulation] uključen, **jedna Kiwi zvučna veza**, koja se broji potpuno isto. To je ista jedinica kojom se već služe brojač korisnika, `users.json` i oznake na slapu, pa Kiwi klijent zauzima mjesto i pojavljuje se na popisu korisnika jednako kao preglednik.

To je važno jer jedna jedina kartica preglednika na računalu otvara **četiri** veze prema vašem poslužitelju:

Veza	Svrha
<code>/audio</code>	tok demoduliranog zvuka — to je slušatelj
<code>/waterfall</code>	prikaz spektra
<code>/events</code>	popis korisnika, frekvencije ostalih slušatelja
<code>/chat</code>	ploča za razgovor

Telefon na `/mobile` otvara dvije (`/audio` i `/chat`), jer nema slapa. Ograničenje koje bi brojalo sirove veze odbilo bi dakle sve: tri kartice su dvanaest veza.

Tri ključa

```
[limits]
per_ip=3          # istodobni slušatelji s jedne adrese
per_ip_sockets=0  # istodobne veze bilo koje vrste; 0 daje per_ip * 4 + 4
per_ip_rate=40    # nove veze u minuti s jedne adrese
per_ip_ban_s=600  # koliko se dugo odbija adresa koja probije brzinu
```

`per_ip` je glavni broj i onaj o kojem ćete razmišljati. Tri je dobra polazna točka.

`per_ip_sockets` postoji jer `/audio` nije jedino što se isplati preplaviti. `/waterfall` je najteži tok koji vaš poslužitelj proizvodi, a ograničenje koje bi brojalo samo slušatelje zaobišlo bi se jednostavno otvaranjem veza slapa. Ostavljeno na 0 postaje `per_ip * 4 + 4`, što stane `per_ip` punih kartica s malo prostora za veze koje jezgra nije stigla pospremiti nakon osvježavanja stranice.

`per_ip_rate` je ono što poplavu doista prekida. Sama gornja granica to ne čini: odbijanje veze jeftino je, ali ne besplatno, a napadač se može ponovno spajati jednako brzo koliko se odbijanja vraćaju, trošeći vaš procesor na postavljanje i rastavljanje. Ograničenje brzine daje mu zabranu umjesto odgovora. Zapamtite da jedna kartica košta četiri, pa je 40 u minuti otprilike deset učitavanja stranice u minuti s jedne adrese.

Namjeran redoslijed. Brzina se naplaćuje *prije* nego se provjeri granica istodobnosti. To izgleda naopako dok ne pomislite na adresu koja je već na svojoj granici i ponovno se spaja u petlji: kad bi joj svaki put odgovaralo jeftino odbijanje „na granici si“, prekršaji brzine nikad se ne bi nakupili i zabrana koja petlju prekida nikad se ne bi okinula.

Tko je izuzet

Povratna petlja nikad se ne ograničava. Autorun tap za dojavu spotova, upravljačka ploča i preglednik otvoren na samom poslužiteljskom stroju svi dolaze s `127.0.0.1`, a pustiti ih da troše mjesta značilo bi da vaša vlastita stanica zaključava vaše slušatelje vani.

To pokriva i `proxy.py`: veze kroz proxy dolaze do `spectrumservera` s povratne petlje, a stvarna adresa klijenta stiže u zaglavlju `X-Forwarded-For`, na koje su ograničenja i vezana.

Od v5.0.0 tom se zaglavlju vjeruje **samo ako dolazi s povratne petlje** — od proxyja na istom računalu. Ako ga pošalje bilo tko drugi, zanemaruje se i vrijedi TCP adresa, jer bi inače izravni posjetitelj mogao napisati `X-Forwarded-For: 127.0.0.1` i proći kao povratna petlja mimo svih ograničenja s ove stranice. `proxy.py` usto odbacuje svaki `X-Forwarded-For` ili `X-Real-IP` koji klijent pošalje prije nego doda svoje, a na izbacivanje od strane sysopa (`/~kick`) odgovara samo klijentima na istom računalu. Stanica s više prijemnika iza proxyja opisana je u [Više prijemnika](#); ondje ograničenja vrijede po prijemniku.

4. Neaktivne veze

Postoji napad koji ništa od navedenog ne može vidjeti.

Otvoriti TCP vezu. Ne poslati ništa. Držati je.

Takav sugovornik nikad ne dovrši zahtjev, pa nikad ne dođe do koda u kojem žive ograničenja po adresi — nema slušatelja, nema objekta klijenta, nema ničega s imenom što bi se brojalo. U međuvremenu njega to košta nekoliko bajtova, a vas opisnik datoteke i jedno mjesto dokle god je drži.

```
[limits]
idle_per_ip=8      # neaktivne veze (prije zahtjeva) po adresi
idle_total=512     # gornja granica preko svih adresa
```

Ograničenje je na **koliko njih**, a ne na **koliko dugo**, i ta je razlika cijela zamisao. **Roka za neaktivnu vezu nema, i to namjerno.** Rok djeluje kao očit odgovor, a zamka je (vidi §10): postoje legitimni sugovornici koji se spoje pa dugo šute prije nego progovore. Od napada ih razdvaja količina — legitiman sugovornik drži jednu takvu vezu, napadač tisuće.

Neaktivna veza tako živi dok ne pošalje zahtjev ili ne umre, a mjesto se otpušta u oba slučaja.

`idle_total` omeđuje poplavu razasutu po mnogo izvorišnih adresa. Kad se granica dosegne, odbijaju se **nove** neaktivne veze umjesto da se izbacuju postojeće, pa legitimna dugovječna veza nikad nije ono što se žrtvuje da bi se napravilo mjesta.

Oboje se primjenjuje u trenutku prihvatanja utičnice, na temelju adrese sugovornika — zahtjeva još nema, pa nema ni `X-Forwarded-For` za pogledati. To ne smeta: povratna petlja se ne broji, čime je pokriveno sve što dolazi kroz `proxy.py`, a izravna poplava nosi vlastitu adresu.

5. Zaštita u jezgri

`setup-firewall.sh` postavlja nftables tablicu koja skida količinu prije nego `spectrumserver` pročita i jedan bajt. Namjerno je gruba, a brojevi su joj znatno iznad onih u `config.toml`, tako da običan slušatelj nikad ne može biti uhvaćen objema.

```
./setup-firewall.sh --show      # ispiši pravila, ne mijenjaj ništa
sudo ./setup-firewall.sh --check # provjeri valjanost na vašoj jezgri
sudo ./setup-firewall.sh --apply # učitaj, uz automatski povrat za 60 s
sudo ./setup-firewall.sh --persist # ponovno učitaj pri pokretanju
sudo ./setup-firewall.sh --status # brojači paketa po pravilu
sudo ./setup-firewall.sh --remove # poništi sve
```

Pokriva četiri stvari: gornju granicu istodobnih veza po izvorišnoj adresi na vašim proxy i prijemničkim vratima, brzinu novih veza po izvoru, kočnicu za grubo pogađanje SSH lozinki, te Windows dijeljenje datoteka zatvoreno za sve izvan privatnih adresnih raspona — `smbd` na mnogim strojevima sluša na `0.0.0.0`, a je li to izvana dohvatljivo ovisi o usmjerivaču koji skripta ne može vidjeti.

Vrata se čitaju iz `admin_config.json`, iste datoteke kojom se služi `proxy.py`, pa prati vašu instalaciju.

Zašto vas ne može zaključati vani

Dva svojstva, oba namjerna:

- Politika tablice je **accept**, i odbacuje samo izrijekom imenovane obrasce. Pravilo koje ne odgovara pušta paket dalje prema svemu ostalom što ste postavili. Ne može stroj učiniti nedostupnim.
- **Uspostavljene veze prihvaćaju se u prvom pravilu.** SSH sesija u kojoj tipkate nikad nije pogođena onim što slijedi.

Povrh toga, `--apply` naoruža automatski povrat: učitaj pravila, pa potvrdi unutar 60 sekundi ili se uklanjaju. Ne recite ništa, zatvorite terminal, izgubite vezu — skup pravila nestaje sam od sebe.

Iskoristite taj prozor kako treba. Provjerite **s drugog uređaja, izvan vlastite mreže**, da se prijemnik i dalje učitava i da se otvara nova SSH sesija. Provjera sesijom koju već imate ne dokazuje ništa, jer ju je prihvatilo prvo pravilo.

6. Referenca postavki

Svaki ključ ispod živi pod `[limits]` u vašoj `.toml` datoteci. **Svi su podrazumijevano isključeni ili velikodušni**, pa se postava koja ih ne spominje ponaša točno kao i prije.

Ključ	Zadano	Značenje
<code>per_ip</code>	0 (isključeno)	Istodobni slušatelji po adresi
<code>per_ip_sockets</code>	$0 \rightarrow \text{per_ip} * 4 + 4$	Istodobne veze bilo koje vrste po adresi
<code>per_ip_rate</code>	0 (isključeno)	Nove veze u minuti po adresi
<code>per_ip_ban_s</code>	600	Sekundi odbijanja nakon probijanja brzine
<code>idle_per_ip</code>	8	Neaktivne veze, prije zahtjeva, po adresi
<code>idle_total</code>	512	Neaktivne veze preko svih adresa

Postavljanje bilo kojeg ključa na 0 isključuje tu pojedinu provjeru.

Tri koja nisu ograničenja

`[limits]` sadrži i `audio`, `waterfall` te `events`, naslijeđene iz izvornog PhantomSDR-a, i **nijedno od tri se ne provodi**. `waterfall` i `events` ne čita nikakav kôd. `audio` se čita na samo jednom mjestu: šalje se kao `max_users` u JSON-u prijave direktorijima iz `[websdr] register_urls`, pa je to broj koji prikazuju `sdr-list.xyz` i ostali, a ne čini baš ništa kad je `register_online=false`. Želite li gornju granicu slušatelja, `per_ip` gore je ono što djeluje.

Još jedan, pod `[server]`

```
[server]
min_client_version=0
```

Nije ograničenje nego odbijanje, a kako dijeli isti kôd zatvaranja, mjesto mu je ovdje. Stranice najavljuju inačicu iz koje su učitane kao `/audio?v=N`, a sve ispod `min_client_version` odbija se porukom „*ova je stranica zastarjela — ponovno je učitajte*“.

Postoji zato što poslužitelj ne može doprijeti do JavaScripta koji već radi u nečijem pregledniku. Kartica zadržava kôd koji je učitala dok je netko ne osvježi, pa promjena stranice stiže samo do slušatelja koji je slučajno osvježe — a stanica koja je upravo promijenila ponašanje sesija možda treba da stare stranice nestanu sada, a ne jednom.

Provjerava se samo `/audio`, jer ondje sesija živi, dok su povratna petlja i Kiwi putanje izuzete kako autorun tap i Kiwi klijenti ostanu netaknuti. 0 ga isključuje i zadana je vrijednost. Ostaviti ga uključenim nakon što se stare stranice isprazne nije problem: trenutni klijent `diversityja` koji bude odbijen na golom `/audio` pokušava ponovno s oznakom, pa se i dalje odbijaju samo klijenti `diversityja` sa starijom verzijom PhantomSDR-Plusa i alati trećih strana koji otvaraju `/audio` bez `?v=`. Vratite ga na 0 ako vam druga stanica javi da vas više ne može koristiti kao partnera za `diversity`.

7. Što vidi odbijeni slušatelj

Ništa se nikad ponovno ne spaja

Počnite odavde, jer to upravlja svime ostalim: **pala** `/audio` **veza završava sesiju**. Automatskog ponovnog spajanja nema, i to namjerno.

Ponovno spajanje pravi je nagon za aplikaciju za razgovor, a krivi za prijemnik. `/waterfall` i `/events` s njim se nikad nisu vraćali, pa je ponovljena sesija bila živa zvučna veza pričvršćena na zamrznuti slap; a zaustavljanje poslužitelja više ga nije praznilo, jer je svaki slušatelj bio natrag nekoliko sekundi poslije, puneći popis korisnika sesijama u kojima zapravo nikoga nije bilo. Stranica utihne i slušatelj je ponovno učita — što je upravo ono što je ovaj prijemnik radio prije nego što je ponovno spajanje uopće dodano.

Za ograničenja iz ovog priručnika to uklanja i zamku: pred ograničenjem brzine automatski bi pokušaj produljio upravo onu zabranu koju je htio zaobići.

Dva koda zatvaranja

Kôd	Značenje	Tko ga šalje
4003	odbijeno — iznad ograničenja po adresi, ili zastarjela stranica	spectrumserver
4001	izbačen od strane sysopa	spectrumserver , kad ploča pozove <code>/~~kick</code>

Oba su u privatnom rasponu 4000–4999, pa se nijedan ne može zamijeniti sa stanjem protokola, i oba svaka stranica smatra konačnima. Tekst razloga putuje u okviru zatvaranja, a ne kao poruka, jer je prvi okvir na `/audio` rezerviran za postavke prijemnika i bilo što poslano prije njega pokvarilo bi svaku normalnu vezu.

Na stranici

Odbijeno pri učitavanju stranice — slušatelj uopće ne krene, pa se obje stranice objasne. Stolno računalo prikazuje ploču: „*Ovaj je prijemnik odbio vezu*“, razlog, i napomenu da se svi koji dijele jednu internetsku vezu broje kao jedna adresa. `/mobile` prikazuje isto u svom prostoru za obavijesti.

Odbijeno ili izbačeno usred sesije — stolna stranica jednostavno stane, namjerno bez poruke: petlja crtanja slapa prestaje i to je sve. `/mobile` prikazuje jedan redak koji slušatelju kaže da osvježi stranicu, osim nakon izbacivanja, gdje i ono šuti.

Ta je šutnja nakon izbacivanja namjerna. **Izbacivanje je sysop koji završava sesiju, a ne kazna** — ništa se ne objavljuje, a tko se želi vratiti ponovno učita stranicu i sekundu poslije je običan slušatelj. Jedino što kôd 4001 donosi jest da preglednik zna kako ovo nije bila pala veza, pa izbacivanje doista uhvati umjesto da samo sebe poništi.

Izbacivanje obavlja `spectrumserver` sam, a ne `proxy.py`: slušatelji se spajaju izravno na priključak prijemnika, pa `proxy` ne posjeduje nijednu njihovu vezu i njegovo izbacivanje ne doseže nikoga. Ploča poziva `/~~kick?ip=&secs=`, dostupan samo s povratne petlje i provjeren na stvarnom TCP sugovorniku, a ne na `X-Forwarded-For`, koji šalje klijent. `secs` je zadano nula — bez zabrane — ali prima trajanje ako adresu želite zadržati izvan neko vrijeme, a to koristi isti mehanizam zabrane kao `per_ip_rate`, pa se ponovno spajanje odbija na vratima.

8. Čitanje zapisa i brojača

Poslužitelj

```
grep Refused spectrumserver.log
```

```
Refused /audio from 203.0.113.5: too many simultaneous connections
Refused /audio from 198.51.100.7: too many connection attempts
```

Ako tamo nema ničega, nikoga se ne odbija. Zapisivanje je ograničeno na najviše jedan redak po adresi svakih pet sekundi — bez toga bi se napad pretvorio u neograničeno pisanje na disk, što je samo sporiji način da se stanica obori.

Jezgra

```
sudo ./setup-firewall.sh --status
```

Čitajte **counter packets N** na svakom pravilu; to je broj doista odbačenih. Sve nule znače da ništa nije blokirano i da su vaši pragovi udobno labavi.

Nemojte se uznemiriti popisima `elements = { ... }` u skupovima iznad pravila. To su vaši obični slušatelji koje se *prati* u odnosu na ograničenja — unos se pojavi pri prvoj vezi neke adrese i istekne koju minutu poslije. Praćenje nije blokiranje.

9. Odabir vaših brojeva

Nagađanje nije potrebno; vaši će vam vlastiti zapisi reći. Ovo je mjerenje koje je ovdje odredilo `per_ip=3`, nad tjedan dana `logs/users_*.jsonl`:

Istodobne sesije s jedne adrese	Broj adresa
1	682
2	54
3	13
više od 3	8

758 različitih adresa u sedam dana, a ograničenje od tri dotaklo bi osam njih. Poplava je sjedila na 181.

Da to ponovite na vlastitoj stanici, uparite identifikatore sesija po adresi u `logs/users_*.jsonl` i uzmite najveće preklapanje. Imajte na umu da zapis bilježi događaje `tune` i `disconnect`, ali nema događaja `connect`, pa se početak sesije mora izvesti iz njezina prvog `tune`.

Ne zaboravite NAT. Klub, škola, ured ili mobilni operater pojavljuju se kao jedna adresa, a svi iza nje dijele jedno dopuštenje. To je svojstveno svakom ograničenju vezanom uz adresu; jedino je ublažavanje odabrati broj s kojim vam je ugodno. Ako slušatelj ikad javi da se ne može spojiti, to je prvo na što treba posumnjati — pogledajte `grep Refused` prije svega ostalog.

10. Zamke koje vrijedi poznavati

Nikad rok za neaktivnu vezu

Ključa `handshake_s` više nema, i evo zašto. Rok od 30 sekundi za neaktivne veze izgleda kao očita obrana od slowlorisa. Napisan je, uključen i u jednoj je minuti dao ovo:

```
Idle deadline: closed 1 connection(s) idle past 30s: 192.87.173.88
```

192.87.173.88 je `etgd-websdr.ewi.utwente.nl` — **vlastito računalo za povratni poziv websdr.org**. Ono se spaja natrag na vaš poslužitelj i zatim miruje, znatno dulje od 30 sekundi, prije nego pošalje svoj `GET /~orgstatus`. Rok ga je svaki put presjekao i prijava je prestala raditi. Postavka je uklonjena, a ne

ostavljena kao opcija: samoranjavanje koje kviri vaš unos u direktoriju ne vrijedi šezdeset redaka koji ga izvode.

Isto vrijedi za `open_handshake_timeout` u `websocketpp`, koji je još tu. Vrijednost od deset minuta u `spectrumserver.cpp` izgleda besmisleno i nosiva je: upravo ona dopušta povratnom pozivu `websdr.org` da miruje. Postoji i drugi razlog — rukovatelj `/~orgstatus` odgovara iz dretve koja drži `dup()` utičnice, a `websocketpp` isteklu vezu završava s `shutdown()`, pozivom na razini utičnice koji seže kroz duplikat i pokvario bi povratni poziv čak i da vrijeme mirovanja nije problem.

Trebate li omeđiti tihe veze, omeđite njihov broj, a ne starost. Za to postoje `idle_per_ip` i `idle_total`.

nftables datoteka mora biti idempotentna

`nft -f` **dodaje** u tablicu koja već postoji, umjesto da je zamijeni. Datoteka pravila bez uvoda `delete table` stoga udvostručuje cijeli lanac pri svakom ponovnom učitavanju — što je točno ono što naivna `systemd` jedinica čini pri svakom ponovnom pokretanju usluge. `setup-firewall.sh` uvod stvara umjesto vas; pišete li vlastita pravila, učinite isto.

```
sudo ./setup-firewall.sh --status | grep -c dport # treba dati 8, ne 16
```

11. Isključivanje svega

Ograničenja poslužitelja: postavite ključeve na `0` ili ih izbrišite iz svog `.toml`-a i ponovno pokrenite prijemnik. Drugog stanja nema; ništa ne preživljava ponovno pokretanje.

Vatrozid:

```
sudo ./setup-firewall.sh --remove
```

To briše tablicu, uklanja spremljeni skup pravila i onemogućuje uslugu pri pokretanju. Vaš se stroj vraća točno u stanje u kojem je bio prije — što na većini instalacija znači bez ikakvog vatrozida. O tome vrijedi razmisliti prije uklanjanja.

POGLAVLJE 6

Više prijemnika

MULTI_RECEIVER.md

Više prijemnika — Priručnik za sysopa

Dva ili više prijemnika na jednom računalu, s izbornikom prijemnika na stranici.

Od v5.0.0 jedno PhantomSDR-Plus računalo može istodobno pokretati više prijemnika — na primjer RX-888 za kratki val i RTL-SDR za 2 m — a slušatelji prelaze s jednog na drugi gumbima u zaglavlju stranice, kao što OpenWebRX nudi svoje profile. Svaki prijemnik zadržava vlastiti vodopad, chat, oznake, popis slušatelja i podatke o stanici; nitko na jednom prijemniku nikada ne smeta nikome na drugom.

Žurite? Stanici s jednim prijemnikom ništa od ovoga ne treba i ne vidi nikakvu promjenu. Za drugi prijemnik: napravite `instances/<ime>/` s vlastitim `config.toml`, pokrenite ga s `INSTANCE=<ime> ./start-<radio>.sh`, upišite oba prijemnika u `receivers.toml` i ponovno pokrenite proxy. Primjer u [odjeljku 3](#) prolazi to redak po redak.

Najjednostavnije: pokrenite `./add-receiver.sh` (instalacijske skripte nude ga na kraju). Pita koji prijemnik i što treba pokrivati, instalira njegov upravljački program, zapisuje sve što je opisano u [odjeljku 3](#) i [odjeljku 4](#) te nudi pokretanje prijemnika i ponovno pokretanje proxyja. Ostatak priručnika objašnjava što radi, za slučaj da nešto želite promijeniti ručno.

Sadržaj

1. Kako radi
2. Dva načina objave prijemnika
3. Dodavanje drugog prijemnika
4. `receivers.toml`
5. Pokretanje i zaustavljanje
6. Što vide slušatelji
7. S-metar iznad 30 MHz
8. Sigurnost
9. Koliko prijemnika stane
10. Zamke koje treba znati

1. Kako radi

Svaki prijemnik je potpun, zaseban spectrumserver, s vlastitim SDR-om, vlastitom konfiguracijom i vlastitim unutarnjim portom. Nikada ne dijele uzorke, postavke ni slušatelje. Povezuje ih `proxy.py`, obrnuti proxy koji već posluhuje administratorsku ploču: čita `receivers.toml` i šalje svaki zahtjev jednom prijemniku.

```
slušatelji → proxy.py {
  → :8900 spectrumserver RX-888 (KV, glavni prijemnik)
  → 127.0.0.1:9002 spectrumserver RTL-SDR (2 m, instanca "vhf")
}
```

Proxy odlučuje prema ovome, ovim redom:

1. `?rx=<id>` u adresi — `http://vas.host:8899/?rx=vhf`. Stranica ga dodaje svakom socketu, upitu i poveznici koju otvara, pa kartica ostaje na svom prijemniku.
2. Kolačić `rx`, koji proxy postavlja kad god zahtjev nosi valjani `?rx=`.
3. Zaglavlje `Host`, ako neki prijemnik navodi to ime pod `hostnames`.
4. Inače zadani prijemnik.

Glavni prijemnik je onaj koji se pokreće na uobičajen način (`./start-rx888mk2.sh`). Svaki drugi prijemnik je **imenovana instanca**: pokreće se s `INSTANCE=<ime>`, konfigurira u `instances/<ime>/` i ima vlastiti zapisnik (`logwebsdr-<ime>.txt`), zapisnik poslužitelja, zaključavanje watchdoga i FIFO. Proces i se razlikuju po oznaci `PHANTOMSDR_INSTANCE` u njihovom okruženju, a ne po imenu, pa dva prijemnika mogu pokretati potpuno isti program — dva `rtl_sdr`, ili `rx_sdr` za RSP1A i Airspy — a ponovno pokretanje jednoga nikada ne dira drugi.

2. Dva načina objave prijemnika

A — glavni prijemnik zadržava svoj port. Glavni prijemnik ostaje točno gdje je bio (npr. `:8900`), njegovi slušatelji ništa ne primjećuju, a do ostalih prijemnika dolazi se preko porta proxyja (`proxy_port` u `admin_config.json`, npr. `:8899`) s `?rx=`. Ništa se ne seli i ništa ne izlazi iz etera; zauzvrat postoje dva javna porta.

```
KV: http://vas.host:8900/
2 m: http://vas.host:8899/?rx=vhf
```

B — jedan javni port za sve. Proxy preuzima javni port, a glavni prijemnik seli se na unutarnji. Postavite `[front] port` u `receivers.toml`, premjestite `[server] port` glavnog prijemnika i u njemu postavite `host = "127.0.0.1"`, u `public_port` datoteke `admin_config.json` upišite novi unutarnji port i dodajte `public_port = <javni port>` pod `[websdr]`, kako bi imenik i dalje objavljivao ispravan. Tada je sve iza jednog porta; cijena je kratak prekid tijekom selidbe.

```
KV: http://vas.host:8900/
2 m: http://vas.host:8900/?rx=vhf
```

Izbornik prijemnika radi jednako u oba slučaja. Raspored A je sigurniji početak; B može doći kasnije bez ikakve promjene na drugom prijemniku.

3. Dodavanje drugog prijemnika

add-receiver.sh obavlja sve korake u nastavku — pokrenite ga i odgovorite na pitanja; instalira i upravljački program (`setup-rtlsdr.sh` za RTL-SDR, odgovarajući `setup-*.sh` za ostale), bira slobodan unutarnji port i imenuje prijemnik prema onome što pokriva. Koraci su ovdje razloženi kako biste mogli provjeriti njegov rad ili ga kasnije promijeniti.

Primjer dodaje RTL-SDR Blog V4 za 2 m kao instancu `vhf`, uz RX-888 koji zadržava port 8900 (raspored A).

1. Upravljački program. RTL-SDR Blog V4 treba upravljački program tvrtke RTL-SDR Blog umjesto paketa `rtl-sdr` iz distribucije, a DVB-T upravljački program jezgre mora otpustiti stick. `install.sh` (opcija prijemnika 2, „RTL-SDR Blog V4: da“) radi oboje. Provjerite s `rtl_test -t`, koji mora ispisati `RTL-SDR Blog V4 Detected`.

2. Mapa instance. Sve što pripada prijemniku nalazi se u `instances/vhf/`, koja se nikada ne commita i koju nijedno ažuriranje ne prepisuje:

```
instances/vhf/
├─ config.toml          konfiguracija njegovog spectrumservera (obavezno)
├─ instance.env         neobavezno: argumenti prijemnika i vezanje na jezgre
├─ markers.json         njegove vlastite oznake
├─ www/                 njegove kopije datoteka stanice za stranicu
│   └─ site_information.json
│   └─ wf-message.json
```

spectrumserver radi s `instances/vhf/` kao radnom mapom, pa su povijest chata, oznake, FFTW wisdom i `logs/` također njegovi.

3. config.toml. Krenite od `config-rtl.toml` i promijenite:

```
[server]
port=9002                # vlastiti unutarnji port
host="127.0.0.1"         # dostupan samo preko proxyja
html_root="www/"         # njegovih nekoliko vlastitih datoteka...
html_fallback_root="../../frontend/dist/" # ...a za ostalo zajednička stranica

[input]
sps=2400000              # isto kao -s u instance.env
frequency=145000000       # isto kao -f u instance.env

[input.defaults]
frequency=145500000
modulation="FM"

[kiwi_emulation]
enabled = false          # vidi zamke dolje
```

`html_fallback_root` omogućuje instanci da drži samo vlastite `site_information.json` i `wf-message.json`, dok sve ostalo — sama izgrađena stranica — dolazi iz `frontend/dist`; jedna izgradnja frontenda tako stiže do svakog prijemnika. Kopija `frontend/dist` napravljena simboličkim poveznicama ne radi: poslužitelj odbija datoteke koje završavaju izvan njegove mape.

4. instance.env. Skripta za pokretanje čita ga nakon vlastitih postavki:

```
RX_ARGS="-f 145000000 -s 2400000 -g 29.7 -" # -f i -s isti kao u config.toml
SPECTRUM_CORES=8-11                        # dalje od jezgri glavnog prijemnika
```

5. Podaci njegove stranice. Kopirajte `frontend/site_information.json` u `instances/vhf/www/` i uredite: `siteReceiver`, `siteAntenna`, `siteSDRBaseFrequency` i `siteSDRBandwidth` (oni određuju koji se gumbi opsega prikazuju) te ključeve opisane u [odjeljku 6](#).

6. Pokrenite ga.

```
INSTANCE=vhf ./start-rtl.sh
```

7. Upišite ga u receivers.toml i ponovno pokrenite proxy (`sudo systemctl restart phantomsdr-proxy`). Drugi prijemnik je sada na `http://vas.host:8899/?rx=vhf`, a obje stranice prikazuju izbornik.

4. receivers.toml

Kopirajte `receivers.toml.example` u `receivers.toml`. Bez te datoteke proxy posluhuje jedan prijemnik, kao i uvijek.

```
# [front]
# port = 8900          # samo raspored B: proxy sluša i na ovom javnom portu

[[receiver]]
id      = "hf"
name    = "HF 0-30 MHz (RX-888 MkII)"
port    = 8900
default = true
launcher = "start-rx888mk2.sh"
url      = "http://vas.host:8900/"

[[receiver]]
id      = "vhf"
name    = "2 m (RTL-SDR Blog V4)"
port    = 9002
launcher = "start-rtl.sh"
instance = "vhf"
url      = "http://vas.host:8899/?rx=vhf"
```

Ključ	Značenje
<code>id</code>	Kratko ime za <code>?rx=</code> ; slova, znamenke, <code>-</code> i <code>_</code>
<code>name</code>	Tekst na gumbu prijemnika
<code>port</code>	<code>[server]</code> port njegovog spectrumservera
<code>host</code>	Neobavezno; gdje radi taj spectrumserver (zadano <code>127.0.0.1</code>)
<code>default</code>	Prijemnik za zahtjeve koji ne navode nijedan; najviše jedan
<code>launcher</code>	Njegov <code>start-*.sh</code> , za <code>start-all.sh</code> ; izostavite za prijemnik koji ovo računalo ne pokreće
<code>instance</code>	Ime njegove instance; izostavite za glavni prijemnik
<code>url</code>	Kamo njegov gumb šalje slušatelja; bez njega <code>/?rx=<id></code> na proxyju
<code>hostnames</code>	Neobavezna DNS imena koja vode izravno na ovaj prijemnik

Proxy popis posluhuje i kao `/receivers.json` (id-ovi, imena i poveznice — nikada unutarnji portovi); to čita izbornik prijemnika.

5. Pokretanje i zaustavljanje

Naredba	Radi
<code>./start-rx888mk2.sh</code>	Pokreće ili ponovno pokreće glavni prijemnik
<code>INSTANCE=vhf ./start-rtl.sh</code>	Pokreće ili ponovno pokreće prijemnik <code>vhf</code>
<code>./start-all.sh</code>	Pokreće svaki prijemnik iz <code>receivers.toml</code> koji ima <code>launcher</code>
<code>./stop-websdr.sh vhf</code>	Zaustavlja samo <code>vhf</code>
<code>./stop-websdr.sh main</code>	Zaustavlja samo glavni prijemnik
<code>./stop-websdr.sh</code>	Zaustavlja sve prijemnike, kao i uvijek

Dok je prijemnik zaustavljen, njegov gumb nestaje iz izbornika — proxy navodi samo prijemnike koji odgovaraju, a otvorene stranice ponovno čitaju popis jednom u minuti — i vraća se kad se prijemnik ponovno pokrene; ako ostane raditi samo jedan prijemnik, redak *Receivers:* potpuno nestaje. Zaustavljeni prijemnik ipak se sam vraća kad god se pokreću svi prijemnici: `./start-all.sh`, **Restart** u administratorskoj ploči i ponovno pokretanje od strane toplinske zaštite pokreću sve što je navedeno u `receivers.toml`. Da jedan trajno ostane isključen, uklonite (ili zakomentirajte) njegov blok `[[receiver]]` i ponovno pokrenite proxy.

U administratorskoj ploči postavite **Default start script** na `start-all.sh`. Njezin Stop već zaustavlja sve prijemnike, a Restart i toplinska zaštita koriste skriptu za pokretanje; sa `start-all.sh` vraćaju sve prijemnike umjesto samo glavnoga.

Pomoćni program RADE pripada samo glavnom prijemniku. Imenovana instanca ga nikada ne pokreće niti zaustavlja.

6. Što vide slušatelji

Izbornik prijemnika. Na stanici s više prijemnika zaglavlje stranice dobiva vlastiti redak, *Receivers:*, s jednim gumbom po prijemniku; trenutni je prikazan žuto. Stranica `/mobile` ima iste gumbe kao drugi red u gornjoj traci, a oni otvaraju stranicu `/mobile` drugog prijemnika.

Vlastiti podaci svakog prijemnika. Stranica otvorena na drugom prijemniku prije pokretanja učitava `site_information.json` tog prijemnika, pa su gumbi opsega, početna frekvencija, popis slušatelja, podaci o stanici te poveznice Receiver i Antenna svi njegovi. Običan posjet glavnom prijemniku ne šalje nikakav dodatni zahtjev.

Ključevi u `site_information.json` za to:

Ključ	Značenje
<code>siteReceiverId</code>	Prijemnik kojem ova stranica pripada (njegov <code>id</code>); izostavite na glavnom prijemniku
<code>siteReceiversList</code>	Odakle izbornik čita popis, npr. <code>http://vas.host:8899/receivers.json</code> ; potrebno na prijemniku čija se stranica ne poslužuje preko proxyja
<code>siteReceiverURL</code>	Kamo vodi ime kod <i>Receiver</i> u <i>Open Additional Info</i>
<code>siteAntennaURL</code>	Kamo vodi ime kod <i>Antenna</i> ; "" prikazuje ime bez poveznice

Kad izbornik ostaje skriven. Izbornik prikazuje samo popis koji navodi host pod kojim je stranica otvorena. Otvorena preko adrese lokalne mreže (`http://192.168.1.10:8900/`) ostaje skriven; otvorena kao `http://vas.host:8900/` pojavljuje se. To je namjerno: stanica koja je preuzela tuđi `site_information.json` bez uređivanja nikada ne prikazuje prijemnike te stanice kao svoje.

7. S-metar iznad 30 MHz

Standard IARU Regije 1 postavlja S9 na -73 dBm ispod 30 MHz i na **-93 dBm iznad**, sa šest dB po S-jedinici u oba slučaja. S-metri (analogna kazaljka, digitalna traka i traka na `/mobile`) prate ugođenu frekvenciju: iznad 30 MHz pokazuju VHF S-jedinice, ispod kao i prije. Vrijednosti u dBm i dBμV nikada se ne mijenjaju. Ako želite da instrumenti miruju na 0 dok je kanal prazan, kao na VHF/UHF primopredajniku, dodajte `"siteSMeterGateDb": 6` u `site_information.json` prijemnika (bez ponovne izgradnje): iznad **60 MHz** kazaljka i traka tada se dižu tek kad signal za toliko dB nadmaši šum vlastite okoline (najjača točka propusnog pojasa prema spektru ± 100 kHz oko nje). **Zadano je isključeno** —

kalibrirani instrument koji pokazuje stvarni šum opsega iskrenije je očitavanje. Stranica /mobile, koja nema vodopad, procjenjuje šum iz same primljene razine.

Kalibrirajte dBm VHF prijemnika pomoću `analog_smeter_offset` (kazaljka i brojke) i `smeter_offset` (digitalna traka) pod `[input]` u njegovom `config.toml`, a zatim ponovno pokrenite taj prijemnik. Najprije postavite pojačanje: RTL-SDR javlja razine u odnosu na vlastiti puni opseg, pa svaka promjena pojačanja pomiče očitavanja. Bez generatora signala, terminator od 50 Ω umjesto antene, u USB-u s filtrom od 2,7 kHz, treba pokazivati oko -136 dBm (toplinski šum u 2,7 kHz iznosi -139,7 dBm, plus faktor šuma sticka).

Za RTL-SDR, `add-receiver.sh` zapisuje početnu kalibraciju: fiksno pojačanje `-g 29.7` u `instance.env` (automatsko pojačanje tunera ne može se kalibrirati) te `analog_smeter_offset=-55` i `smeter_offset=-38`, izmjereno na RTL-SDR Blog V4 pri tom pojačanju sa signalom od -71 dBm (63 μ V). Drugi stick istog modela obično odstupa tek nekoliko dB; provjerite ga poznatim signalom i ponovno kalibrirajte ako promijenite pojačanje.

8. Sigurnost

Postavljanje proxyja ispred slušatelja zatvorilo je dvije rupe, obje ispravljene u v5.0.0:

- **Izbacivanje od strane sysopa preko proxyja.** spectrumserver dopušta `/~~kick` samo s vlastitog računala, a za njega svaki zahtjev koji proxy prosljeđuje dolazi s vlastitog računala. Tko god je mogao doći do porta proxyja, mogao je zato odspojiti i blokirati bilo kojeg slušatelja. Proxy sada na `/~~kick` odgovara samo klijentima na istom računalu.
- **Lažne adrese klijenata.** spectrumserver je vjerovao zaglavlju `X-Forwarded-For` od bilo koga, pa se posjetitelj mogao predstaviti kao `127.0.0.1` i zaobići ograničenja po adresi. Zaglavlju se sada vjeruje samo ako dolazi od lokalnog proxyja, a proxy odbacuje svaku kopiju koju klijent pošalje prije nego doda svoju.

Vežite svaki prijemnik do kojeg se dolazi preko proxyja na `127.0.0.1 ([server] host)`, kako se njegov port ne bi mogao koristiti zaobilazeći proxy. Ograničenja po adresi iz [Ograničenja veza](#) vrijede po prijemniku.

9. Koliko prijemnika stane

Softver ne postavlja ograničenje; hardver da. Uskopojasni prijemnici su skromni: RTL-SDR na 2,4 Msps troši oko 5–10 % jedne jezgre, nekoliko desetaka MB memorije i oko 38 Mbit/s USB-a.

- **USB 2.0 je uobičajena granica.** Svi USB 2 uređaji dijele jednu sabirnicu od 480 Mbit/s, čak i u plavoj USB 3 utičnici. Tri do četiri RTL-SDR-a po sabirnici sigurna su brojka; dodajte ih jedan po jedan i pratite svaki vodopad zbog praznina.
- **Napajanje:** za više od dva sticka koristite hub s vlastitim napajanjem.
- **Drugi širokopojasni prijemnik** (još jedan RX-888, ili HackRF na 20 Msps) druga je priča: s prvim se natječe za USB 3, CPU i GPU.
- **Slušatelji koštaju više od prijemnika.** Svaki slušatelj troši CPU na svom prijemniku i propusnost za slanje, pa je ukupan broj slušatelja važniji od broja prijemnika.

10. Zamke koje treba znati

- **Dva sticka istog modela imaju isti serijski broj.** Svaki RTL-SDR Blog V4 javlja 00000001. Dajte svakome vlastiti s `rtl_eeprom -s <serijski>` (s po jednim priključenim stickom) i navedite ga u `instance.env` (`RX_ARGS="-d <serijski> ..."`), inače nakon ponovnog pokretanja stickovi mogu zamijeniti prijemnike.
- **Kiwi klijenti ne mogu birati prijemnik.** KiwiSDR klijent bira golu adresu i port; preko proxyja uvijek završi na zadanom prijemniku. Na ostalima držite `[kiwi_emulation]` isključenim.
- **Kolačići vrijede po hostu, ne po portu.** Posjetitelj koji je koristio drugi prijemnik na `:8899` nosi `rx=vhf` i na `:8900`. Zato stranica odlučuje koji je prijemnik prema `siteReceiverId`, nikada prema kolačiću.
- **Unosi u imenicima.** Dajte svakom prijemniku vlastito ime u `[websdr]`, a na prijemniku iza proxyja postavite `[websdr] public_port` na port koji koriste slušatelji, inače unos oglašava unutarnji.
- **Izgradnja frontenda nakratko prikazuje „Not Found“.** `frontend/dist` se ponovno gradi na mjestu, pa tijekom nekoliko sekundi izgradnje stranica nedostaje na svakom prijemniku.
- **Ništa ne pokreće prijemnike pri pokretanju računala,** osim ako to sami ne uredite. Nakon ponovnog pokretanja `./start-all.sh` vraća ih sve.

POGLAVLJE 7

Poslužitelj statistike

sdr-stats/README.md

Vodič za instalaciju System Stats

Sveobuhvatan vodič za dodavanje nadzora poslužitelja u stvarnom vremenu u vašu PhantomSDR aplikaciju pomoću automatizirane instalacijske skripte.

Pregled

Ova značajka dodaje gumb **Stats** u vaše PhantomSDR sučelje koje prikazuje informacije o sustavu u stvarnom vremenu:

- Korištenje procesora, jezgre, frekvencija, temperatura i vodeći procesi
- Korištenje memorije (iskorišteno/ukupno/postotak)

Statistika se automatski ažurira svakih 5 sekundi dok je modalni prozor otvoren.

Nije isto što i stranica „Grafikoni“ na administratorskoj ploči. Ovaj je prozor trenutačni snimak koji se osvježava dok je otvoren i ne čuva povijest. Administratorska ploča ima zasebnu stranicu **Grafikoni** koja svake 2 sekunde uzorkuje frekvenciju procesora, opterećenje, temperaturu i broj korisnika te ih crta kroz zadnjih 15 minuta do 24 sata — vidi [vodič za administratorsku ploču](#).

Preduvjeti

- **Linux poslužitelj** (preporučuje se Ubuntu/Debian)
- **Internetska veza** (za preuzimanje Node.js-a i paketa)
- **Sudo pristup** (za postavljanje systemd usluge)
- **PhantomSDR-Plus** već instaliran i pokrenut
- **lm-sensors** (preporučeno za temperaturu Intel/AMD procesora - skripta ga automatski instalira)

Metode instalacije

NAPOMENA

Glavna instalacijska skripta to već nudi. `./install.sh` — i četiri distribucijske skripte — pokreću `install-stats-server.sh` prema zadanome u sklopu normalne instalacije PhantomSDR-Plusa, pa je na svježem stroju poslužitelj statistike obično već postavljen. Ovu stranicu koristite da ga instalirate zasebno ili da mu naknadno promijenite port, adresu ili uslugu.

Opcija 1: Automatizirana instalacija (Preporučeno)

Automatizirana skripta obavlja sve umjesto vas.

Korak 1: Pokrenite skriptu

```
cd ~/PhantomSDR-Plus
chmod +x install-stats-server.sh # only if it is not already executable
./install-stats-server.sh
```

VAŽNO: NEMOJTE pokretati kao root ili s sudo. Pokrenite kao vaš uobičajeni korisnik.

Korak 2: Slijedite interaktivne upite

Skripta će vas voditi kroz:

1. Instalacija Node.js-a (ako je potrebno)

```
△ Node.js / npm is not installed
Would you like to install Node.js 22 via nvm now? (y/n): y
```

Skripta instalira Node.js 22 putem nvm-a, na bilo kojoj Linux distribuciji — bez root ovlasti i bez apt izvora. Isto pitanje postavlja i kada je Node.js prisutan, ali stariji od 22.

2. Direktorij instalacije

```
Installation directory [/home/user/sdr-stats-server]:
```

Pritisnite Enter za zadanu vrijednost ili navedite prilagođenu putanju.

3. Konfiguracija porta (Poboljšano!)

```
Port Configuration:
  ▫ Default port: 3001
  ▫ Common alternatives: 8080, 5000, 8888
Enter port number [3001]:
```

Skripta provjerava vaš port:

- Provjerava je li valjan broj
- Upozorava ako je port < 1024 (zahtijeva root ovlasti)
- Provjerava raspon (1-65535)
- Provjerava je li port već u upotrebi
- Prikazuje koji ga proces koristi ako je zauzet
- Omogućuje vam odabir drugog porta

Primjeri scenarija:

Korištenje zadanog porta:

```
Enter port number [3001]: ↵
✓ Port 3001 selected
```

Odabir prilagođenog porta:

```
Enter port number [3001]: 8080
✓ Port 8080 selected
```

Port već u upotrebi:

```

Enter port number [3001]: 3001
△ Port 3001 is currently in use
Process using port 3001:
node    12345 user    20u  IPv6 123456  TCP *:3001 (LISTEN)
Choose a different port? (y/n): y
Enter port number [3001]: 3002
✓ Port 3002 selected

```

Nevaljan port:

```

Enter port number [3001]: abc
x Port must be a number
Enter port number [3001]: 99999
x Port must be between 1 and 65535
Enter port number [3001]: 3001
✓ Port 3001 selected

```

4. Adresa poslužitelja

```

What is your server's public address?
Examples: Your_site_IP, 192.168.1.100, localhost
Server address: Your_site_IP

```

Unesite javnu domenu ili IP adresu vašeg poslužitelja.

5. Potvrda

```

Please confirm your settings:
  Installation directory: /home/user/sdr-stats-server
  Port: 3001
  Server address: Your_site_IP
  Stats URL will be: http://Your_site_IP:3001
Continue with these settings? (y/n): y

```

6. Postavljanje Systemd usluge (Opcionalno)

```

Would you like to set up the server as a system service (auto-start on boot)? (y/n): y

```

Preporučeno: Odaberite **y** kako bi se poslužitelj automatski pokretao pri podizanju sustava.

Korak 3: Instalacija dovršena! ✓

Skripta će prikazati sažetak:

```

Installation Complete! ✓
Open the port in the router!

```

```

Configuration Summary:
Installation: /home/user/sdr-stats-server
Port: 3001
Server: Your_site_IP
API URL: http://Your_site_IP:3001/api/system-stats

```

Konfigurirajte svoju PhantomSDR aplikaciju

Nakon što skripta završi, morate ažurirati svoju Svelte aplikaciju.

Korak 1: Ažurirajte `site_information.json`

Budući da je `site_information.json` već uređen tijekom početnog postavljanja, trebate samo dodati novi redak `siteStats` s vašim portom:

```
{
  "siteSysop": "your name or callsign",
  "siteSysopEmailAddress": "mail@mail.net",
  "siteGridSquare": "QTH locator",
  "siteCity": "City Country",
  "siteInformation": "https://github.com/svlbtl/PhantomSDR-Plus",
  "siteHardware": "Hardware you are using, ",
  "siteSoftware": "Software you are using",
  "siteReceiver": "Receiver model",
  "siteAntenna": "Receiving Antenna.",
  "siteNote": "This is a bright new open-source WebSDR project, under active development.",
  "siteIP": "http://Your_site_IP:port",
  "siteStats": "http://Your_site_IP:3001", ← DODAJTE OVAJ REDAK (koristite svoj port)
  "siteSDRBaseFrequency": 0,
  "siteSDRBandwidth": 30000000,
  "siteRegion": 1,
  "siteChatEnabled": true
}
```

Napomena: Koristite port koji ste odabrali tijekom instalacije!

Korak 2: Ažurirajte `App.svelte` (najnovija GitHub verzija već uključuje ažuriranu datoteku)

Napravite ove **4 promjene** u svojoj `App.svelte` datoteci:

Promjena 1: Osigurajte da je `siteStats` deklariran (oko retka 50)

```
import {
  siteSysop,
  siteSysopEmailAddress,
  siteInformation,
  siteGridSquare,
  siteCity,
  siteHardware,
  siteSoftware,
  siteReceiver,
  siteAntenna,
  siteNote,
  siteIP,
  siteStats, // ← DODAJTE OVAJ REDAK
  siteSDRBaseFrequency,
  siteSDRBandwidth,
  siteRegion,
  siteChatEnabled,
} from "../site_information.json";
```

Promjena 2: Ažurirajte objekt systemStats (oko retka 530)

```
let systemStats = {
  cpu: { usage: 0, cores: 0, temperature: null, frequency: null, topProcesses: [] }, // ← DODAJTE
  topProcesses: []
  memory: { used: 0, total: 0, percent: 0 }
};
```

Promjena 3: Ažurirajte fetch URL (oko retka 541)

```
async function fetchSystemStats() {
  try {
    const response = await fetch(`${siteStats}/api/system-stats`); // ← PROMIJENITE OVAJ REDAK
    if (response.ok) {
      systemStats = await response.json();
    } else {
      console.error('Failed to fetch system stats:', response.statusText);
    }
  } catch (error) {
    console.error('Error fetching system stats:', error);
  }
}
```

Promjena 4: Dodajte prikaz vodećih procesa (u odjeljku CPU, oko retka 3838)

Dodajte ovaj kod nakon odjeljka o temperaturi:

```
{#if systemStats.cpu.frequency}
<div style="display: flex; justify-content: space-between; margin-bottom: 0.25rem;">
  <span>Frequency:</span>
  <span style="color: #4ade80;">{systemStats.cpu.frequency.current} GHz{#if
systemStats.cpu.frequency.max} <span style="opacity: 0.7;">(max {systemStats.cpu.frequency.max})</
span>{/if}</span>
</div>
{/if}

{#if systemStats.cpu.temperature !== null}
<div style="display: flex; justify-content: space-between; margin-bottom: 0.25rem;">
  <span>Temperature:</span>
  <span style="color: {systemStats.cpu.temperature > 70 ? '#fbbf24' :
'#4ade80'};">{systemStats.cpu.temperature}°C</span>
</div>
{/if}

<!-- DODAJTE CIJELI OVAJ ODJELJAK: -->
{#if systemStats.cpu.topProcesses && systemStats.cpu.topProcesses.length > 0}
<div style="margin-top: 0.75rem; padding-top: 0.75rem; border-top: 1px solid
rgba(255,255,255,0.1);">
  <h4 style="margin: 0 0 0.5rem 0; font-size: 0.85rem; color: rgba(0, 225, 255, 0.8);">Top
Processes:</h4>
  {#each systemStats.cpu.topProcesses as process}
    <div style="display: flex; justify-content: space-between; margin-bottom: 0.25rem; font-size:
0.85rem;">
      <span style="overflow: hidden; text-overflow: ellipsis; white-space: nowrap; max-width:
70%;">{process.name}</span>
      <span style="color: {process.cpu > 50 ? '#ef4444' : process.cpu > 25 ? '#fbbf24' :
'#4ade80'};">{process.cpu}%</span>
    </div>
  {/each}
</div>
{/if}
```

Korak 3: Ponovno kompilirajte svoju aplikaciju

```
cd /path/to/your/phantomsdr-app  
npm ./recompile.sh
```

Testiranje instalacije

Test 1: Provjerite API krajnju točku

```
curl http://localhost:3001/api/system-stats
```

Očekivani izlaz (JSON):

```
{  
  "cpu": {  
    "usage": 45.2,  
    "cores": 8,  
    "coresUsed": 1.3,  
    "temperature": 62.5,  
    "frequency": {  
      "current": 3.0,  
      "max": 3.9,  
      "limit": 4.4  
    },  
    "topProcesses": [  
      {"name": "node", "cpu": 12.5},  
      {"name": "phantomsdr", "cpu": 8.3}  
    ]  
  },  
  "memory": {  
    "used": 8.5,  
    "total": 16,  
    "percent": 53  
  }  
}
```

Test 2: Provjerite status usluge

Ako ste je instalirali kao uslugu:

```
sudo systemctl status sdr-stats.service
```

Trebalo bi prikazati: Active: active (running)

Test 3: Testirajte u pregledniku

1. Otvorite svoje PhantomSDR web sučelje
2. Pomaknite se do odjeljka **Additional Info**
3. Kliknite na gumb **Open Additional Info**
4. Potražite gumb **Stats** pored informacija o računalu
5. Kliknite na njega da biste otvorili modalni prozor statistike
6. Provjerite prikazuje li se i ažurira li se statistika

Upravljanje uslugom

Ako ste je instalirali kao systemd uslugu, koristite ove naredbe:

```
# Start the service
sudo systemctl start sdr-stats.service

# Stop the service
sudo systemctl stop sdr-stats.service

# Restart the service (after updating files)
sudo systemctl restart sdr-stats.service

# Check status
sudo systemctl status sdr-stats.service

# View real-time logs
sudo journalctl -u sdr-stats.service -f

# View last 50 log entries
sudo journalctl -u sdr-stats.service -n 50

# Enable auto-start on boot
sudo systemctl enable sdr-stats.service

# Disable auto-start
sudo systemctl disable sdr-stats.service

# Edit service
sudo nano /etc/systemd/system/sdr-stats.service

# Delete service
sudo rm /etc/systemd/system/sdr-stats.service
```

Ručna instalacija (Opcija 2)

VAŽNO

Za normalnu instalaciju ovaj odjeljak vam ne treba. `install-stats-server.sh` sve to obavlja umjesto vas i glavna ga instalacijska skripta pokreće prema zadanome. Slijedite ga samo za ručno postavljanje ili za ručni popravak pojedinog dijela.

⚠ **Upozorenje:** Ručna instalacija može koristiti zastarjele datoteke. **Automatizirana skripta je izričito preporučena** jer:

- Instalira najnoviju verziju s poboljšanim otkrivanjem temperature
- Automatski instalira i konfigurira lm-sensors
- Automatski upravlja svim ovisnostima

Ako ipak preferirate ručnu instalaciju:

1. Instalirajte Node.js

```
# Any distribution - nvm needs no root and no apt source.
# NOT NodeSource: deb.nodesource.com now answers HTTP 403 on every path.
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.4/install.sh | bash
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
nvm install 22 && nvm use 22 && nvm alias default 22
```

2. Instalirajte lm-sensors (Potrebno za Intel/AMD temperaturu)

```
sudo apt-get update
sudo apt-get install -y lm-sensors
sudo sensors-detect --auto
```

3. Stvorite direktorij

```
mkdir ~/sdr-stats-server
cd ~/sdr-stats-server
```

4. Preuzmite najnovije datoteke

Važno: datoteke poslužitelja više se ne čuvaju kao kopije u repozitoriju - stvara ih instalacijska skripta.

Umjesto toga, jednom pokrenite automatiziranu skriptu za generiranje najnovijih datoteka, a zatim ih kopirajte:

```
# Run the automated script from the repository root
cd ~/PhantomSDR-Plus
./install-stats-server.sh
# When prompted, use a temporary port like 9999
# After installation completes, copy the generated files
cp ~/sdr-stats-server/system-stats-server.js ~/your-manual-install-dir/
cp ~/sdr-stats-server/package.json ~/your-manual-install-dir/
```

ILI ručno stvorite datoteke koristeći najnoviji kod iz instalacijske skripte.

5. Uredite port (ako je potrebno)

```
nano ~/sdr-stats-server/system-stats-server.js
```

Promijenite redak 6:

```
const PORT = process.env.PORT || 3001; // Change this number
```

6. Instalirajte ovisnosti

```
npm install
```

7. Probno pokretanje

```
npm start
```

8. Postavite uslugu (Opcionalno)

Slijedite postavljanje systemd usluge iz odjeljka o automatiziranoj instalaciji.

Promjena porta nakon instalacije

Ako trebate promijeniti port nakon instalacije:

Korak 1: Uredite datoteku poslužitelja

```
nano ~/sdr-stats-server/system-stats-server.js
```

Promijenite redak 6:

```
const PORT = process.env.PORT || 3001; // Change this number
```

Korak 2: Ažurirajte site_information.json

```
"siteStats": "http://Your_site_IP:NEW_PORT"
```

Korak 3: Ponovno pokrenite uslugu

```
sudo systemctl restart sdr-stats.service
```

Korak 4: Ponovno izgradite svoju Svelte aplikaciju

```
cd /path/to/phantomsdr-app  
npm run build
```

Rješavanje problema

Problem: Port je već u upotrebi

Pogreška:

```
Error: listen EADDRINUSE: address already in use :::3001
```

Rješenje:

1. Pronađite što koristi port:

```
sudo lsof -i :3001
```

1. Zaustavite taj proces:

```
kill <PID>  
# or  
sudo systemctl stop sdr-stats.service
```

1. Ponovno pokrenite uslugu:

```
sudo systemctl start sdr-stats.service
```

Alternativa: Ponovno pokrenite instalacijsku skriptu i odaberite drugi port kada se od vas zatraži.

Problem: Usluga se ne pokreće

Pogreška:

```
status=217/USER
```

Rješenje:

1. Provjerite datoteku usluge:

```
cat /etc/systemd/system/sdr-stats.service
```

1. Provjerite podudara li se redak `User=` s vašim korisničkim imenom:

```
whoami
```

1. Uredite ako je potrebno:

```
sudo nano /etc/systemd/system/sdr-stats.service
```

1. Ponovno učitajte i ponovno pokrenite:

```
sudo systemctl daemon-reload
sudo systemctl restart sdr-stats.service
```

Problem: Statistika se ne prikazuje u pregledniku

Simptomi: Modalni prozor se otvara, ali nema podataka ili prikazuje vrijednosti 0

Rješenja:

1. Provjerite konzolu preglednika (F12) za pogreške
2. Provjerite je li API dostupan:

```
curl http://localhost:3001/api/system-stats
```

1. Provjerite je li usluga pokrenuta:

```
sudo systemctl status sdr-stats.service
```

1. Provjerite podudara li se `siteStats` u `site_information.json` s adresom i portom vašeg poslužitelja
2. Provjerite CORS pogreške - osigurajte da poslužitelj dopušta vašu domenu

Problem: Temperatura prikazuje null

Uzrok: Očitavanje temperature nije dostupno na vašem sustavu

Rješenja:

1. **Za većinu Linux sustava:** Provjerite postoji li toplinska zona:

```
cat /sys/class/thermal/thermal_zone0/temp
```

1. Instalirajte lm-sensors:

```
sudo apt-get install lm-sensors
sudo sensors-detect
sensors
```

1. **Napomena:** Temperatura možda neće biti dostupna na:
2. Virtualnim strojevima
3. Nekim VPS pružateljima usluga
4. Windows Subsystem for Linux (WSL)
5. Ne-Linux sustavima

To je normalno - ostatak statistike i dalje će raditi!

Problem: Vrijednosti statistike ne podudaraju se s `top`

Uzrok: Razdoblje uzorkovanja procesora bilo je prekratko u starijim verzijama

Rješenje: Trenutna verzija koristi uzorkovanje od 1 sekunde za točna očitavanja. Ako imate staru instalaciju:

1. Ažurirajte datoteku poslužitelja:

```
nano ~/sdr-stats-server/system-stats-server.js
```

1. Pronađite redak 18 i promijenite:

```
}, 100); // Old value
```

u:

```
}, 1000); // New value (1 second)
```

1. Ponovno pokrenite:

```
sudo systemctl restart sdr-stats.service
```

Ili jednostavno ponovno pokrenite instalacijsku skriptu da biste dobili najnoviju verziju.

Problem: Node.js nije pronađen

Pogreška:

```
node: command not found
```

Rješenje:

1. Instalirajte Node.js:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.4/install.sh | bash
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
nvm install 22 && nvm use 22 && nvm alias default 22
```

1. Provjerite:

```
node --version
npm --version
```

Što ćete vidjeti

Nakon instalacije i konfiguracije, modalni prozor statistike prikazuje:

Odjeljak CPU

- **Korištenje:** Ukupni postotak iskorištenosti procesora
- **Jezgre:** Koliko se procesora doista koristi, izraženo u cijelim jezgrama — npr. 1.3 / 12 in use znači da stroj obavlja posao 1,3 jezgre. Podatak je točan i ne treba mu nikakav prag. Namjerno nije broj zauzetih jezgri: raspoređivač posao jedne jezgre razmazuje po više njih, pa bi takav broj ovu vrijednost mogao samo približno pogoditi.
- **Frekvencija:** Prosječna trenutna frekvencija procesora na svim jezgrama u GHz, s najbržom jezgrom u zagradi (ako je dostupna)
- **Temperatura:** Temperatura procesora u Celzijevim stupnjevima (ako je dostupna), označena bojom: zelena do 69 °C, narančasta 70-79 °C, crvena od 80 °C naviše
- **Vodeći procesi:** 5 procesa koji najviše troše procesor s postotcima označenim bojama:
 - Zelena: do 50 % procesora
 - Narančasta: 51-80 % procesora
 - Crvena: iznad 80 % procesora

Odjeljak memorije

- **Iskorišteno:** Memorija u upotrebi (GB)
- **Ukupno:** Ukupna memorija sustava (GB)
- **Korištenje:** Postotak korištenja memorije (označen bojama)

⚙️Automatsko ažuriranje

- Statistika se osvježava svakih **5 sekundi** dok je modalni prozor otvoren
- Prestaje se ažurirati kada zatvorite modalni prozor (štedi resurse)

Otkrivanje temperature

Poslužitelj statistike koristi napredno otkrivanje temperature s više metoda:

Podržani sustavi:

- **Intel procesori:** Čita s coretemp senzora putem lm-sensors
- **AMD procesori:** Čita s k10temp senzora (Tdie/Tctl)
- **ARM procesori:** Čita iz toplinskih zona (Raspberry Pi, itd.)

Zahtjevi:

- **Intel/AMD:** Zahtijeva paket `lm-sensors` (skripta ga automatski instalira)
- **ARM/Raspberry Pi:** Radi odmah, nisu potrebni dodatni paketi

Prioritet instalacije:

1. Pokušava naredbu `sensors` za temperaturu Package/Core (Intel/AMD)
2. Vraća se na izravno čitanje `coretemp` (Intel)
3. Vraća se na `thermal_zone0` (ARM/Raspberry Pi)

Ako temperatura prikazuje `null`, instalirajte `lm-sensors`:

```
sudo apt-get install lm-sensors
sudo sensors-detect --auto
sudo systemctl restart sdr-stats.service
```

Sažetak značajki

Nadzor u stvarnom vremenu - Statistika sustava uživo s vašeg poslužitelja **Metrike procesora** - Korištenje, jezgre, temperatura, vodeći procesi **Praćenje memorije** - Iskorišteno, ukupno i postotak
Automatska ažuriranja - Osvježava se svakih 5 sekundi **Označeno bojama** - Vizualni pokazatelji za upozorenja **Lagano** - Minimalno korištenje resursa **Jednostavna instalacija** - Automatizirana skripta obavlja sve **Provjera porta** - Pametan odabir porta s otkrivanjem sukoba
Upravljanje uslugom - Automatsko pokretanje pri podizanju sustava

Sigurnosne napomene

- Poslužitelj statistike prihvaća veze s bilo kojeg izvora (CORS: *)
- Za produkciju razmislite o ograničavanju CORS-a samo na vašu domenu
- Odabir porta provjerava unos i provjerava sukobe
- Očitavanje temperature procesora je samo za čitanje i sigurno je
- Nikakve osjetljive informacije o sustavu nisu izložene

Podrška i resursi

Uključene datoteke:

U ovoj mapi (sama instalacijska skripta nalazi se u korijenu, kao `PhantomSDR-Plus/install-stats-server.sh` - postoji samo jedna kopija):

- `package.json` - Ovisnosti, identične onima koje skripta zapisuje
- Ovaj dokument, dostupan i na de, el, es, fr, hr i ru

Skripta ih stvara u `~/sdr-stats-server/`:

- `system-stats-server.js` - Backend poslužitelj, generiran s odabranim portom
- `package.json` - Ovisnosti
- `node_modules/` - instalira `npm install`

Dobivanje pomoći:

- Provjerite odjeljak za rješavanje problema iznad
- Pregledajte zapisnike usluge: `sudo journalctl -u sdr-stats.service -n 50`
- Ručno testirajte API: `curl http://localhost:3001/api/system-stats`
- Provjerite konfiguracijske datoteke

Deinstalacija poslužitelja statistike:

- Izbrišite mapu `/sdr-stats-server`
- Ako ste stvorili uslugu, samo je izbrišite koristeći:

```
sudo rm /etc/systemd/system/sdr-stats.service`
```

- Izmijenite `App.svelte`: Pronađite dio koda:

```

        <!-- In case you don't want the Stats Button to appear, please comment this
button section (12 lines)-->
        <!-- System Stats Button -->
        <button
            type="button"
            class="glass-button text-white py-1 px-2 ml-2 rounded text-xs"
            on:click={openSystemStats}
            title="System Resources"
            aria-haspopup="dialog"
            aria-expanded={showSystemStats}
            aria-controls="system-stats-dialog"
            style="color:rgba(0, 225, 255, 0.993); font-size: 0.75rem;"
        >
            Stats
        </button>
    ...
    i zamijenite s:

```

```

        <!-- In case you don't want the Stats Button to appear, please comment this button
section (12 lines)-->
        <!-- System Stats Button -->
        <!--
        <button
            type="button"
            class="glass-button text-white py-1 px-2 ml-2 rounded text-xs"
            on:click={openSystemStats}
            title="System Resources"
            aria-haspopup="dialog"
            aria-expanded={showSystemStats}
            aria-controls="system-stats-dialog"
            style="color:rgba(0, 225, 255, 0.993); font-size: 0.75rem;"
        >
            Stats
        </button>
        -->
    ...

```

- Ponovno kompilirajte pomoću naredbe:

```

    cd PhantomSDR-Plus
    ./recompile.sh
    \\\

- Ponovno pokrenite računalno i pokrenite poslužitelj

---

**Autor:** Kreirano za PhantomSDR-Plus
**Verzija:** 1.1 (Poboljšano otkrivanje temperature procesora za Intel/AMD) **Ažurirano:** Siječanj
2026.
**Licenca:** MIT

---

## Brza referenca naredbi

```bash
Installation
cd ~/PhantomSDR-Plus
chmod +x install-stats-server.sh
./install-stats-server.sh

Service Management
sudo systemctl start sdr-stats.service # Start
sudo systemctl stop sdr-stats.service # Stop
sudo systemctl restart sdr-stats.service # Restart
sudo systemctl status sdr-stats.service # Status
sudo journalctl -u sdr-stats.service -f # Logs

Testing
curl http://localhost:3001/api/system-stats # Test API
curl http://localhost:3001/api/health # Health check

Troubleshooting
sudo lsof -i :3001 # Check port
whoami # Check username
node --version # Check Node.js

```

Sretno praćenje!

# Dekoderi

DECODERS.md

## PhantomSDR-Plus — Korisnički priručnik za dekodere

Ovaj vodič obuhvaća sve ugrađene dekodere dostupne u PhantomSDR-Plusu. Svi se dekoderi uključuju na isti način, opisan u nastavku, nakon čega slijede upute za postavljanje svakog pojedinog dekodera.

### Sadržaj

1. Kako pokrenuti dekodер
2. FT8
3. FT4-FT2
4. JS8
5. CW — Morseov kod
6. QRSS Grabber
7. WSPR
8. HF FAX / WEFAX
9. NAVTEX
10. FSK / RTTY — uključujući PSK31 i Oliviju
11. SSTV
12. Opći savjeti

### 1. Kako pokrenuti dekodер

Svim se dekoderima pristupa iz odjeljka **Decoder Options**, smještenog ispod kontrola audiospektrograma na glavnoj ploči.

#### Koraci:

1. Kliknite gumb **Decoder: OFF** da biste ga prebacili u **Decoder: ON** (kad je aktivan, postaje plav).
2. Otvorite padajući izbornik koji se pojavljuje desno od gumba i odaberite željeni dekodер.
3. Ploča odabranog dekodera pojaviti će se ispod kontrola — slijedite upute za taj dekodер u odgovarajućem odjeljku ovog vodiča.
4. Da biste zaustavili dekodiranje, odaberite — **Select decoder** — iz padajućeg izbornika ili kliknite gumb **Decoder: ON** da ga vratite na OFF.

#### Gumbi dekodera na jedan pritisak

Padajući izbornik nije jedini put. Glavna ploča ima red gumba **Decoders** — odmah ispod **Wheel Tuning Steps** — s po jednim gumbom za svaki dekodер:

Gumb	Dekoder	Gumb	Dekoder
FT8	FT8	SSTV	SSTV
FT4	FT4	NAVTEX	NAVTEX
FT2	FT2	RTTY	FSK / RTTY
CW	CW		
WSPR	WSPR	FAX	HF FAX / WEFAX

Pritisak na gumb odrađuje cijeli slijed odjednom: odabire dekoder, uključuje dekoder (ON) i dovodi njegov prozor na zaslon. Gumb je plav dok njegov dekoder radi. **Ponovni pritisak na isti gumb isključuje dekoder** — njegov se prozor zatvara s njim.

**RADEL** i **RADEU** namjerno nisu u ovom redu. To su načini digitalnog glasa, a ne tekstualni dekoderi, pa imaju vlastiti par gumba uz naslov **Modes selector** te u skočnim prozorima **Modes** i **Bands**. Ponašaju se posve jednako — pritisak za pokretanje, ponovni pritisak za zaustavljanje. Pogledajte [priručnik za RADE](#).

Gumbi, padajući izbornik i gumb ON/OFF upravljaju istim stanjem, pa što god upotrijebili, ostalo ga slijedi.

Istodobno može biti aktivan samo jedan dekoder. Prelazak na drugi dekoder automatski zaustavlja prethodni.

**QRSS grabber** nije dio ovog izbornika — ima vlastiti odjeljak **QRSS** i može raditi istodobno s nekim dekoderom.

**Dekodiranje se odvija u pozadini.** SSTV, HF FAX, NAVTEX, FSK/RTTY i CW rade svaki u zasebnoj dretvi Web Workera, pa se dekodiranje nikada ne natječe s reprodukcijom zvuka ni sa slapom. Pokretanje ili zaustavljanje dekodera ne prekida zvuk, a sučelje ostaje odzivno dok se prima slika ili stranica. Svi dekoderi dobivaju sirovi zvuk uzet *prije* AGC-a, smanjenja šuma i utišavanja — utišavanje prijamnika ili prilagodba tih postavki vlastitom uhu stoga ne utječe na dekodiranje.

**Dekoder koji radi drži način rada i propusni pojas.** Uobičajeno način rada slijedi plan opsega iz `bands-config.js`: pomaknite ugađanje u odsječak označen kao LSB ili AM i prijamnik se prebaci na njega. Dok dekoder radi, toga više nema. Dekoder zadržava način rada koji mu treba (USB kod većine, vlastiti kod RADE-a) i propusni pojas koji mu treba — PSK31 oko  $\pm 100$  Hz, Olivia svoju punu širinu, RTTY svoj shift — i oboje preživljava preugađanje, uključujući skok na drugi opseg. Bez toga bi FT8 na 40 m prebacio prijamnik na LSB čim se ugađanje pomakne, a uski propusni pojasi dekodera opet bi se raširili na puni SSB filter.

Vlastiti način rada opsega vraća se čim isključite dekoder. Način rada uvijek možete nametnuti ručno: gumbi načina rada svjestan su izbor i uvijek pobjeđuju. CW dekoder je iznimka od svega ovoga — dekodira u načinu rada u kojem slušate i nikada ne preuzima prijamnik.

## Decoder ID — automatsko prepoznavanje načina rada

Iznad gumba dekodera nalazi se okvir **Decoder ID**. Odgovara na pitanje „*što ovo slušam?*“ kada naidete na digitalni signal koji ne možete prepoznati, a zatim vam jednim klikom nudi odgovarajući dekoder.

Prema zadanim postavkama je **isključen** i dok je isključen ne troši ništa. Pritisnite **On** i ostavite signal ugođenim. Otprilike pola minute prikazuje *Listening...* s trakom napretka — mjerenja vremena moraju vidjeti dva potpuna FT8 ciklusa da bi imala smisla — a zatim imenuje način rada:

Decoder ID [On] NAVTEX / SITOR-B 99% 170 Hz shift · 100.00 Bd · 43 dB S/N [ Use NAVTEX ]

- **Način rada** i vrijednost **pouzdanosti**. Zeleno iznad 75 %, žuto iznad 50 %, narančasto ispod — narančasti odgovor shvatite kao naznaku, ne kao presudu.
- **Što je zapravo izmjereno**: zauzeta širina pojasa, razmak tonova, brzina simbola, trajanje rafala, UTC mreža kojoj odgovara, odnos signal/šum. To je dokaz iza presude, kako biste mogli i sami prosuditi.
- **Use** pokreće ispravan dekodler, točno kao da ste pritisnuli njegov gumb — i namješta prijamnik za rad tim načinom. Za načine koji žive u FSK ploči dodatno odabire ispravnu varijantu — RTTY, Weather RTTY, PSK31 ili Olivia.
- Sljedeći kandidati navedeni su iza **or**. Klikom isprobajte njega, pa vas pogrešan prvi odgovor ništa ne stoji.
- \*\*\*\* znači da je frekvencija na koju ste ugođeni poznata pozivna frekvencija za taj način rada i da je to utjecalo na poredak.

Nikada sam ne mijenja vaš dekodler. Uvijek samo predlaže. A čim pokrenete dekodler — s reda gumba, s padajućeg izbornika ili s vlastitog gumba **Use** unutar Decoder ID-a — Decoder ID se sam isključuje: njegov je posao bio odabrati jedan, a dekodler sada traži isti zvuk.

**Namješta prijamnik za taj način rada.** Odabir načina rada odluka je da se njime i radi, pa prijamnik slijedi: ugođena frekvencija dolazi na sredinu slapa, a pogled se postavlja na oko 100 kHz oko nje, bočni pojas postaje onaj na kojem se taj način radi, a propusni pojas se sužava na prozor u kojem taj način živi — cijeli podpojas od 3 kHz za FT8, FT4, FT2 i JS8, 1350–1650 Hz za WSPR, 250–750 Hz za NAVTEX, 800–2700 Hz za HF FAX, 900–2600 Hz za SSTV te  $\pm 250$  Hz s obje strane ugođene frekvencije za CW, pri čemu prijamnik dodatno prelazi na CW. Uski propusni pojas drži susjedne signale izvan dekodera — upravo ono što biste učinili rukom. Dvije iznimke: SSTV zadržava bočni pojas koji je sam odabrao — po dogovoru LSB na 80 i 40 m, gdje bi nametnuti USB izokrenuo raspored tonova — a FSK načini zadržavaju prozor koji njihova vlastita ploča izvodi iz razmaka ili širine pojasa varijante, uži od bilo koje stalne vrijednosti. Pritisak na **Use** za dekodler koji već radi i dalje ga isključuje, a prijamnik tada ostaje točno onakav kakav je bio.

**Spori načini trebaju više vremena.** WSPR leži na dvominutnoj mreži, a SSTV slike traju minutu ili dvije, pa se ta dva imenuju kasnije od ostalih — dajte im tri do četiri minute na frekvenciji. HF FAX i SSTV usto se prepoznaju samo ako su ugođeni na uobičajen način — slikovni pojas leži na otprilike 1500 do 2300 Hz u audiju, ondje gdje ga očekuju i njihovi dekodleri. Promjena frekvencije pokreće mjerenje ispočetka, jer sve prikupljeno do tada opisuje frekvenciju koju ste upravo napustili.

**Što prepoznaje:** FT8, FT4, JS8, WSPR, CW, NAVTEX/SITOR-B, RTTY na 45,45 Bd, Weather RTTY (DWD), PSK31, Olivia, HF FAX, SSTV i DSC. U ovoj inačici nema DSC dekodera, pa se DSC imenuje i označava s *no decoder*.

**Recentre & retry.** Gdje se signal nalazi unutar audio propusnog pojasa nije važno između otprilike 400 Hz i 2700 Hz. Izvan tog raspona mjerenje se pogoršava i pojavljuje se žuti gumb **Recentre & retry**. Pritiskom se ugađanje pomiče tako da signal padne u sredinu propusnog pojasa i započinje novo mjerenje. To je jedan izračunati pomak, a ne pretraživanje.

### Tri poštena ograničenja:

- **FT8, JS8 i FT2 su isti signal.** Ista 8-FSK modulacija, isti razmak tonova, a JS8 pri brzini Normal koristi FT8-ov ciklus od 15 sekundi. Ništa u samom signalu ih ne razdvaja. Frekvencija ih razdvaja: na FT8 pozivnoj frekvenciji javlja FT8, na JS8 frekvenciji javlja JS8, a bilo gdje drugdje pošteno javlja obitelj — *FT8 / JS8 / FT2*. Pokrenite bilo koji od ta dva dekodera i vidite koji daje tekst.

- **Radije šuti nego da nagađa.** Ispod otprilike 10 dB odnosa signal/šum javlja *Nothing above the noise* ili *Not recognised* umjesto da imenuje način rada koji zapravo ne vidi. To je namjerno: samouvjeren pogrešan odgovor gori je od nikakvog.
- **Signal koji slabi čita se drukčije.** WSPR odašilje 110 sekundi bez prekida, dulje nego što traje uobičajeni QSB propad, pa duboko slabljenje razreže emisiju na komade i njezin dvominutni ritam iz njih se više ne može očitati. Tada se dolazi do njega drugim putem: cijela se povijest slušanja uspoređuje s dvominutnom mrežom umjesto s pojedinačnim emisijama, a redak s mjerenjima piše *120 s grid (through fading)*. To je slabiji dokaz od emisije čute u cijelosti, pa je prikazana pouzdanost namjerno niža. Slabljenje dovoljno duboko da signal zakopa u šum ne da se vratiti uopće. Olivia je stroža od ostalih: neće biti imenovana dok joj se ne izmjeri simbolska brzina, pa zašuti ranije.

## 2. FT8

**Što je to:** FT8 je popularan digitalni način rada za slabe signale koji radioamateri koriste diljem svijeta. Emisije traju točno 15 sekundi, a protokol može primiti signale do 20–25 dB ispod razine šuma. To je najčešće korišten način rada za veze na velike udaljenosti (DX).

### Preporučene frekvencije (USB)

Pojas	Frekvencija
160 m	1.840 MHz
80 m	3.573 MHz
40 m	7.074 MHz
30 m	10.136 MHz
20 m	14.074 MHz
17 m	18.100 MHz
15 m	21.074 MHz
12 m	24.915 MHz
10 m	28.074 MHz

### Postavljanje

1. Ugodite se na jednu od gore navedenih FT8 frekvencija i postavite način rada na **USB**.
2. Uključite dekodeer i iz padajućeg izbornika odaberite **FT8**. Ili jednostavno pritisnite gumb **FT8**.
3. Ploča **FT8 Messages** automatski se pojavljuje ispod.

### Čitanje ispisa

Popis poruka prikazuje dekodirane emisije kako pristižu. Svaki ciklus od 15 sekundi donosi novu skupinu poruka. Polje **Farthest** (gore desno na ploči) pokazuje najveću udaljenost dekodiranu u trenutačnoj sesiji, u kilometrima.

Tipičan oblik poruke: CQ DX AA1BB FN31 — CQ poziv pozivnog znaka AA1BB smještenog u polju FN31.

**SNR** prikazan uz svako dekodiranje pravi je omjer signala i šuma, izmjeren na dekodiranom signalu u odnosu na šum oko njega i sveden na širinu pojasa od 2500 Hz, kako ga daje WSJT-X. Slabe stanice pokazuju negativne vrijednosti, do oko –20 dB, a jake pozitivne; na istim snimkama oba programa u prosjeku se slažu, obično unutar oko 3 dB za pojedino dekodiranje. FT4 i FT2 koriste istu metodu.

**Nakon ažuriranja:** FT8/FT4/FT2 dekodera zasebna je datoteka koju preglednici dugo čuvaju. Ako su svi SNR i dalje između oko 0 i +5 dB, ponovno učitajte stranicu potpunim osvježavanjem (**Ctrl+Shift+R**, ili **Cmd+Shift+R** na Macu).

**Savjet:** FT8 je usko vremenski sinkroniziran. Vaš preglednik koristi sat vašeg računala; ako sat sustava odstupa više od nekoliko sekundi, dekodiranje neće uspjeti. Držite vrijeme sustava usklađeno putem NTP-a.

### 3. FT4-FT2

**Što je to:** FT4 je brža inačica FT8-a namijenjena radu u natjecanjima. Svaki ciklus emitiranja traje 7,5 sekundi (polovica FT8-a), zbog čega je dvostruko brži, ali traži nešto jači signal. FT2 je još brža inačica FT8-a. Riječ je o iznimno brzom 77-bitnom načinu rada s TR razdobljima od 3,75 sekundi (= polovica FT4) — još je eksperimentalan.

#### Preporučene frekvencije (USB)

Pojas	Frekvencija FT4
80 m	3.575 MHz
40 m	7.047 MHz
30 m	10.140 MHz
20 m	14.080 MHz
15 m	21.140 MHz
10 m	28.180 MHz

Pojas	Frekvencija FT2
160 m	1.843 do 1.846
80 m	3.578 do 3.581
60 m	5.360 (provjerite lokalne regionalne propise)
40 m	7.052 do 7.062
30 m	10.144
20 m	14.084
17 m	18.108
15 m	21.144
12 m	24.923
10 m	28.184

#### Postavljanje

1. Ugodite se na FT4 frekvenciju i postavite način rada na **USB**.
2. Uključite dekodera i iz padajućeg izbornika odaberite **FT4** ili **FT2**. Ili jednostavno pritisnite gumb **FT4** / **FT2**.
3. Ploča **FT4 Messages** ili **FT2 Messages** pojavljuje se ispod, istog rasporeda kao FT8 ploča.

**Napomena:** FT4 i FT8 koriste različite spektralne formate i nisu zamjenjivi. Pri korištenju ovog dekodera pobrinite se da ste na FT4 frekvenciji.

## 4. JS8

**Što je to:** JS8 (način rada koji koristi JS8Call) uzima FT8-ov mehanizam za slabe signale i pretvara ga u način razgovora s tipkovnice na tipkovnicu. Dok FT8 šalje fiksne poruke od 13 znakova, JS8 šalje slobodan tekst u malim dijelovima i spaja uzastopne emisije natrag u cijele rečenice — sve dulje od otprilike dvanaest znakova stiže kroz nekoliko ciklusa od 15 sekundi. Dekodira otprilike jednako duboko kao FT8, pa radi ondje gdje govor i CW više ne prolaze.

### Preporučene frekvencije (USB)

Pojas	Frekvencija
160m	1.842 MHz
80m	3.578 MHz
40m	7.078 MHz
30m	10.130 MHz
20m	14.078 MHz
17m	18.104 MHz
15m	21.078 MHz
12m	24.922 MHz
10m	28.078 MHz

### Brzine

JS8 ima pet brzina. Sve stanice u razgovoru moraju koristiti istu. **Normal** je pozivna brzina i na njoj se odvija gotovo sav promet — počnite od nje.

Brzina	Ciklus	Širina	Kada se koristi
Slow	30 s	25 Hz	Vrlo slabe putanje; najosjetljivija
Normal	15 s	50 Hz	Uobičajena pozivna brzina
Fast	10 s	80 Hz	Brža razmjena, traži jači signal
Turbo	6 s	160 Hz	Jaki lokalni signali
Ultra	4 s	250 Hz	Eksperimentalna, rijetko se viđa

### Postavljanje

1. Ugodite se na JS8 frekvenciju i postavite način rada na **USB**.
2. Uključite dekoder i odaberite **JS8** s popisa, ili jednostavno pritisnite tipku **JS8**.
3. Pojavljuje se ploča **JS8 Decoder**. Ostavite **Speed** na *Normal*, osim ako znate da stanica koju tražite koristi drugu.

### Kako čitati ispis

Ploča ima dva popisa.

**Poruke koje još pristižu** prikazuju se na vrhu u zelenoj boji, s trepćućim pokazivačem. JS8 poruka može trajati četiri ciklusa — punu minutu na Normal — pa se ovdje vidi kako se rečenica gradi. Nije greška ako ondje ostane neko vrijeme.

**Dovršene poruke** popunjavaju glavni popis, svaka u svom retku:

Stupac	Značenje
Mode	Uvijek JS8
Hz	Audio frekvencija signala unutar propusnog pojasa
dB	Odnos signal-šum u referentnoj širini od 2500 Hz
Message	Dekodirani tekst; <b>pozivni znakovi prikazani su zeleno</b>

Tipične poruke:

- SV1BTL KM17: HB — heartbeat: stanica javlja da je u eteru, sa svojim lokatorom.
- KN4CRD: K0OG SNR -05 — usmjerena poruka: KN4CRD javlja K0OG-u da ga prima na -5 dB.
- MP 100W 8M/BALUN JN58KH AUGSBURG, MARTIN — slobodan tekst, ovdje opis stanice pristigao kroz četiri ciklusa.

Redak prikazan **bljedo i kurzivom** istekao je prije nego što je stigao njegov posljednji okvir. Tekst je stvaran, ali može biti skraćen.

## Sync offset

Klizač **Sync offset** određuje koliko nakon granice UTC ciklusa počinje snimanje, čime se nadoknađuje kašnjenje audio lanca. Ostavite **Auto** uključen: dekodirani vremenski razmak signala koje čuje i sam se prilagođava.

## Oznake na spektru

Mala traka spektra na ploči crta žutu okomitu crtu na frekvenciji svakog signala dekodiranog u prethodnom ciklusu, pa se stupac Hz može očitati izravno sa zaslona.

## Vidjeti vlastite spotove

Ako prijamnik šalje JS8 spotove na PSK Reporter (vidi karticu Spot Reporting u administracijskoj ploči), gumb **JS8 map** u okviru **Note:** otvara kartu PSK Reportera filtriranu na pozivni znak ovog prijamnika i način JS8 — isto ono što **FT8 map** i **FT4 map** rade za te načine.

**Savjet:** JS8 je mnogo tiši od FT8. Poslijepodne možete vidjeti jednu emisiju svakih nekoliko minuta, a duga razdoblja bez ičega sasvim su normalna. Dajte mu pet do deset minuta prije nego što zaključite da nešto ne valja. 20m (14.078) i 40m (7.078) uobičajena su mjesta.

**Savjet:** Kao i FT8, JS8 je strogo vremenski sinkroniziran i koristi sat vašeg računala. Ako sistemsko vrijeme odstupi više od sekundu ili dvije, ništa se neće dekodirati. Držite ga sinkroniziranim putem NTP-a.

## 5. CW — Morseov kod

**Što je to:** CW dekodir osluškuje signale Morseova koda (neprekinuti val) i pretvara ih u tekst u stvarnom vremenu. Automatski prati frekvenciju signala i prilagođava se brzini odašiljanja operatera.

## Preporučene frekvencije

CW je aktivan na svim amaterskim pojasevima, obično u njihovu donjem dijelu. Uobičajena mjesta:

Pojas	Segment
40 m	7.000–7.040 MHz
20 m	14.000–14.070 MHz
15 m	21.000–21.080 MHz

## Postavljanje

1. Ugodite se na CW signal koristeći način demodulacije **CW** ili **CW-L**, ovisno o situaciji.
2. Uključite dekodeer i iz padajućeg izbornika odaberite **CW**. Ili jednostavno pritisnite gumb **CW**.
3. Ploča **CW Decoder** pojavljuje se ispod.

## Čitanje ispisa

- Zaglavlje ploče prikazuje otkrivenu frekvenciju signala u Hz (npr.  $\approx 700$  Hz) i procijenjenu brzinu odašiljanja u riječima u minuti (npr.  $\cdot 22$  WPM).
- Ako signal nije otkriven, zaglavlje prikazuje **scanning...**
- Na praznom kanalu ništa se ne ispisuje, a brzina i ton se ne prikazuju. Dekoder počinje pisati tek kad postoji jasna razlika između pritisnutog i otpuštenog tipkala, što sam šum ne proizvodi, dok signal još čitljiv na oko  $-10$  dB to ima u izobilju.
- Dekodirani tekst kliže u jantarnom fontu jednolike širine. Trepćući pokazivač (■) označava mjesto na kojem se tekst trenutačno ispisuje.
- Kliknite **Clear** da izbrišete međuspremnik ispisa.

**Savjeti:** - Centrirajte propusni pojas na CW ton. Dekoder radi najbolje kada CW signal leži otprilike između 400 i 900 Hz u audiospektru. - Čitaju se brzine od oko 8 do 40 WPM, uključujući Farnsworth odašiljanje (znakovi poslani brzo s duljim razmacima među njima): zaključavanje brzine drži, a slova se ne raspadaju u zasebne riječi. Izrazito ručno tipkan (nepravilan) Morse i dalje može smanjiti točnost. - Dekoder daje najbolje rezultate na jednom čistom signalu. Jak QRM od obližnjih signala na istom pojasu može ga zbuniti.

## 6. QRSS Grabber

**Što je to:** QRSS je CW odašiljan tolako sporo da jedna točka traje sekundama umjesto milisekundama. Pri toj brzini signal zauzima tek djelić herca, pa dovoljno uska analiza može izvući trag i 20–30 dB *ispod* razine šuma. Uhom se tu nema što pročitati — QRSS se **gleda**, ne sluša. Grabber računa vlastiti vrlo dugi FFT nad zvukom prijamnika i crta klasični grabber prikaz: frekvencija na okomitoj osi, vrijeme klizi slijeva nadesno, a najnoviji stupac je uvijek uz desni rub.

Većina onoga što ćete vidjeti dolazi od MEPT svjetionika (Manned Experimental Propagation Transmitter) — odašiljača vrlo male snage, često nekoliko stotina milivata u žicu, koji se neprekidno identificiraju vrlo sporim Morseom ili FSK uzorcima.

### Gdje gledati

QRSS svjetionici žive u uskim prozorima od 100–200 Hz pri donjem kraju svakog opsega. Ne morate ih pamtit: grabber ima popis **Band** s prozorima iz tablice, a odabir jednog ugađa na njega i priprema prijamnik za taj način rada (vidi *Otvaranje grabbera*).

Opseg	Središte prozora	Napomena
30m	10.140,00 kHz	Glavni QRSS prozor — daleko najprometniji, danju i noću
40m	7.039,90 kHz	
40m	7.000,85 kHz	Knights prozor, 7.000,8–7.000,9 kHz
20m	14.096,90 kHz	
80m	3.569,90 kHz	
80m	3.568,60 kHz	Stariji prozor
80m	3.500,85 kHz	Knights prozor, 3.500,8–3.500,9 kHz
160m	1.837,90 kHz	
160m	1.843,30 kHz	Stariji prozor
630m	476,10 kHz	
2200m	137,70 kHz	LF QRSS / DFCW
60m	5.288,55 kHz	
17m	18.105,90 kHz	
15m	21.095,90 kHz	
12m	24.925,90 kHz	
10m	28.125,70 kHz	
10m	28.000,85 kHz	Stariji prozor
10m	28.322,00 kHz	Alternativa
6m	50.294,30 kHz	

Nekoliko se opsega pojavljuje dvaput jer su doista u uporabi dvije konvencije. Moderni prozori leže 200 Hz ispod WSPR frekvencije toga opsega; stariji *Knights* prozori posve su drugdje, a na 40 i 80 m znatno niže. Ako je opseg tih na jednome, probajte drugi.

Riječ je o konvencijama, ne o propisima, a neke su regionalne — osobito se brojke za 10 m razlikuju. Uzmite popis kao polazište i slijedite lokalnu praksu.

## Otvaranje grabbera

QRSS grabber **nije** u izborniku dekodera. Ima vlastiti odjeljak **QRSS**, uz kontrole spektrograma i dekodera.

1. Kliknite **Show**. Pojavljuje se platno grabbera, a kontrole se razotvore pokraj gumba.
2. Odaberite prozor s popisa **Band** i pritisnite **Tune**. To u jednom koraku napravi sve što način rada traži: ugoditi na prozor, prebaciti prijamnik na **CW** i postaviti propusni pojas taman dovoljno širok za odsječak koji gledate. Skala tada pokazuje pravu QRSS frekvenciju, a tragovi padaju na središnju crtu prikaza.
3. Postavite **Speed** prema duljini točke svjetionika.
4. **Centre** i **Span** rade kao i prije; promjena bilo kojega preoblikuje propusni pojas, pa prikaz i prijamnik ostaju usklađeni.
5. Kliknite **Hide** za zaustavljanje. Vraća se uobičajeni način rada opsega.

I dalje možete i ručno — ugodite u **USB** oko 1 kHz ispod prozora tako da tragovi padnu blizu 800 Hz zvuka. Popis **Band** samo vam štedi računanje.

Dok je ugođen prozor s popisa, grabber drži prijamnik jednako kao dekoder: CW i njegov propusni pojas preživljavaju preugađanje, a prozor slijedi skalu, pa trag ostaje na središnjoj crti dok pretražujete opseg. Prijamnik vraća kad sakrijete grabber, kliknete gumb načina rada ili pokrenete dekoder.

## Brzine

Postavka	Duljina točke	Razlučivost	Prozor analize	Novi stupac svakih
QRSS 3	3 s	0,73 Hz	1,4 s	0,7 s
QRSS 6	6 s	0,37 Hz	2,7 s	1,4 s
QRSS 10	10 s	0,18 Hz	5,5 s	2,7 s
QRSS 30	30 s	0,09 Hz	10,9 s	5,5 s
QRSS 60	60 s	0,05 Hz	21,8 s	10,9 s

Postavka brža od svjetionika troši osjetljivost — trag ispadne tanak i šumljiv. Postavka sporija od svjetionika razmazuje uzastopne točke i crte u jednu prugu. Ploča se otvara na **QRSS 6**, razumnoj polazišnoj točki za nepoznat signal: osjetljivija je od QRSS 3, a osvježava se brže i tolerantnija je na drift od QRSS 10. Čim uočite oblik manipulacije, prijedite na brzinu koju svjetionik doista šalje — svaki korak prema bržoj postavci stoji 3 dB osjetljivosti.

## Kontrole

Kontrola	Što radi
<b>Band</b>	QRSS prozori iz gornje tablice. <b>Tune</b> ugađa na odabrani, u CW-u s prikladnim propusnim pojasom.
<b>Speed</b>	Postavlja duljinu transformacije — kompromis između frekvencijske i vremenske razlučivosti (tablica gore).
<b>Centre</b>	Audio frekvencija u sredini prikaza, 100–3000 Hz.
<b>Span</b>	Visina prikazanog isječka: 20, 50, 100 ili 200 Hz, pri pokretanju 100 Hz — širina jednog QRSS podpojasa. Uži isječak razvlači svaki trag na više piksela. Kod širih isječaka u ploču stane više binova nego što ima redaka piksela; svaki redak tada prikazuje najjači bin koji pokriva, pa se ništa ne može sakriti između redaka.
<b>Gain</b>	–10 do +40 dB. Pomiče paletu boja u odnosu na izmjerenu razinu šuma; povišite da posvijetlite slabe tragove.
<b>Color</b>	Rainbow, Green ili Grayscale.
<b>Clear</b>	Briše platno i poništava referencu šuma.

## Čitanje prikaza

Frekvencijske oznake teku uz lijevi rub, najviša frekvencija je na vrhu. Ispod platna statusni redak pokazuje odabranu brzinu, stvarno korištenu razlučivost (npr. 0.183 Hz/bin · 5.5 s window) i brzinu stupaca (npr. 2.7 s/column).

Paleta boja referencirana je na **trenutačni** šum, koji se neprekidno mjeri iz praznih dijelova svakog stupca, a ne na apsolutnu razinu. Promjene glasnoće, AGC-a, antene ili opsega zato ne traže ponovno namještanje Gaina — prikaz zadržava stalan kontrast prema šumu opsega kakav god bio. Kao i kod dekodera, zvuk se uzima prije AGC-a, smanjenja šuma i utišavanja, pa prijamnik možete utišati i i dalje gledati.

Što znače oblici:

- **Stabilan nosilac** crta ravnu vodoravnu crtu.

- **Spori Morse** crta tu liniju kao niz kratkih i dugih odsječaka — točke i crte čitaju se slijeva nadesno.
- **Zakrivljeni tragovi ili tragovi koji bježe** dolaze od nestabiliziranog oscilatora svjetionika koji se grije ili hladi. To je normalno i često upravo taj potpis pomaka omogućuje redovitom promatraču da prepozna stanicu.
- **Okomite pruge** preko cijelog isječka su atmosferski praznici ili lokalni impulsi šuma, a ne signal.

**Savjeti:** - Budite strpljivi. Pri QRSS 30 jedan stupac traje 5,5 sekundi, pa cijeli pozivni znak može trebati deset minuta ili više da prijeđe zaslone. Pustite da radi. - Ugađanje s popisa **Band** već sužava propusni pojas oko prozora. Ako ste ugodili ručno, učinite to sami: ne mijenja ono što transformacija razlučuje, ali sprječava jake susjede da upravljaju AGC-om prijamnika. - Grabber je neovisan o dekoderima — možete ga ostaviti uključenim dok dekođer radi na nečem drugom. - Širok span pri sporij brzini je najzahtjevnija kombinacija; cijela transformacija radi u pregledniku, pa na skromnijem računalu radije odaberite span od 50 Hz.

## 7. WSPR

**Što je to:** WSPR (Weak Signal Propagation Reporter, izgovara se „whisper“) način je rada svjetionika s iznimno slabim signalima koji kartira putove KV prostiranja diljem svijeta. Svaka emisija traje otprilike 110 sekundi i stane u kanal širine 200 Hz. Dekoder čeka potpuni dvominutni interval usklađen s UTC-om prije dekodiranja.

### Preporučene frekvencije (USB, prikazane)

Pojas	Prikazana frekvencija
160 m	1.836.600 MHz
80 m	3.568.600 MHz
40 m	7.038.600 MHz
30 m	10.138.700 MHz
20 m	14.095.600 MHz
17 m	18.104.600 MHz
15 m	21.094.600 MHz

### Postavljanje

1. Ugodite se na jednu od gore navedenih prikazanih WSPR frekvencija i postavite način rada na **USB**.
2. WSPR signal zauzima audiopodručje 1400–1600 Hz. Odabir **WSPR**-a obavlja ostalo umjesto vas: prijamnik prelazi na USB, a propusni pojas se sužava na 1350–1650 Hz, što pokriva cijelo područje koje dekođer pretražuje. Dodatno podešavanje nije potrebno.
3. Uključite dekođer i iz padajućeg izbornika odaberite **WSPR**. Ili jednostavno pritisnite gumb **WSPR**.
4. Ploča **WSPR-2 Decoder** pojavljuje se ispod.

### Čitanje ispisa

Ploča prikazuje traku napretka za trenutačni dvominutni interval:

- **Cijan traka se puni** — prikupljaju se podaci signala (0–116 s unutar intervala).
- **Jantarna traka pulsira** — dekodiranje je u tijeku (posljednjih ~4 s intervala).
- **Prazna traka** — čeka se sljedeća parna UTC minuta.

Svaki uspješno dekodirani spot prikazuje se u tablici sa sljedećim stupcima:

Stupac	Značenje
UTC	Vrijeme spota (parna minuta)
Callsign	Postaja koja je odašiljala
Grid	Maidenhead lokator odašiljača
Power	Snaga odašiljanja u dBm
Freq	Točna audiofrekvencija (Hz) unutar WSPR propusnog pojasa
SNR	Omjer signala i šuma u dB

Kliknite **Clear** da izbrišete popis spotova.

**Savjet:** WSPR dekodiranje zahtijeva vrlo točno vrijeme sustava (unutar  $\pm 1$  sekunde od UTC-a). Prvi interval nakon uključivanja dekodera početak će u sljedećoj parnoj UTC minuti — kratko je čekanje uobičajeno.

## 8. HF FAX / WEFAX

**Što je to:** HF radiofaks (poznat i kao WEFAX) koriste obalne straže i meteorološke službe diljem svijeta za emitiranje vremenskih karata, karata stanja mora i prizemnih analiza na kratkim valovima. Dekoder rekonstruira sliku redak po redak kako je prima.

### Postavljanje

1. Uključite dekodera i iz padajućeg izbornika odaberite **HF FAX / WEFAX**. Ili jednostavno pritisnite gumb **FAX**.
2. Ploča **HF FAX / WEFAX Receiver** pojavljuje se ispod.
3. **Odaberite postaju** iz padajućeg izbornika Station. Dostupno je više od 20 postaja koje pokrivaju Europu, Aziju, Oceaniju i Ameriku (npr. DDH3/DDK3 Njemačka, SVJ4/GR Grčka, JMH Japan, NMG SAD New Orleans). Tu je i CBM Punta Arenas (Čile). Rasporedi prate NOAA-in svjetski raspored radiofaksa (izdanje od 7. ožujka 2025.); Charleville (VMC, Australija) uključuje svoje dodatne emisije samo na 11030 kHz.
4. Ako postaja emitira na više frekvencija, odaberite željenu iz podizbornika **Frequency**.
5. Kliknite ► **Tune** da automatski ugodite slap na tu postaju.
6. Način rada automatski se postavlja na **USB**.

### Raspored emitiranja

Kada je postaja odabrana, pojavljuje se tablica odbrojanja **Next Transmissions** koja prikazuje sljedeće 4 zakazane emisije u UTC-u, uz odbrojanje u stvarnom vremenu:

- Uobičajeno (sivo/zeleno) — emisija predstoji.
- **Jantarno** — emisija počinje za manje od 3 minute; odmah pripremite dekodera.
- **Crveno** ● — emisija počinje za manje od 30 sekundi; prijam samo što nije počeo.

### Parametri

Većina postaja koristi standardne zadane vrijednosti (označene s ★). Mijenjajte ih samo ako znate da postaja koristi nestandardne postavke.

Parametar	Zadano	Opis
LPM	120 ★	Redaka u minuti — brzina vrtnje bubnja
IOC	576 ★	Indeks suradnje — određuje broj piksela po retku
Shift	800 Hz ★	Frekvencijski pomak između tonova crnog i bijelog

## Kontrole

- ⇄ **Auto-align** — uključeno prema zadanom. Automatski se sinkronizira s faznim signalom na početku svake slike. Isključite ga samo ako imate problema s poravnanjem na signalu za koji znate da je dobar.
- ⚭ **Invert** — zamjenjuje crno i bijelo. Koristite ako slika izgleda kao negativ (bijela područja ondje gdje bi trebalo biti crno).
- ↻ **Refresh** — čisti platno i vraća dekodera na početak. Koristite ga između emisija ili ako se slika trga ili pomiče.
- ↓ **Save PNG** — sprema trenutno platno kao PNG datoteku na vaše računalo.

## Pokazatelji stanja

Na dnu slike dva pokazatelja tona prikazuju:

- **300 Hz phasing** — svijetli cijan kada je otkriven fazni ton početka slike.
- **450 Hz stop** — svijetli crveno kada je otkriven ton završetka slike.

Oba signala prepoznaju se po ritmu preko cijele širine retka, koju pravi signali početka i kraja uvijek ispunjavaju, a sadržaj karte praktički nikad, pa se šrafirana područja karte ne smatraju signalom početka. Prepoznaju se do oko 0 dB SNR-a, a kod vrlo slabih signala slika se poravnava prema prosjeku nekoliko redaka faziranja, pa karta i dalje počinje na lijevom rubu.

**Napomena:** slika klizi prema gore — najnoviji primljeni redak uvijek je na dnu platna. Ako u zaglavlju vidite **[PHASING]**, dekodera se uhvatio za početak nove slike.

## 9. NAVTEX

**Što je to:** NAVTEX je međunarodni pomorski sustav emitiranja obavijesti o sigurnosti u priobalju — navigacijskih upozorenja, vremenskih prognoza te obavijesti o traganju i spašavanju. Koristi FSK od 100 bauda (SITOR-B s FEC-om) i prima se na namjenskim kanalima diljem svijeta.

### Dostupni kanali

Kanal	Frekvencija	Namjena
Međunarodni	518 kHz	Na engleskom, međunarodni
Nacionalni	490 kHz	Emisije na nacionalnom jeziku
KV (×5)	4209.5 / 6314 / 8416.5 / 12579 / 16806.5 kHz	KV NAVTEX velikog dometa

### Postavljanje

1. Uključite dekodera i iz padajućeg izbornika odaberite **NAVTEX**. Ili jednostavno pritisnite gumb **NAVTEX**. Prijamnik automatski prelazi na **USB**.
2. Pojavljuje se ploča **NAVTEX Receiver**.

3. Odaberite željeni kanal iz izbornika **Station** (npr. International – 518 kHz).
4. Kliknite ⇒ **Tune & Set IF** da automatski ugodite slap i sužite propusni pojas na ispravan audioprozor. Način rada automatski se postavlja na **USB**. Prikazana frekvencija postavlja se 500 Hz ispod središta kanala, pa se NAVTEX signal pojavljuje na 500 Hz u zvuku.

## Raspored emitiranja

Tablica rasporeda navodi sve poznate postaje na odabranom kanalu s:

- njihovim ITU identifikacijskim slovom i zastavom države
- vremenom sljedeće emisije u UTC-u
- odbrojavanjem u stvarnom vremenu do sljedeće emisije
- **Jantarno** unutar 2 minute / **Crveno** ● unutar 30 sekundi — odmah pripremite dekode

## Čitanje ispisa

Dekodirani tekst pojavljuje se u tirkiznom fontu jednolike širine. Granice poruka jasno su označene:

```
— ZCZC MA12 —
... message content ...
— NNNN —
```

ZCZC označava početak poruke. Četiri znaka nakon njega označavaju stanicu (M), temu (A = navigacijska upozorenja) i redni broj (12). Natpis čeka cijelo zaglavlje, pa glasi ZCZC MA12, a ne ZCZC ????. NNNN označava kraj.

Ako se ugodite nakon što je poruka već počela, ili joj se zaglavlje izgubi u šumu, tekst se svejedno pojavljuje, ispod natpisa — **(header missed)** —. On se otvara tek kad stigne nekoliko znakova zaredom kod kojih se obje NAVTEX emisije slažu, što šum praktički nikad ne proizvodi. Zatvara se na NNNN kao i obično, s — **(signal lost)** — ako signal nestane prije, ili kad počne nova poruka.

Kliknite **Clear** da izbrišete međuspremnik poruka.

**Napomena:** NAVTEX radi na srednjim i dugim valovima (518/490 kHz). Domet prijama obično iznosi 200–400 nautičkih milja od odašiljača. KV kanali (4–17 MHz) pružaju znatno veći domet.

## 10. FSK / RTTY — uključujući PSK31 i Oliviju

**Što je to:** univerzalni dekode za uskopojasne tekstualne načine rada, s pet radnih inačica koje se biraju iz jednog izbornika. Tri su pravi FSK (frekvencijsko pomično ključanje): pomorski FSK (SITOR), meteorološki RTTY i amaterski RTTY. Druge dvije uopće nisu FSK, ali dijele isti prozor: **PSK31**, koji je fazno ključanje, i **Olivija**, koja je viševalni FSK s korekcijom pogrešaka. Svaka inačica dolazi s postavkom prilagođenom svojim standardnim parametrima.

## Inačice i postavke

Inačica	Središte	Shift	Baud	Okvir	Kodiranje
Pomorski FSK / SITOR	500 Hz	170 Hz	100	7N1	CCIR-476
Meteorološki RTTY	1000 Hz	450 Hz	50	5N1.5	ITA2
Amaterski RTTY	1000 Hz	170 Hz	45.45	5N1.5	ITA2
PSK31 (BPSK)	1000 Hz	—	31.25	—	Varicode
Olivia (MFSK)	1000 Hz	—	vidi Mode	—	7 bita + FEC

Ploča se prilagođava odabranoj inačici. **Shift**, **Baud**, **Framing**, **Encoding**, **Invert mark / space** i **Auto shift detect** skriveni su za PSK31 i Oliviju jer nijedan od ta dva načina nema par tonova mark/space niti okvir tipa UART. Umjesto toga **Center audio** postaje polje za slobodan unos broja (nosilac može biti bilo gdje u propusnom pojasu), a Olivia dobiva birač **Mode** i klizač **Squelch**.

## Postavljanje

1. Uključite dekodeer i odaberite **FSK / RTTY** iz izbornika. Ili jednostavno pritisnite gumb **RTTY**.
2. Pojavljuje se ploča dekodera. Naslov prati inačicu — *FSK / RTTY Decoder*, *PSK31 Decoder* ili *Olivia Decoder*.
3. Odaberite **Variant**. Parametri se ažuriraju automatski.
4. Za Oliviju postavite **Mode** (tonovi / širina pojasa) tako da odgovara emisiji — vidi napomene o Oliviji niže.
5. Iz izbornika **Known frequency** odaberite uobičajenu frekvenciju za odabranu inačicu i kliknite **Tune** da skočite na nju.
6. Fino ugađajte dok dekodirani tekst ne postane stabilan i čitljiv.

## Poznate frekvencije po inačici

### Pomorski FSK / SITOR

- 518,0 kHz — međunarodni NAVTEX
- 490,0 kHz — nacionalni NAVTEX
- 4209,5 / 6314,0 / 8416,5 / 12579,0 / 16806,5 / 22376,0 kHz — HF SITOR

### Meteorološki RTTY

- 4583,0 / 7646,0 / 10100,8 / 11039,0 / 14467,3 kHz — DWD (njemačka meteorološka služba)

### Amaterski RTTY

- 3590 kHz (80 m), 7043 kHz (40 m), 10143 kHz (30 m), 14083 kHz (20 m), 21083 kHz (15 m), 28083 kHz (10 m)

### PSK31

- 3580,15 kHz (80 m), 7040,15 kHz (40 m), 10142,15 kHz (30 m), 14070,15 kHz (20 m), 18100,15 kHz (17 m), 21080,15 kHz (15 m), 24920,15 kHz (12 m), 28120,15 kHz (10 m)

### Olivia

- 3577,75 kHz (80 m), 7073,75 kHz (40 m), 10142,25 kHz (30 m), 14075,5 kHz (20 m), 18103,75 kHz (17 m), 21075,75 kHz (15 m), 24921,75 kHz (12 m), 28123,75 kHz (10 m)

## Parametri

Parametar	Odnosi se na	Opis
Center audio (Hz)	sve	Audio frekvencija sredine između mark i space; za PSK31 nosilac, za Oliviju središte bloka tonova. Padajući izbornik za FSK inačice, polje za slobodan unos za PSK31 i Oliviju
Shift (Hz)	samo FSK	Frekvencijska razlika između tonova mark i space
Baud	samo FSK	Brzina simbola
Framing	samo FSK	Podatkovni bitovi, paritet, stop bitovi (npr. 7N1 = 7 podataka, bez pariteta, 1 stop)
Encoding	samo FSK	Skup znakova (CCIR-476, ITA2/Baudot ili ASCII)
Invert mark / space	samo FSK	Zamjenjuje tonove mark i space
Auto shift detect	samo FSK	Pokušava automatski izmjeriti shift iz dolaznog signala
Mode (tonovi / Hz)	samo Olivia	Broj tonova i širina pojasa — mora točno odgovarati emisiji
Squelch (FEC S/N)	samo Olivia	Koliko jako podudaranje korekcije pogrešaka mora biti prije ispisa teksta
Squelch (SNR)	Amaterski i meteorološki RTTY	Sprječava da se šum ispisuje kao nasumična slova. Počinje na $-5$ dB za amaterski RTTY i na $-8$ dB za meteorološki; skroz lijevo ( <b>off</b> ) je isključen. Brojka SNR postaje siva dok je squelch zatvoren

## Mjerenja signala

Statusna traka prikazuje mjerenja uživo, a polja se mijenjaju s inačicom:

Inačica	Prikazana polja
FSK inačice	<b>Mark / Space</b> (izmjerene frekvencije tonova), <b>SNR</b> , <b>Lock</b> , <b>Timing</b>
PSK31	<b>Carrier</b> (Hz, nakon automatske korekcije frekvencije), <b>IMD</b> (dB), <b>SNR</b> , <b>Lock</b> , <b>Timing</b>
Olivia	<b>Centre</b> (Hz), <b>Mode</b> , <b>S/N</b> , <b>FEC</b> (%), <b>Sync</b>

Timing/Sync pokazuje LOCKED/SYNCED kada dekodir prati signal, a SEARCH dok ga još traži.

**SNR** FSK varijanti i PSK31 mjeri se na samom signalu i daje u širini pojasa od 3 kHz, pa se oba čitaju na istoj ljestvici: s točnošću od oko 1 dB od  $+20$  dB do  $-5$  dB. Prazna frekvencija pokazuje oko  $-12$  do  $-15$  dB, a kod PSK31 0 dB je već jak signal. **S/N** kod Olivije je nešto drugo: pokazatelj kvalitete ispravljanja pogrešaka.

Amaterski RTTY (45,45 bauda, 170 Hz) čita se bez pogrešaka do oko 0 dB, s oko 1,5 % pogrešnih znakova na  $-3$  dB i 20 % na  $-6$  dB.

## Dodatne kontrole

- ⇒ **Set IF Band-Pass** — sužava propusni pojas prijamnika tako da tijesno obuhvati signal. Širina prati varijantu: 300 Hz za amaterski RTTY (dovoljno usko da ukloni jaču stanicu 250 Hz dalje, a i dalje čita signal razdešen za 40 Hz), 594 Hz za meteorološki RTTY, mark/space plus rezerva za pomorski FSK, oko  $\pm 100$  Hz za PSK31 i cijela širina bloka tonova plus rezerva za Oliviju. Propusni pojas prati dekodir kad **Auto-tune** ili **Auto shift detect** pomakne tonove te kad promijenite postavku **Baud**.

- **Auto-tune Center** — automatsko traženje signala. Za FSK inačice traži uravnotežen par tonova, za PSK31 pronalazi nosilac, a za Oliviju najjači blok odabrane širine pojasa. **Auto shift detect** mijenja shift samo kad je prisutan pravi dvotonski signal, pa više ne skače na šum.

## Napomene o PSK31

PSK31 je najrašireniji način rada tipkovnica-na-tipkovnicu na KV. Širok je samo 62 Hz, pa nekoliko veza stoji jedna uz drugu unutar nekoliko stotina herca oko okupljališne frekvencije, a ugađanje se svodi na odabir jednog traga iz skupine na slapu.

- Dekoder sam ispravlja svoju pogrešku ugađanja u rasponu od otprilike **±25 Hz**, pa je dovoljno da se približite. **Carrier** u retku mjerenja pokazuje gdje se zaista smjestio.
- **IMD** mjeri kvalitetu odašiljača, a ne prijam: čist signal pokazuje oko **−20 dB** ili bolje. Loša vrijednost znači da suprotna stanica preporučuje svoj odašiljač, a ne da ste vi loše ugođeni.
- Tekst se pojavljuje znak po znak bez korekcije pogrešaka, pa se slab signal degradira u povremena kriva slova umjesto da stane.

## Napomene o Oliviji

Olivia žrtvuje brzinu za otpornost. Znatno je sporija od PSK31, ali dekodira signale koji se uhom uopće ne čuju, zbog čega je omiljena za slabe signale i veze na velike udaljenosti.

- **Postavka Mode mora točno odgovarati emisiji.** Pogrešna kombinacija tonova i širine pojasa ne dekodira baš ništa — ne iskrivljen tekst, nego tišinu. Ploča se otvara na **8 / 250**, uskoj konfiguraciji koja se ostavlja uključena na pozivnim frekvencijama; **16 / 500** i **32 / 1000** druga su dva u čestoj uporabi, a nudi se i **16 / 1000**.
- Olivia ne šalje preambulu, pa dekode mora tražiti sinkronizaciju. **Pričekajte nekoliko sekundi** nakon ugađanja prije nego što se pojavi tekst. Polje **Sync** pokazuje **SEARCH** dok se ne zaključa.
- Korekcija pogrešaka radi po blokovima, pa tekst stiže **u naletima, a ne u ravnomjernom toku**, uz kašnjenje od nekoliko blokova između emitiranja i prikaza.
- **Squelch (FEC S/N)** određuje koliko sigurna korekcija pogrešaka mora biti prije ispisa. Zadana vrijednost 4,0 drži šum vani; 3,0 je donja granica ispod koje slučajni šum počinje ispisivati pokoji znak. Dobar signal pokazuje 8–9 na mjerачу **FEC**, pa ima dosta prostora za podizanje squelcha na prometnom opsegu.

**Napomena o načinu rada:** dekode preuzima upravljanje načinom demodulacije i propusnim pojasom MF dok je aktivan. Oboje se automatski vraća kada ga isključite. Svih pet inačica koristi **USB**.

**Napomena o polaritetu (samo FSK inačice):** za meteorološki RTTY obično treba označiti **Invert mark / space**. Za pomorski FSK (tipa SITOR/NAVTEX) i amaterski RTTY ostavite neoznačeno — amaterski RTTY šalje mark kao višu radiofrekvenciju, a USB je zadržava kao viši audio ton, što je upravo neoznačeni slučaj. Ako je dekodirani tekst iskrivljen, prvo što treba probati jest prebaciti ovaj okvir. Okvir je skriven za PSK31 i Oliviju, koje nemaju par mark/space.

**Napomena o slovima/brojka (samo FSK inačice):** Baudot drži slova i brojke u dva odvojena stanja, a šum može prebaciti dekode u pogrešno — što izobličuje svaki sljedeći znak, ne samo oštećeni. Zato se dekode na svakom razmaku vraća na slova; to je uobičajena praksa i popravljiva pokvareno prebacivanje unutar jedne do dvije riječi umjesto cijelog retka. Cijena je da skupine brojk odvojene razmacima traže da pošiljalac ponovi prebacivanje na brojke nakon svakog razmaka, kako odašiljači inače i rade.

## 11. SSTV

**Što je to:** televizija sporog raščlanjivanja prenosi nepomične slike običnim govornim SSB kanalom, redak po redak, kao frekvencijski moduliran ton između 1500 Hz (crno) i 2300 Hz (bijelo). Puna slika traje između 36 sekundi i 4 i pol minute, ovisno o načinu rada.

### Preporučene frekvencije (USB)

Pojas	Frekvencija	Način rada	Napomene
20 m	14.230 MHz	USB	Glavna međunarodna pozivna frekvencija za SSTV — daleko najaktivnija
20 m	14.233 MHz	USB	Pričuvna, kada je 14.230 zauzeta
15 m	21.340 MHz	USB	
10 m	28.680 MHz	USB	Aktivna tijekom otvaranja pojasa
40 m	7.171 MHz	LSB	
80 m	3.845 MHz	LSB	Regionalna, navečer

**Bočni pojas bira se umjesto vas.** Pokretanjem dekodera odabire se **LSB ispod 10 MHz i USB iznad**, prema uobičajenoj amaterskoj praksi. Na pogrešnom bočnom pojasu preslikavanje tonova je obrnuto i slika se neće dekodirati. Ako naiđete na postaju koja zanemaruje konvenciju, jednostavno ručno promijenite bočni pojas — dekodir nastavlja raditi, briše okvir i kreće iznova s novom postavkom.

### Postavljanje

1. Uključite dekodir i iz padajućeg izbornika odaberite **SSTV**. Ili jednostavno pritisnite gumb **SSTV**. Prijamnik automatski mijenja bočni pojas — USB iznad 10 MHz, LSB ispod.
2. Ugodite se tako da tonovi slike padnu u sredinu propusnog pojasa. Ispravno ugođen signal ima sinkroimpulsa na 1200 Hz, a sadržaj slike između 1500 i 2300 Hz.
3. Ostavite **Mode** na **Auto** osim ako već znate što se šalje. Slika se gradi redak po redak kako pristije.

### Načini rada

Postavka	Slika	Trajanje
<b>Auto</b>	Otkriva se automatski	—
Martin M1 / M2	320×256 u boji	114 s / 58 s
Scottie S1 / S2	320×256 u boji	110 s / 71 s
Scottie DX	320×256 u boji	269 s
Robot 36 / 72	320×240 u boji	36 s / 72 s

**Auto** radi na dva načina: čita **VIS zaglavlje** — digitalni kod načina rada poslan u prvih 300 ms emisije — a ako je zaglavlje propušteno (ugodili ste se prekasno ili se izgubilo u QSB-u), način rada prepoznaje prema vremenskim odnosima sinkroimpulsa. Odabir određenog načina rada nameće taj način, ali dekodir i dalje provjerava sinkronizaciju prije nego što išta nacrti, pa pogrešan odabir daje izostanak slike umjesto šuma.

### Čitanje ispisa

Statusni redak ispod kontrola pokazuje što dekodir radi:

Stanje	Značenje
Waiting for VIS / AUTO lock	Osluškuje; još ništa nije prepoznato
Martin M1? verifying sync...	Pronađen je kandidat i potvrđuje se prema sljedećim recima
Martin M1 lock (VIS)	Zaključano prema VIS zaglavlju
Martin M1 lock (AUTO)	Zaključano prema vremenskim odnosima sinkroimpulsa
Martin M1 lock (MANUAL)	Pokrenuto gumbom ● <b>Force</b>
Candidate rejected (no sync)	Kandidat nije potvrđen — uobičajeno na šumu
– sync lost, resetting	Signal je nestao usred slike
Frame complete	Primljena je cijela slika

Prepoznati način rada pojavljuje se i zelenom bojom uz naslov **SSTV**, a brojač redaka pokazuje napredak.

## Kontrole

Gumb	Radnja
► <b>Start</b>	Priprema dekode. Sam po sebi ne crta sliku — zaključavanje i dalje mora doći iz VIS zaglavlja ili iz otkrivanja sinkronizacije.
■ <b>Stop</b>	Zaustavlja dekodiranje.
↺ <b>Reset</b>	Čisti platno i ponovno priprema dekode na mjestu. Koristite ga između slika ili nakon ponovnog ugađanja.
● <b>Force</b>	Počinje crtati <b>odmah</b> u načinu odabranom u izborniku, potpuno preskačući VIS i otkrivanje sinkronizacije. Dostupno samo kada Mode nije <b>Auto</b> .
<b>Save</b>	Sprema trenutačnu sliku kao PNG.

**Kada koristiti Force.** Ako u slapu vidite sliku, ali se dekode ne može zaključati na nju — neuobičajen način rada, signal preslab ili previše izobličen za detektore, ili propušteno zaglavlje — odaberite način rada ručno i pritisnite ● **Force**. Okvir se usidruje u trenutku klika, a oznaka načina rada prikazuje **MANUAL** kako biste ga razlikovali od VIS ili AUTO zaključavanja. Dekoder se i dalje poravnava prema stvarnom sinkroimpulsu ako ga pronađe, pa se klik koji je malo preran ili prekas ispravlja.

Budući da Force preskače svaku sigurnosnu provjeru, rado će nacrtati šum ako ga pritisnete na praznom kanalu i neće se sam zaustaviti — pritisnite ■ **Stop** ili ↺ **Reset**.

## Napomene

- **Šum ne pokreće dekode.** Nekoć su atmosferski praskovi, iskre i mrežne smetnje bili dovoljni da pokrenu dekodiranje. Svaki stupanj otkrivanja sada provjerava je li ton doista ton i potvrđuje kandidata prema sljedećim recima prije nego što se nacrti i jedan piksel. Očekujte da će dekode mirovati na praznom pojasu.
- **Slika se kosi dijagonalno ako niste na frekvenciji.** SSTV ne oprašta pogreške u ugađanju. Ako se reci naginju, podesite frekvenciju u malim koracima i pustite da počne sljedeća slika.
- Ravnoteža boja i oština su fiksne; nema se što podešavati.

## 12. Opći savjeti

**Prvo se mora uključiti Decoder: ON.** Padajući je izbornik onemogućen (zasivljen) dok ne kliknete gumb Decoder.

**Jedan dekodier istodobno.** Odabir novog dekodera iz izbornika automatski zaustavlja prethodno aktivni i poništava sve promjene načina rada ili propusnog pojasa koje je napravio.

**Načinom rada upravlja se umjesto vas.** HF FAX i NAVTEX prebacuju prijamnik na USB čim ih odaberete, SSTV bira bočni pojas prema pojasu (LSB ispod 10 MHz, USB iznad), a FAX, NAVTEX i FSK usto podešavaju propusni pojas kada kliknete njihov gumb Tune. FT8, FT4, FT2, JS8 i WSPR čine isto čim ih odaberete — iz reda gumba ili s padajućeg izbornika: prijamnik prelazi na USB i dobiva svoj propusni pojas — cijeli podpojas od 3 kHz za obitelj FT8, 1350–1650 Hz za WSPR. Kada zaustavite dekodier, vraća se zadani način rada za taj pojas.

**Uključivanje dekodera više ne prekida zvuk.** Dekoderi rade u vlastitim dretvama, pa nema praznine, klika ni ispada kada se koji pokrene, zaustavi ili zamijeni drugim.

**Točnost sata sustava je važna.** FT8, FT4 i WSPR vremenski su kritični. Dekodiraju u fiksnim prozorima usklađenima s UTC-om. Ako sat vašeg računala odstupa više od 1–2 sekunde, stopa dekodiranja znatno će pasti. Koristite NTP klijent kako bi sat bio točan.

**Filtri šuma ne dopiru do dekodera.** NR, NB, NS i AN pomagala su za slušanje, namijenjena samo vašim ušima. Svaki dekodier — FT8, FT4/FT2, CW, WSPR, FAX, NAVTEX, FSK/RTTY/PSK31/Olivia, SSTV i QRSS grabber — uzima zvuk *prije* tih filtara, pa ih slobodno namjestite kako najbolje zvuči, bez brige za kvalitetu dekodiranja. Iz istog razloga dekodier rade i dok je prijamnik utišan ili pod squelchom: možete ugasiti zvučnik i pustiti da dekodier, ili noćno QRSS snimanje, nastavi prikupljati. Jedino što doista prati ono što čujete jest audio spektrogram, koji namjerno prikazuje filtrirani zvuk.

**Kvaliteta signala važnija je od njegove jakosti.** Većina je ovih dekodera namijenjena slabim signalima. Mirniji pojas s manje šuma često je plodonosniji od glasnog signala punog smetnji. Prije uključivanja dekodera koristite slap i kontrole propusnog pojasa kako biste prepoznali i izbjegli QRM.

**Slobodno koristite gumbe Refresh i Clear.** Slike FAX-a se pomiču ako je prikazana frekvencija malo pomaknuta, a tekstualni dekodier nakupljaju šumne znakove. Novi početak nakon podešavanja ugađanja često daje mnogo čišći ispis.

### Automatska prijava spotova i grafikoni poslužitelja

Dekodiranja FT8, FT4 i WSPR može slati i sam poslužitelj — FT8/FT4 na PSK Reporter, WSPR na WSPRnet — pomoću autorun daemona koji sysop pokreće s administratorske ploče. Neovisan je o dekodierima u vašem pregledniku: radi bez obzira sluša li tko ili ne, a ništa što dekodirate u pregledniku ne prijavljuje se.

Sysop ga prati preko dva brojača koja je lako zamijeniti: pločice po dekodieru broje spotove poslane od zadnjeg pokretanja daemona, dok je broj uz svaki potvrdni okvir pojasa/načina rada ukupan zbroj tog mjesta od početka i preživljava ponovna pokretanja. Ista ploča ima stranicu **Grafikoni** koja crta frekvenciju procesora, opterećenje, temperaturu i broj korisnika kroz zadnjih 15 minuta do 24 sata. Oboje je opisano u [vodiču za administratorsku ploču](#).

# RADE README

RADE\_README.md

## Digitalni glas RADE v1 za PhantomSDR-Plus

**RADE** (Radio AutoencoDEr) vodeći je FreeDV-ov način rada digitalnog glasa na kratkim valovima. Koristi hibrid strojnog učenja i DSP-a (neuronski vokoder FARGAN) i pruža govor visoke kvalitete na KV-u pri omjerima signal/šum do  $-2$  dB, u tek 1500 Hz RF širine pojasa — uže od SSB signala.

Ovaj dokument obrađuje potpunu integraciju prijama RADE v1 u PhantomSDR-Plus, izvedenu kao sidecar proces u Pythonu (`rade_helper.py`) koji povezuje preglednik s lancem dekodiranja `radac_rx.py` + `lpcnet_demo`.

## Instalacija — kraći put

```
cd ~/PhantomSDR-Plus
./install_rade.sh
```

To je cijela instalacija. Skripta instalira sistemske pakete preko upravitelja paketa koji stroj ima (`apt`, `pacman`, `dnf` ili `zypper`), za ono što distribucija ne isporučuje poseže za `pipom`, nadograđuje `websockets` ako je verzija iz distribucije starija od 11.0 koju sidecar treba, klonira i gradi `radac`, provjerava težine modela, ponovno gradi frontend i pokreće sidecar. Na Ubuntuu 22.04 predaje posao skripti `install_rade_ubuntu22.sh`. `./install.sh` nudi da je pokrene u sklopu normalne instalacije, pa je na svježem stroju RADE već tu.

**Ovaj dokument objašnjava kako RADE radi i kako ga pokrenuti — nije postupak instalacije.**

Ako instalaciju morate obaviti ručno, na sustavu koji skripta ne pokriva ili da biste popravili jedan korak, slijedite vodič povezan niže; koraci su zapisani ondje i nigdje drugdje. Sve ovdje — arhitektura, zakrpane datoteke, vrata, upotreba RADE-a u pregledniku, varijable okoline, rješavanje problema i mjerenje istodobnosti — vrijedi za svaku instalaciju, kako god je napravljena.

- **Ručna instalacija na Linuxu** - ručni put, korak po korak, za Ubuntu, Debian, Fedoru, Arch i Raspberry Pi OS

## Sadržaj

1. Instalacija — kraći put
2. Kako to radi
3. Pregled arhitekture
4. RADEL naspram RADEU
5. Preduvjeti
6. Ručna instalacija
7. Postavljanje zakrpanih datoteka
8. Upravljanje sidecarom
9. Vrata 8074
10. Korištenje RADE-a u pregledniku
11. Provjera i otklanjanje pogrešaka

12. [Varijable okruženja](#)
13. [Promijenjene datoteke](#)
14. [Sažetak toka signala](#)
15. [Rješavanje problema](#)
16. [Mjerenje istodobnosti na vlastitom hardveru](#)
17. [Ažuriranje RADE v1](#)

## Kako to radi

RADE v1 ne može raditi u pregledniku — traži PyTorch i neuronski vokoder FARGAN, koji su preveliki za WASM. Rješenje je sidecar proces u Pythonu ( `rade_helper.py` ) koji radi na istom poslužitelju kao PhantomSDR-Plus.

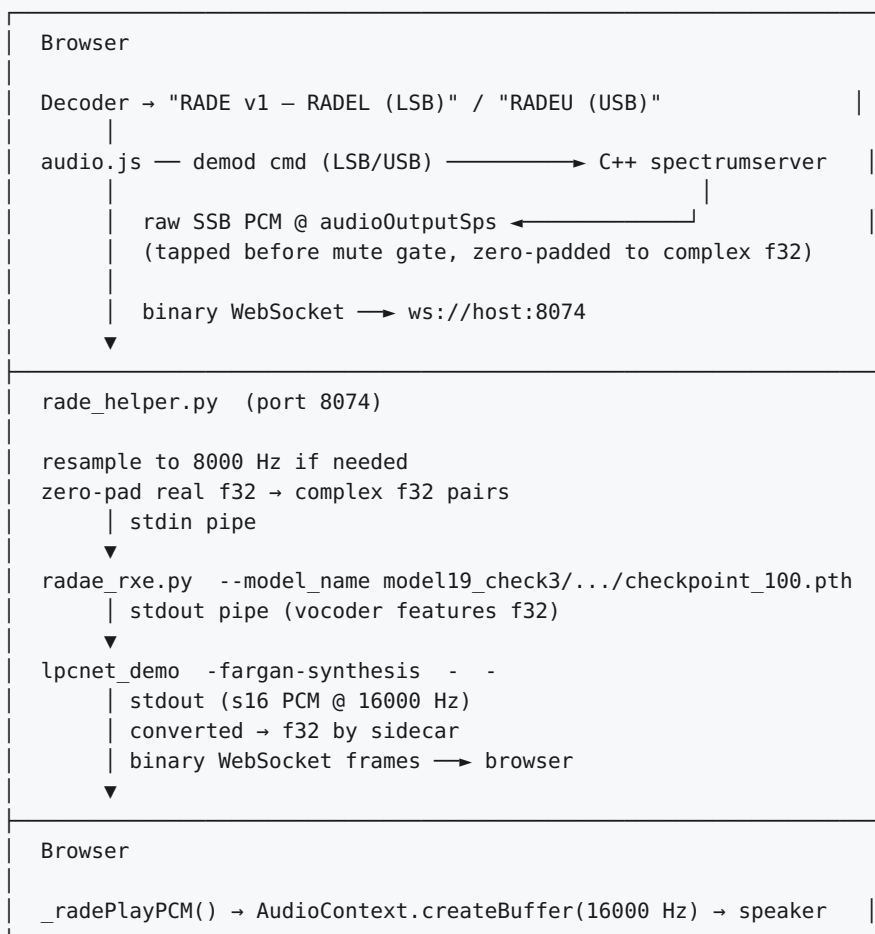
**Važno:** `freedv_rx` iz repozitorija codec2 **ne** podržava RADEV1. Lanac dekodiranja RADE u cijelosti se nalazi u zasebnom repozitoriju `radae` Davida Rowea (VK5DGR). Nemojte pokušavati koristiti codec2-ov `freedv_rx` za RADE.

Kada u pregledniku odaberete RADE:

1. Sučelje postavlja osnovnu demodulaciju na USB ili LSB — poslužitelj u C++-u demodulira SSB signal na uobičajen način.
2. Sirovi demodulirani PCM preuzima se u `audio.js` **prije** bilo kakvog utišavanja ili squelch vrata — `radae_rxe.py` treba neprekidan ulaz da bi održao sinkronizaciju okvira.
3. Svaki blok PCM-a dopunjava se nulama iz realnog f32 u kompleksni f32 (realni + 0.0 imaginarni) — upravo to `radae_rxe.py` očekuje na standardnom ulazu.
4. Sidecar te uzorke prosljeđuje u `radae_rxe.py`, koji ispisuje značajke vokodera.
5. `lpcnet_demo -fargan-synthesis` pretvara značajke u govor s16 pri 16000 Hz.
6. Sidecar pretvara s16 → f32 i vraća to pregledniku kao binarne WebSocket okvire.
7. Preglednik reproducira dekodirani govor putem Web Audio API-ja pri 16000 Hz.

Poslužitelj u C++-u ( `spectrumserver.cpp` ) uopće se ne mijenja.

## Pregled arhitekture



## RADEL naspram RADEU

RADE v1 uvijek se prenosi putem SSB-a. Prema konvenciji:

Način rada	Bočni pojas	Koristiti na pojasevima
RADEL	LSB	160 m, 80 m, 40 m ( $\leq 10$ MHz)
RADEU	USB	20 m, 17 m, 15 m, 12 m, 10 m ( $> 10$ MHz)

## Preduvjeti

Zahtjev	Verzija	Napomene
PhantomSDR-Plus	bilo koja	s Vite/Svelte sučeljem
Linux	Ubuntu 24.04+ / Debian Bookworm+	testirano
Python	3.8+	za <code>rade_helper.py</code> i <code>raae_rxe.py</code>
repozitorij radea	najnoviji master	daje <code>raae_rxe.py</code> i <code>lpcnet_demo</code>
cmake	3.10+	za izgradnju <code>lpcnet_demo</code> iz radea
PyTorch	2.0+	traži ga <code>raae_rxe.py</code>
Node.js	16+	za <code>npm run build</code>
websockets (Python)	10-16+	<code>pip3 install websockets</code> — podržane sve verzije
matplotlib	bilo koja	traži ga <code>raae_rxe.py</code> pri uvozu
numpy	1.23+	obavezan; omogućuje i ponovno uzorkovanje
scipy	bilo koja	neobavezno, omogućuje precizno ponovno uzorkovanje

## Ručna instalacija

`./install_rade.sh` obavlja cijelu instalaciju. Kada je želite obaviti sami — na sustavu koji skripta ne pokriva ili da biste popravili jedan korak — postupak se nalazi u **Vodiču za ručnu instalaciju RADE v1**, i samo ondje. Pokriva Ubuntu, Debian, Fedoru, Arch i Raspberry Pi OS u jednom prolazu:

Korak 1	Paketi sustava i provjera Node.js 22+
Korak 2	Python paketi, uključujući zastavicu PEP 668 i torch kotačić samo za CPU
Korak 3	Kloniranje i izgradnja repozitorija radea, provjera <code>lpcnet_demo</code> i težina modela
Korak 4	Provjera lanca dekodiranja izvan mreže, prije spajanja na preglednik
Korak 5	Izgradnja PhantomSDR-Plus frontenda
Korak 6	Upravljanje sidecarom
Korak 7	Otvaranje vrata 8074
Koraci 8–10	Pokretanje poslužitelja, provjera RADE-a, upotreba u pregledniku

Ostatak ovog dokumenta pretpostavlja da je to obavljeno.

## Postavljanje zakrpanih datoteka

Kopirajte sljedeće datoteke iz skupa zakrpa u svoj repozitorij PhantomSDR-Plus.

## Nova datoteka — stavite je u korijen repozitorija, uz početnu skriptu koju koristite:

```
rade_helper.py
```

## Zakrpane datoteke sučelja — stavite ih u `frontend/src/`:

```
audio.js
App.svelte
```

## Što zacrpe rade

**audio.js** — 5 izmjena:

- Konstruktor: 5 novih polja stanja za RADE (`decodeRADE`, `_radeSideband`, `_radeSocket`, `_radeCallback`, `_radeReady`, `_radeNextTime`)
- Spremanje PCM-a prije pojačanja: `pcmArrayPreBoost` sprema se prije 300× FLAC pojačanja kako bi RADE dobio izvornu amplitudu (pojačanje bi zasitilo `radae_rxe.py`)
- `playAudio()`: preuzimanje PCM-a za RADE pomoću zvuka prije pojačanja, prije vrata utišavanja/squelcha
- `playAudio()`: zaštita `if (this.decodeRADE) return` — potiskuje reprodukciju sirovog SSB-a dok svira govor koji je dekodirao RADE
- Nove metode: `setRADEDecoding()`, `setRADECallback()`, `_radePlayPCM()` s planiranom reprodukcijom bez praznina pomoću sata `_radeNextTime` (reproducira na **16000 Hz** — izlaznoj frekvenciji `lpcnet_demo`)

**Svaka Svelte inačica** — 6 izmjena:

- `demodulationDefaults`: `RADEL {type: 'LSB', offsets: [2200, -700]}`, `RADEU {type: 'USB', offsets: [-700, 2200]}` — propusni pojas počinje 700 Hz od nosioca i širok je 1500 Hz
- Varijable stanja: `radeEnabled`, `radeConnected`, `radeSynced`, `radeSnr`, `_radeDeactivate()`
- `_radeDeactivate()`: zaustavlja dekodirer i **vraća zadani način rada za pojas** iz `bands-config.js` — ponašanje identično isključivanju FAX-a, NAVTEX-a i FSK-a
- `_deactivateAll()`: redak za čišćenje RADE-a
- `activateSelectedDecoder()`: grane `radel` i `radeu`
- Padajući izbornik dekodera: dvije nove stavke `<option>`
- Ploča stanja: točka povezanosti, stanje sinkronizacije/SNR-a, traka pogreške

## Početne skripte

Pokretači prijarnika (`start-rx888mk2.sh`, `start-airspyhf.sh`, `start-rtl.sh`, `start-rspla.sh`, `start-fobos-hf.sh`, `start-fobos.sh`, `start-hackrf.sh`) sami pokreću sidecar čim poslužitelj proradi, a `stop-websdr.sh` zaustavlja ga zajedno s poslužiteljem. `rade.sh` ostaje za samostalno pokretanje sidecara — vidi [Upravljanje sidecarom](#).

## Upravljanje sidecarom

**Obično vam ne treba zasebna upravljačka skripta za RADE.** Pokretači prijarnika (`start-rx888mk2.sh`, `start-airspyhf.sh`, `start-rtl.sh`, `start-rspla.sh`, `start-fobos-hf.sh`, `start-fobos.sh`, `start-hackrf.sh`) sami pokreću sidecar čim se potvrdi da poslužitelj radi, ponovno ga pokreću ako se ugasi i javljaju njegovo stanje u završnom sažetku — ✓ `RADE sidecar running`, ili napomena `•` koja

objašnjava zašto nije aktiviran (nije instaliran, `RADE_ENABLED=0` ili nema `python3`). `./stop-websdr.sh` zaustavlja ga zajedno s poslužiteljem. Za rad poslužitelja bez RADE-a pokrenite ga s `RADE_ENABLED=0`.

`rade.sh` i dalje postoji za samostalnu upotrebu — pokretanje sidecara bez PhantomSDR-Plus pokretača ili njegovo zasebno testiranje:

```
chmod +x ~/PhantomSDR-Plus/rade.sh
cd ~/PhantomSDR-Plus
./rade.sh start # start sidecar with self-restarting watchdog
./rade.sh stop # stop sidecar + watchdog + all lpcnet_demo children
./rade.sh restart # stop then start cleanly
./rade.sh status # show running / stopped + process info
```

Aktivnost sidecara bilježi se u `~/PhantomSDR-Plus/rade.log`:

```
tail -f ~/PhantomSDR-Plus/rade.log
```

**Nemojte mijesati to dvoje dok poslužitelj radi.** Ako je čuvar nekog pokretača aktivan, vratit će sidecar u roku od otprilike pet sekundi, pa će izgledati kao da `./rade.sh stop` nije uspio. Da biste zaustavili RADE dok poslužitelj radi, pokrenite poslužitelj s `RADE_ENABLED=0` ili zaustavite sve s `./stop-websdr.sh`.

Ako `rade.sh` pokrećete samostalno, **uvijek koristite `./rade.sh stop`** — sam `kill -f rade_helper.py` ne pobjeđuje njegova čuvara, koji proces ponovno pokreće u roku od 3 sekunde.

## Vrata 8074

Sidecar sluša na TCP vratima **8074**, a preglednik se spaja izravno na njih, pa vrata moraju biti dostupna izvana. Njihovo otvaranje — `ufw`, `firewalld`, `iptables`, NAT pravilo na usmjerivaču i Nginx proxy za slučaj da davatelj usluge posve blokira vrata — [7. je korak ručnog vodiča](#).

Dvije stvari vrijedi ovdje ponoviti jer oduzimaju najviše vremena:

```
Is the sidecar listening?
ss -tlnp | grep 8074
```

**Upozorenje o NAT hairpinu:** provjera naredbom `curl` s poslužitelja prema njegovu javnom imenu često vraća `Connection refused` i onda kada su vrata otvorena — mnogi usmjerivači ne vraćaju promet natrag. Uvijek testirajte s vanjskog računala ili upotrijebite <https://portchecker.co>.

## Korištenje RADE-a u pregledniku

1. Otvorite svoje web-sučelje PhantomSDR-Plusa
2. Aktivne postaje potražite na [qso.freedv.org](https://qso.freedv.org)
3. Ugodite se na prikazanu frekvenciju postaje
4. U **Decoder Options** odaberite:
5. **RADE v1 — RADEL (LSB)** za 40 m / 80 m / 160 m
6. **RADE v1 — RADEU (USB)** za 20 m / 17 m / 15 m / 12 m / 10 m
7. Kliknite **Decoder: ON**

## Gumbi RADEL / RADEU (jedan pritisak)

Korake 4 i 5 može zamijeniti jedan pritisak. Par gumba **RADEL** / **RADEU** dostupan je na tri mjesta:

- uz naslov **Modes selector** na glavnoj ploči;
- u skočnom prozoru **Modes**;
- u skočnom prozoru **Bands**.

Pritisak odabire dekodera, uključuje dekodera (ON), zatvara skočni prozor u kojem ste ga pritisnuli i dovodi RADE ploču na zaslon. Gumb je plav dok RADE radi. **Ponovni pritisak na isti gumb isključuje RADE** — ploča se zatvara, a način rada i propusni pojas vraćaju se kako je opisano niže. Gumbi, padajući izbornik **Decoder Options** i gumb ON/OFF dijele isto stanje.

## Stanja pokazatelja na ploči

Pokazatelj	Značenje
Crveno — „Connecting to sidecar...”	Port 8074 nedostupan ili sidecar ne radi
Žuto — „Searching for signal...”	Sidecar spojen, RADE okvir još nije otkriven
Zeleno — „Synced · SNR x.x dB”	Dekodiranje — govor se reproducira

## Kada isključite dekodera

Način rada i propusni pojas automatski se vraćaju na ispravnu zadanu vrijednost za trenutačnu frekvenciju, kako je određeno u `bands-config.js` — jednako kao kod FAX-a, NAVTEX-a i FSK-a.

## Provjera i otklanjanje pogrešaka

### Radi li sidecar?

```
ps aux | grep rade_helper | grep -v grep
ss -tlnp | grep 8074
```

### Praćenje aktivnosti uživo

```
tail -f ~/PhantomSDR-Plus/rade.log
```

Kada se preglednik spoji:

```
[RADE] client connected: x.x.x.x:XXXXX
[RADE] x.x.x.x:XXXXX sps=12000 sideband=LSB
[RADE] x.x.x.x:XXXXX spawning pipeline (torch_threads=1)
```

## Brza provjera WebSocketa

```
python3 - <<< 'EOF'
import asyncio, websockets, json

async def test():
 async with websockets.connect('ws://localhost:8074') as ws:
 await ws.send(json.dumps({'type': 'init', 'sps': 8000, 'sideband': 'LSB'}))
 print(await ws.recv()) # expect: {"type": "status", "connected": true}

asyncio.run(test())
EOF
```

## Konzola preglednika (F12)

[RADE] ► ENABLED LSB @ 12000 Hz → helper ws://localhost:8074

Dobro — WebSocket je otvoren. Vrijednost u Hz odgovara `audioOutputSps` vašeg poslužitelja (obično 8000–12000 Hz).

[RADE] sidecar socket error

Port 8074 nije dostupan — provjerite vatrozid i NAT pravilo usmjerivača.

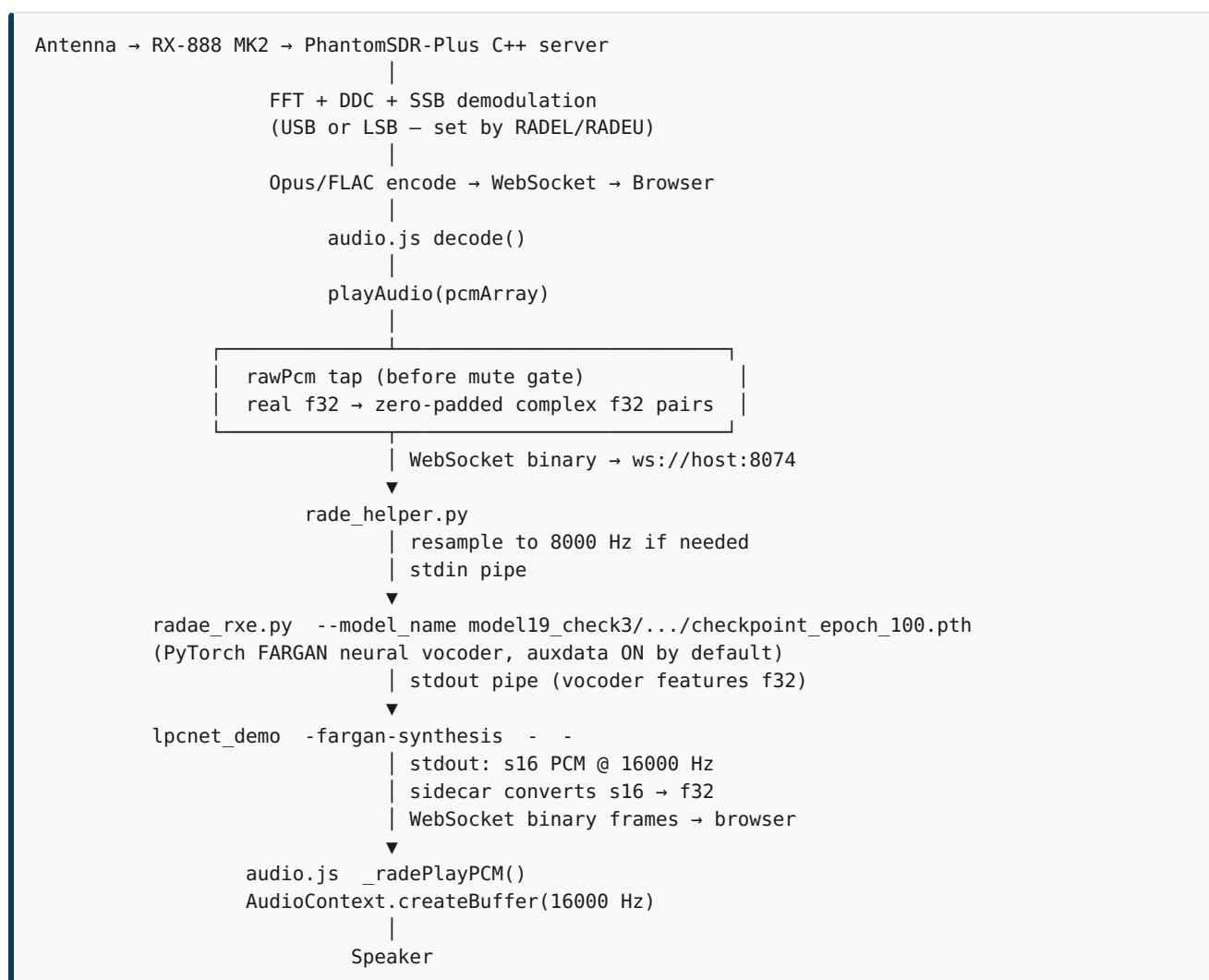
## Varijable okruženja

Varijabla	Zadano	Opis
<code>RADE_HELPER_PORT</code>	8074	TCP port na kojem sluša sidecar
<code>RADE_HELPER_HOST</code>	0.0.0.0	Adresa vezanja (127.0.0.1 iza proxyja)
<code>RADAE_DIR</code>	<code>~/radae</code>	Korijen repozitorija radae
<code>RADE_MODEL</code>	<code>RADAE_DIR/model19_check3/checkpoints/checkpoint_epoch_100.pth</code>	Težine modela
<code>LPCNET_DEMO</code>	<code>RADAE_DIR/build/src/lpcnet_demo</code>	Binarna datoteka lpcnet_demo
<code>RADE_AUXDATA</code>	1	Postavite na 0 da se <code>radae_rxe.py</code> proslijedi <code>--noauxdata</code>
<code>RADE_TORCH_THREADS</code>	1	PyTorch/OpenBLAS dretve po instanci <code>radae_rxe.py</code> — ograničava potrošnju CPU-a
<code>RADE_PIN_CORES</code>	1	Veže svaki lanac dekodiranja uz vlastiti skup jezgri; 0 isključuje vezanje
<code>RADE_CORES_PER_CLIENT</code>	2	Jezgri dodijeljenih po klijentu kada je vezanje uključeno

## Promijenjene datoteke

Datoteka	Vrsta	Napomene
rade_helper.py	Nova	Python sidecar: WebSocket poslužitelj + dvoprocesni lanac dekodiranja
frontend/src/audio.js	Izmijenjena	4 zakrpe
frontend/src/App.svelte	Izmijenjena	6 zakrpa
rade.sh	Novo	Upravljačka skripta RADE sidecara za samostalnu upotrebu (start/stop/restart/status)
start-rx88mk2.sh	Izmijenjeno	pokreće, nadzire i javlja stanje sidecara; isto vrijedi za start-airspyhf.sh, start-rtl.sh, start-rspla.sh, start-fobos-hf.sh, start-fobos.sh, start-hackrf.sh
stop-websdr.sh	Izmijenjeno	zaustavlja sidecar zajedno s poslužiteljem
spectrumserver.cpp	Nepromijenjena	Nisu potrebne izmjene u C++-u

## Sažetak toka signala



## Rješavanje problema

### Crvena traka — „Sidecar not reachable“

```
Is sidecar running?
ps aux | grep rade_helper | grep -v grep

Start it manually for testing
python3 ~/PhantomSDR-Plus/rade_helper.py &

Check port is open externally – use portchecker.co NOT curl from the server
(curl from the server uses NAT loopback and gives false "Connection refused")
```

### Port 8074 prikazuje se zatvorenim na portchecker.co unatoč iptables pravilu

Možda pružatelj filtrira port ili nedostaje NAT pravilo na usmjerivaču. Mogućnosti:

1. Isprobajte drugi port: `RADE_HELPER_PORT=8080 python3 rade_helper.py`
2. Prosljedite promet kroz postojeća javna vrata pomoću Nginxa (vidi [7. korak ručnog vodiča](#))

### `radae_rxe.py: error: unrecognized arguments: model_path`

Putanja modela mora se predati kroz `--model_name`, a ne kao pozicijski argument:

```
WRONG – positional argument
python3 radae_rxe.py model19_check3/checkpoints/checkpoint_epoch_100.pth

CORRECT – named argument
python3 radae_rxe.py --model_name model19_check3/checkpoints/checkpoint_epoch_100.pth
```

### `ModuleNotFoundError: No module named 'matplotlib'`

```
pip3 install matplotlib
```

`radae_rxe.py` bezuvjetno uvozi `matplotlib` na vrhu datoteke.

### `ModuleNotFoundError: No module named 'torch'`

```
pip3 install torch
```

### `lpcnet_demo: No such file or directory`

Izgradnja `radae` nije dovršena. Izgradite ponovno:

```
cd ~/radae/build
cmake .. && make -j$(nproc)
ls src/lpcnet_demo # should exist now
```

### Pogreška `size mismatch u inference.sh`

`model19_check3` traži `--auxdata` pri kodiranju pomoću `inference.sh`:

```
./inference.sh model19_check3/checkpoints/checkpoint_epoch_100.pth \
wav/brian_g8sez.wav /dev/null \
--rate_Fs --pilots --pilot_eq --eq_ls --cp 0.004 \
--write_rx rx.f32 --auxdata # ← required for model19_check3
```

Napomena: pri **dekodiranju** pomoću `radae_rxe.py` `--auxdata` je zadan — nemojte ga predavati.

## Žuti pokazatelj — „Searching for signal” — nikad se ne sinkronizira

- Potvrdite ispravan bočni pojas: RADEL za  $\leq 10$  MHz, RADEU za  $> 10$  MHz
- Provjerite na [qso.freedv.org](https://qso.freedv.org) odašilje li neka postaja u tom trenutku
- RADE v1 koristi 30 nosilaca u 1500 Hz širine — u slapu se pojavljuje kao zbijena skupina
- Predvidite do 1,5 sekunde za uspostavu

## underrun!!! iz aplaya tijekom testa lanca (Korak 3)

Očekivano samo pri izvanmrežnom testiranju s datotekama. `radae_rxe.py` obrađuje sporije od datotečnog U/I-ja, pa audio međuspremnik ostaje prazan. Pri prijemu uživo to se ne događa jer preglednik dovodi zvuk u stvarnom vremenu.

### ConnectionClosedError: received 1011 (internal error)

Sidecar je prihvatio WebSocket vezu, ali se interno srušio prije nego što je odgovorio. Uzrok je nekompatibilna verzija `rade_helper.py` — starija je verzija pokušavala predati `asyncio StreamReader` kao standardni ulaz podprocesa, što tiho ne uspijeva i zatvara vezu s pogreškom 1011.

**Rješenje:** zamijenite `rade_helper.py` aktualnom verzijom iz skupa zakrpa. Aktualna verzija koristi `os.pipe()` za međuprocensnu cijev i kompatibilna je s `websockets`om od 10.x do 16.x+.

## Više istodobnih korisnika

Svaka veza preglednika pokreće vlastiti neovisni par `radae_rxe.py` + `lpcnet_demo`, pa se svaki korisnik može slobodno ugoditi na drugu frekvenciju.

Prema zadanom, `radae_rxe.py` koristi **sve dostupne CPU jezgre** za matrične operacije PyTorch-a, uzrokujući potrošnju CPU-a od ~900 % po instanci. `rade_helper.py` to ograničava s dvije optimizacije:

1. `OMP_NUM_THREADS=1` (i odgovarajuće MKL/OpenBLAS varijable) — ograničava PyTorch na 1 dretvu
2. Funkcije pretvorbe zvuka vektorizirane numpyjem — uklanjaju trošak Python petlji

Rezultat: svaka instanca troši otprilike **8-10 %** jedne jezgre, dovoljno za dekodiranje RADE-a u stvarnom vremenu. Ako čujete ispade zvuka, povećajte na 2 dretve:

```
RADE_TORCH_THREADS=2 ./start-rx888mk2.sh # or your receiver's launcher
standalone: RADE_TORCH_THREADS=2 ./rade.sh restart
```

Istodobnih korisnika RADE-a	Približno opterećenje CPU-a
1	~9 %
5	~45 %
10	~90 %
20	~180 %

Na i5-12450H (12 dretvi), gdje spectrumserver troši ~42 %, a rx888\_stream ~11 %, imate otprilike **1000 % rezerve** — dovoljno za **20 i više istodobnih** korisnika RADE-a prije nego što CPU postane problem.

**Gornja tablica pretpostavlja da trošak CPU-a raste linearno s brojem korisnika. Mjerenje govori da nije tako.** Pokretanje `rade_loadtest.py` na tom istom i5-12450H pokazalo je trošak CPU-a po slušatelju znatno ispod ove tablice do ~24 slušatelja, a zatim oštar rast, pri čemu je računalo zaustavljeno **temperaturom kućišta procesora pri 32 slušatelja** — ne CPU-om ni propusnošću. Te brojke uzmite tek kao grubu smjernicu za male brojeve i izmjerite vlastito računalo: vidi [Mjerenje istodobnosti na vlastitom hardveru](#). Imajte na umu da test opterećenja lanac hrani šumom, a ne stvarnim RADE signalom, pa apsolutna potrošnja CPU-a uz stvaran promet može biti drukčija; pouzdan je *oblik* krivulje.

## Mjerenje istodobnosti na vlastitom hardveru

Gornje brojke potječu s jednog određenog računala. `rade_loadtest.py` (u korijenu repozitorija) mjeri isto na **vašem** — u koracima povećava broj sintetičkih RADE slušatelja prema sidecaru i javlja posljednji broj slušatelja koji je vaše računalo izdržalo prije nego što je nešto popustilo.

Nakon svakog koraka pusti računalo da se smiri, zatim uzorkuje temperaturu kućišta procesora, ukupni CPU, dostupni RAM, swap i zbrojeni RSS svih procesa `radae_rxe.py` i `lpcnet_demo`. Zaustavlja se na prvom **koljenu** — koje god ograničenje prvo bude prekoračeno — i ispisuje posljednji stabilan broj.

### Preduvjeti

```
sudo apt install python3-websockets python3-psutil
```

Pokrenite ga **na računalu sa SDR-om** (čita lokalne senzore i memoriju procesa), uz već pokrenute spectrumserver i RADE sidecar:

```
cd ~/PhantomSDR-Plus
python3 rade_loadtest.py
```

Ctrl-C u svakom trenutku uredno zatvara sve veze.

### Prije prvog pokretanja

Otvorite datoteku i provjerite dva bloka na vrhu:

- **CONFIG** — `WS_URL` je prema zadanom `ws://127.0.0.1:8074`; promijenite ga ako ste sidecar premjestili sa zadanog porta. `RADE_SPS` (12000) i `RADE_SIDE BAND` (USB) trebaju odgovarati onome što javlja vaše sučelje.
- **handshake\_messages()** — jedini `init` okvir koji šalje mora odgovarati inicijalizaciji RADE-a u `frontend/src/audio.js`. Ako se to rukovanje razišlo, svaki se slušatelj ne uspije spojiti i test odmah završava neuspjelim vezama.

### Pragovi koljena

Podesite ih u `CONFIG` po želji — namjerno su konzervativni:

Prag	Zadano	Zaustavlja kada
TEMP_KNEE_C	85.0	Temperatura kućišta procesora dosegne tu granicu
MIN_AVAIL_MB	800	Dostupni RAM padne ispod te vrijednosti
SWAP_GROWTH_MB	50	Swap naraste iznad polazne vrijednosti uzete pri pokretanju
FAIL_LIMIT	3	Toliko se slušatelja ne uspije spojiti u jednom koraku
MAX_LISTENERS	60	Tvrdo zaustavljanje čak i ako koljeno nikad ne nastupi

Oblik porasta određuju STEP (slušatelja dodanih po koraku, zadano 2), SETTLE\_S (25 s smirivanja prije uzorkovanja — to omogućuje nakupljanje topline) i SAMPLE\_S (petosekundni prozor usrednjavanja CPU-a). Potpuni prolaz do zadane granice od 60 slušatelja traje stoga otprilike 15 minuta.

## Čitanje rezultata

Svaki korak ispisuje redak i dodaje zapis u `rade_loadtest.csv`:

Stupac	Značenje
listeners	Sintetički slušatelji spojeni u ovom koraku
connect_fails	Slušatelji koji nisu uspjeli uspostaviti ili zadržati vezu
pkg_c	Temperatura kućišta procesora, medijan 5 uzoraka (otporan na skokove)
cpu_pct	Postotak CPU-a <b>cijelog sustava</b> , ne po slušatelju
avail_mb	Dostupni RAM
swap_mb	Zauzeti swap
dec_rss_mb	Zbrojeni RSS svih procesa dekodiranja
dec_procs	Broj procesa dekodiranja — očekujte <b>2 po slušatelju</b> ( <code>radex_rxe.py</code> + <code>lpcnet_demo</code> )

## Izmjereni prolaz

Na gore spomenutom i5-12450H potpuni je prolaz završio ovako:

```
KNEE at n=32: temp 91.0°C ≥ 85.0
Last stable concurrency: 30 RADE listeners
```

**Toplina je bila ograničavajući čimbenik**, i to uvjerljivo. Na koljenu je još bilo 7004 MB dostupnog RAM-a — 6,2 GB iznad granice MIN\_AVAIL\_MB — i nula neuspjelih veza u svakom koraku.

Dva stupca vrijedi pažljivo pročitati:

**dec\_rss\_mb precjenjuje stvarni trošak memorije.** Raste vrlo ravnomjerno, 295 MB po slušatelju (292, 294, 294 ... 295 kroz svih 16 koraka), ali *dostupni* RAM pada samo oko **202 MB po slušatelju**. Dva procesa iza svakog slušatelja dijele stranice biblioteka, a RSS te stranice broji jednom po procesu. Ekstrapolacijom nagiba avail\_mb, memorijsko bi koljeno nastupilo blizu **63 slušatelja** — iza tvrdog zaustavljanja na 60, pa se na ovom računalu nikad ne može aktivirati. Memoriju dimenzionirajte prema avail\_mb, a ne prema dec\_rss\_mb.

**cpu\_pct je ravan — dok odjednom nije.** Do 24 slušatelja stoji na ~5,3 % cijelog računala, a zatim postaje nadlinearan:

Slušatelja	24	26	28	30	32
cpu_pct	6,7 %	18,4 %	30,1 %	49,6 %	65,2 %

To je deseterostruki porast CPU-a dok broj slušatelja raste za trećinu. Točka pregiba reproducira se na istom mjestu kroz više prolaza, pa je tretirajte kao svojstvo računala, a ne kao šum — trošak CPU-a po slušatelju iznosi otprilike 3,4 % jedne jezgre ispod pregiba i oko 20 % iznad njega. Nemojte ekstrapolirati vrijednost CPU-a po korisniku izmjerenu pri malom broju slušatelja.

**Test nema koljeno po CPU-u.** Četiri su granice temperatura, dostupni RAM, rast swapa i neuspjele veze — CPU se bilježi, ali nikad ne prekida prolaz. U gornjem je prolazu CPU dosegao 65 % i nastavio bi rasti da nije prva okinula temperatura. Ako vam je važna gornja granica CPU-a, sami pratite taj stupac ili dodajte prag.

**Što ovo dokazuje, a što ne.** Svaki sintetički slušatelj šalje slučajan šum male amplitude, a ne stvaran RADE signal. To je dovoljno da cijeli lanac dekodiranja radi i troši resurse, pa su brojke o resursima smislene — ali test ništa ne govori o *kvaliteti* dekodiranja ni o neprekidnosti zvuka pod opterećenjem. To potvrdite sa stvarnim slušateljima na stvarnom signalu.

## Ako se jezgre zasite prije koljena

RADE čvrsto veže jednu dretvu po instanci, pa pojedinačne jezgre mogu doseći ~90 °C dok ukupni CPU još izgleda neopterećeno. `rade_helper.py` lance raspoređuje po jezgrama kružno kako bi to ublažio. Ako prolaz pokazuje rana toplinska koljena, proširite dodjelu jezgri po klijentu:

```
RADE_CORES_PER_CLIENT=3 ./rade.sh restart
```

Nije potrebno uređivati datoteke — riječ je o varijabli okruženja (vidi [Varijable okruženja](#)); `RADE_PIN_CORES=0` posve isključuje vezanje.

## Ažuriranje RADE v1

Integracija s PhantomSDR-Plusom (`rade_helper.py`, zakrpe sučelja) samo je most — sva logika dekodiranja RADE-a nalazi se u repozitoriju `radae`. Ažuriranja su stoga gotovo uvijek jednostavan `git pull` uz ponovnu izgradnju, bez ikakvih izmjena u samom PhantomSDR-Plusu.

### Uobičajeno ažuriranje (novi kod, isti model)

```
1. Pull latest radae code
cd ~/radae
git pull

2. Rebuild lpcnet_demo (in case C code changed)
cd build
cmake ..
make -j$(nproc)

3. Restart the sidecar – no server restart needed
cd ~/PhantomSDR-Plus
./rade.sh restart
```

To je sve. Bez ponovne izgradnje sučelja, bez ponovnog pokretanja poslužitelja i bez uređivanja datoteka.

### Samo nove težine modela

Ako izađe nova kontrolna točka (npr. `model20`) bez izmjena koda, usmjerite sidecar na nove težine putem varijable okruženja — uređivanje datoteka nije potrebno:

```
One-off: start with new model
RADE_MODEL=~/.radeae/model20/checkpoints/checkpoint_epoch_100.pth ./rade.sh start

Or permanently – add to your shell profile (~/.bashrc):
export RADE_MODEL=~/.radeae/model20/checkpoints/checkpoint_epoch_100.pth
```

## Što koja vrsta promjene zahtijeva

Što se promijenilo u radae	Potrebna radnja
Nove težine modela (nova .pth datoteka)	Postaviti varijablu RADE_MODEL, ./rade.sh restart
Izmjena koda u radae_rxe.py	git pull, ./rade.sh restart
Promijenjen C kod lpcnet_demo	git pull, ponovna izgradnja, ./rade.sh restart
radae_rxe.py preimenovan ili premješten	Ažurirati putanju RADAЕ_RX u rade_helper.py (jedan redak)
Argument --model_name preimenovan	Ažurirati radae_cmd u rade_helper.py (jedan redak)
Nova izlazna frekvencija uzorkovanja (≠ 16000 Hz)	Ažurirati SPS_OUT u rade_helper.py i createBuffer() u audio.js
RADE v2 koristi drugu binarnu datoteku	Ažurirati RADAЕ_RX u rade_helper.py (jedan redak)

## Provjera je li ažuriranje uspjelo

Nakon ponovnog pokretanja provjerite radi li novi kod:

```
Check sidecar picked up new radae_rxe.py
./rade.sh status

Tail log to see startup lines
tail -20 ~/PhantomSDR-Plus/rade.log
```

Zapisnik bi trebao prikazivati očekivanu putanju modela:

```
[RADE] model : /home/sv1btl/radae/model19_check3/checkpoints/checkpoint_epoch_100.pth
```

## Kako pratiti novosti

Pretplatite se na stranicu izdanja repozitorija radae kako biste primali obavijesti o novim modelima i ažuriranjima koda:

```
https://github.com/drowe67/radae/releases
```

Pratite FreeDV blog za najave novih RADE valnih oblika i modela:

```
https://freedv.org/blog/
```

Razvijeno i testirano na PhantomSDR-Plusu (fork sv1btl/PhantomSDR-Plus) s RX-888 MK2 na Ubuntuu 24.04. Poslužitelj u C++-u nije mijenjan.

RADE razvijaju David Rowe VK5DGR i FreeDV tim. Vidi [freedv.org/radio-autoencoder](https://freedv.org/radio-autoencoder).

## Potpuna deinstalacija i čista ponovna instalacija

Evo potpunog uklanjanja prije ponovnog pokretanja `install_rade.sh`:

1. Zaustavite sidecar `cd ~/PhantomSDR-Plus && ./rade.sh stop`
2. Uklonite repozitorij `radae` i izgradnju `rm -rf ~/radae`
3. Uklonite `torch` (instaliran u korisničkom direktoriju) `pip3 uninstall -y torch` `rm -rf ~/.local/lib/python3.11/site-packages/torch*`
4. Uklonite Python pakete iz `apt` (neobavezno — preskočite ako ih koristi nešto drugo) `sudo apt-get remove -y python3-numpy python3-scipy python3-matplotlib python3-websockets` `sudo apt-get autoremove -y`
5. Provjerite je li sve uklonjeno `python3 -c "import torch" 2>&1 # should say ModuleNotFoundError` `ls ~/radae 2>&1 # should say No such file or directory`
6. Čista instalacija `chmod +x ~/PhantomSDR-Plus/install_rade.sh` `~/PhantomSDR-Plus/install_rade.sh`

## POGLAVLJE 10

# RADE opća instalacija (Linux)

RADE\_General\_INSTALL\_MANUAL\_LINUX.md

## RADE v1 — Vodič za ručnu instalaciju

### Svi Linuxi — Ubuntu / Debian / Fedora / Arch / Raspberry Pi OS · PhantomSDR-Plus

**Preduvjet:** zakrpane datoteke (`rade_helper.py`, `rade.sh`, `audio.js`, `App.svelte`) već se nalaze u stablu direktorija PhantomSDR-Plusa. Ovaj vodič gradi sve ostalo oko njih.

#### VAŽNO

**Za normalnu instalaciju ovaj vodič vam ne treba.** `./install_rade.sh` u mapi PhantomSDR-Plus obavlja svaki korak u nastavku umjesto vas — systemske pakete, Python module, građenje radaea, provjeru težina modela i pokretanje sidecara — i radi jednako na apt, pacman, dnf i zypper sustavima. `./install.sh` i četiri distribucijske skripte nude da je pokrenu u sklopu normalne instalacije. Slijedite ovaj vodič samo za ručno postavljanje, za sustav koji skripta ne pokriva ili za ručni popravak pojedinog koraka.

**Korisnici Raspberry Pija:** ovo je i vaš vodič. Raspberry Pi OS je Debian, pa svugdje slijedite blokove za Debian i pročitajte napomene **Raspberry Pi / Bookworm** ondje gdje se pojavljuju — pokrivaju dvije stvari koje se na Piju razlikuju: PEP 668 i torch kotačić samo za CPU.

## Korak 1 — Sistemski paketi

Odaberite blok koji odgovara vašoj distribuciji.

### Ubuntu / Debian / Raspberry Pi OS

```
sudo apt update
sudo apt install -y \
 build-essential cmake git \
 python3 python3-pip \
 nodejs npm \
 alsa-utils
```

### Fedora / RHEL / Rocky

```
sudo dnf install -y \
 gcc gcc-c++ cmake git \
 python3 python3-pip \
 nodejs npm \
 alsa-utils
```

### Arch / Manjaro

```
sudo pacman -Sy --needed \
 base-devel cmake git \
 python python-pip \
 nodejs npm \
 alsa-utils
```

## Provjera verzije Node.js-a (sve distribucije)

Izgradnja RADE sučelja zahtijeva Node.js 22 ili noviji:

```
node --version
```

Ako je niža od 22.x, instalirajte ga pomoću `nvm`-a (bilo koja distribucija, bez root ovlasti):

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.4/install.sh | bash
export NVM_DIR="$HOME/.nvm"
[-s "$NVM_DIR/nvm.sh"] && \. "$NVM_DIR/nvm.sh"
nvm install 22 && nvm use 22 && nvm alias default 22
```

**Ne putem NodeSourcea.** `deb.nodesource.com` i `rpm.nodesource.com` sada na svakoj putanji repozitorija vraćaju HTTP 403, pa stara naredba `curl -fsSL https://deb.nodesource.com/setup_NN.x | sudo -E bash -` više ne radi — a na Debianu/Ubuntuu za sobom ostavlja `apt` izvor koji ruši svaki sljedeći `apt update`. Iz istog razloga instalacijske skripte koriste `nvm`.

## Korak 2 — Python paketi

### Ubuntu 24.04 i stariji / Fedora / Arch

Na tim sustavima radi obična naredba `pip3 install`:

```
pip3 install websockets matplotlib numpy scipy

torch — CPU-only build (~150–250 MB, avoids the ~3 GB CUDA wheel)
Use this if the server has no GPU, which is the typical case for a WebSDR
pip3 install torch --index-url https://download.pytorch.org/whl/cpu
```

Ako vaš poslužitelj **ipak** ima NVIDIA GPU s instaliranom CUDA-om, možete izostaviti `--index-url` i dobiti puno CUDA izdanje. Za RADE to ne donosi nikakvu dobit u performansama — `radae_rxe.py` koristi PyTorch samo za matrične operacije na procesoru.

### Ubuntu 23.04+ / Debian Bookworm+ / Raspberry Pi OS (sustavi s PEP-om 668)

Te distribucije blokiraju голу naredbu `pip3 install`. Dodajte zastavicu:

```
All packages except torch
pip3 install --break-system-packages websockets matplotlib numpy scipy

torch — CPU-only build
pip3 install --break-system-packages torch \
 --index-url https://download.pytorch.org/whl/cpu
```

#### Raspberry Pi / Bookworm — dva pravila koja grizu:

1. Svaka naredba `pip3 install` traži `--break-system-packages` (provedba PEP-a 668). Bez toga je instalacija potpuno blokirana.
2. Obična naredba `pip3 install torch` preuzima CUDA izdanje (~3 GB). Na Piju nema CUDA-e — koristite indeks samo za CPU kako biste dobili lagano ARM64 izdanje (~150 MB).

Alternativno, upotrijebite virtualno okruženje i posve izbjegnite tu zastavicu:

```
python3 -m venv ~/rade-venv
source ~/rade-venv/bin/activate
pip install websockets matplotlib numpy scipy torch \
 --index-url https://download.pytorch.org/whl/cpu
```

Ako koristite venv, svim sljedećim pozivima `python3` u ovom vodiču prethodite naredbom `source ~/rade-venv/bin/activate` ili koristite punu putanju `~/rade-venv/bin/python3`.

## Provjera

```
python3 -c "import websockets, matplotlib, torch, numpy, scipy; print('All OK!')"
```

Očekivani ispis:

```
All OK
```

## Korak 3 — Kloniranje i izgradnja repozitorija radae

RADE dekodier nalazi se u repozitoriju odvojenom od codec2. `freedv_rx` iz codec2 **ne** podržava RADE v1.

```
Remove any previous incomplete clone
rm -rf ~/radae

Clone
git clone https://github.com/drowe67/radae.git ~/radae
cd ~/radae

Build
mkdir build && cd build
cmake ..
make -j$(nproc)
```

### Provjerite je li `lpcnet_demo` izgrađen

```
ls -la ~/radae/build/src/lpcnet_demo
```

Očekivano: binarna datoteka postoji i izvršna je.

### Provjerite postoje li težine modela

```
ls ~/radae/model19_check3/checkpoints/checkpoint_epoch_100.pth
```

Očekivano: `.pth` datoteka je navedena. Težine dolaze uz repozitorij — nije potrebno zasebno preuzimanje.

## Korak 4 — Provjera lanca dekodiranja

Ovaj korak potvrđuje da cijeli izvanmrežni lanac radi prije nego što ga povežete s preglednikom. Pokrenite iz `~/radae`:

```
cd ~/rade
```

#### 4a — Stvorite ispitni signal kodiran RADE-om

```
./inference.sh model19_check3/checkpoints/checkpoint_epoch_100.pth \
wav/brian_g8sez.wav /dev/null \
--rate_Fs --pilots --pilot_eq --eq_ls --cp 0.004 \
--write_rx rx.f32 --auxdata
```

Pričekajte da završi. Posljednji ispisani redci trebali bi biti:

```
loss: 0.741 Auxdata BER: 0.012
```

#### 4b — Dekodirajte i reproducirajte

```
cat rx.f32 \
| python3 rade_rxe.py \
--model_name model19_check3/checkpoints/checkpoint_epoch_100.pth \
| ./build/src/lpcnet_demo -fargan-synthesis - - \
| aplay -f S16_LE -r 16000
```

Trebali biste čuti glas. Ispis prikazuje uspostavu sinkronizacije:

```
1 state: search ...
5 state: sync ... SNRdB: 4.03 uw_err: 0
Playing raw data 'stdin' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
```

**Poruke `underrun!!!` tijekom ovog testa očekivane su i bezopasne.** Pojavljuju se jer `radae_rxe.py` obrađuje sporije od datotečnog U/I-ja. Pri prijemu uživo ne javljaju se — preglednik dovodi zvuk u stvarnom vremenu.

#### 4c — Provjerite pokreće li se sidecar ispravno

```
python3 ~/PhantomSDR-Plus/rade_helper.py
```

Očekivani ispis (bez WARNING redaka):

```
[RADE] helper starting on ws://0.0.0.0:8074
[RADE] rade_rx.py : /home/<user>/rade/rade_rxe.py
[RADE] model : /home/<user>/rade/model19_check3/checkpoints/checkpoint_epoch_100.pth
[RADE] lpcnet_demo : /home/<user>/rade/build/src/lpcnet_demo
[RADE] auxdata : ON (default)
[RADE] torch threads : 1 per instance (RADE_TORCH_THREADS to override)
[RADE] architecture : per-connection (each user tunes independently)
[RADE] listening : waiting for PhantomSDR-Plus clients
```

Pritisnite **Ctrl+C** za prekid.

`<user>` u gornjim putanjama je račun s kojim ste prijavljeni — na standardnoj slici Raspberry Pi OS-a to su `/home/pi/rade/...`

## Korak 5 — Izgradnja sučelja PhantomSDR-Plusa

```
cd ~/PhantomSDR-Plus/frontend

Only if node_modules is missing:
npm install

cd ~/PhantomSDR-Plus
./recompile.sh
```

Pripazite na pogreške Vite/acorn raščlanjivača. Zakrpane datoteke namjerno izbjegavaju `?.`, `??` i prazne `catch {}` kako bi zadovoljile ograničenje acorna.

## Korak 6 — Upravljanje sidecarom

**Obično vam ne treba zasebna upravljačka skripta za RADE.** Skripte za pokretanje (`start-rx88mk2.sh`, `start-airspyhf.sh`, `start-rtl.sh`, `start-rspla.sh`, `start-fobos-hf.sh`, `start-fobos.sh`, `start-hackrf.sh`) same pokreću sidecar čim poslužitelj proradi, ponovno ga pokreću ako se ugasi i javljaju njegovo stanje; `stop-websdr.sh` zaustavlja ga zajedno s poslužiteljem. Idite izravno na 7. korak.

**`rade.sh` i dalje dolazi uz projekt za samostalnu upotrebu — pokretanje sidecara bez PhantomSDR-Plus pokretača ili njegovo zasebno testiranje. Njegove naredbe, dnevnik i pripadajuća upozorenja o čuvaru nalaze se u Upravljanje sidecarom.**

## Korak 7 — Otvorite port 8074

Sidecar sluša na TCP portu **8074**. Preglednik se spaja izravno na taj port. Morate ga otvoriti ručno.

### ufw (Ubuntu / Debian / Raspberry Pi OS)

```
sudo ufw allow 8074/tcp && sudo ufw reload
```

### firewalld (Fedora / RHEL / Rocky)

```
sudo firewall-cmd --add-port=8074/tcp --permanent
sudo firewall-cmd --reload
```

### iptables (bilo koja distribucija, trajno)

```
sudo iptables -I INPUT -p tcp --dport 8074 -j ACCEPT

Ubuntu / Debian / Raspberry Pi OS – persist across reboots:
sudo apt install iptables-persistent
sudo netfilter-persistent save

Fedora / RHEL – persist:
sudo service iptables save
```

## Usmjerivač

Dodajte pravilo NAT-a / prosljeđivanja porta: **TCP 8074 → LAN IP poslužitelja : 8074**

## Testiranje izvana

Upotrijebite **https://portchecker.co** i provjerite port 8074 prema svojem javnom nazivu poslužitelja.

**Nemojte** testirati naredbom `curl` sa samog poslužitelja — NAT hairpin daje lažne rezultate „Connection refused“ čak i kad je port otvoren.

## Alternativa — Nginx proxy (ako pružatelj blokira port 8074)

Dodajte unutar postojećeg bloka `server {}`:

```
location /rade {
 proxy_pass http://127.0.0.1:8074;
 proxy_http_version 1.1;
 proxy_set_header Upgrade $http_upgrade;
 proxy_set_header Connection "upgrade";
 proxy_set_header Host $host;
 proxy_read_timeout 3600s;
}
```

```
sudo nginx -t && sudo nginx -s reload
```

Zatim uredite `audio.js` unutar funkcije `setRADEDecoding()`:

```
// Change:
var helperUri = uri || ('ws://' + window.location.hostname + ':8074');
// To:
var helperUri = uri || ('ws://' + window.location.host + '/rade');
```

Nakon te izmjene ponovno izgradite sučelje (`./recompile.sh`).

## Korak 8 — Pokrenite poslužitelj

Pokrenite PhantomSDR-Plus kao i inače. RADE se **ne** pokreće automatski:

```
cd ~/PhantomSDR-Plus
./start-rx888mk2.sh # or start-airspyhf.sh,
 # start-rtl.sh, start-rspla.sh,
 # start-fobos-hf.sh, start-fobos.sh,
 # start-hackrf.sh
```

Upotrijebite pokretač koji odgovara vašem prijamniku. Svaki ima vlastiti watchdog i zapisnik; `./stop-websdr.sh` zaustavlja onaj koji radi.

## Korak 9 — Pokrenite RADE

```
cd ~/PhantomSDR-Plus
./rade.sh start
./rade.sh status
tail -f rade.log
```

Očekivani zapisnik:

```
[RADE] sidecar starting at ...
[RADE] helper starting on ws://0.0.0.0:8074
[RADE] rade_rx.py : /home/<user>/rade/rade_rxe.py
[RADE] model : /home/<user>/rade/model19_check3/checkpoints/checkpoint_epoch_100.pth
[RADE] lpcnet_demo : /home/<user>/rade/build/src/lpcnet_demo
[RADE] auxdata : ON (default)
[RADE] torch threads : 1 per instance
[RADE] architecture : per-connection (each user tunes independently)
[RADE] listening - waiting for PhantomSDR-Plus clients
```

## Korak 10 — Korištenje RADE-a u pregledniku

1. Otvorite svoje web-sučelje PhantomSDR-Plusa
2. Aktivne postaje potražite na <https://qso.freedv.org>
3. Ugodite se na prikazanu frekvenciju postaje
4. U **Decoder Options** odaberite:
5. **RADE v1 — RADEL (LSB)** za 40 m / 80 m / 160 m ( $\leq 10$  MHz)
6. **RADE v1 — RADEU (USB)** za 20 m / 17 m / 15 m / 12 m / 10 m ( $> 10$  MHz)
7. Kliknite **Decoder: ON**

Korake 4 i 5 može zamijeniti jedan pritisak: gumbi **RADEL / RADEU** nalaze se uz naslov **Modes selector** te u skočnim prozorima **Modes** i **Bands**. Jedan pritisak odabire dekode, uključuje ga i dovodi RADE ploču na zaslon; ponovni pritisak isključuje RADE. Pogledajte [priručnik za RADE](#).

Pokazatelj	Značenje
Crveno — „Connecting to sidecar...”	Port 8074 nedostupan ili sidecar ne radi
Žuto — „Searching for signal...”	Sidecar spojen, RADE okvir još nije otkriven (predvidite ~1,5 s)
Zeleno — „Synced · SNR x.x dB”	Dekodiranje — govor se reproducira

## Napomena o procesoru

Svaki korisnik RADE-a troši oko 8–10 % jedne jezgre (sidecar ograničava broj PyTorch dretvi na 1). Ako čujete ispade zvuka, povećajte na 2 dretve:

```
RADE_TORCH_THREADS=2 ./rade.sh restart
```

Istodobnih korisnika	Približno opterećenje CPU-a
1	~9 %
5	~45 %
10	~90 %
20	~180 %

Ove su brojke izmjerene na x86\_64. Raspberry Pi bez problema dekodira RADE, ali je trošak po korisniku veći i gornja se tablica ne prenosi — izmjerite vlastitu pločicu naredbom `top` dok je korisnik sinkroniziran, prije nego što objavite ograničenje broja korisnika.

## Buduće ažuriranje RADE-a

```
cd ~/rade
git pull
cmake -S . -B build && make -C build -j$(nproc)

cd ~/PhantomSDR-Plus
./rade.sh restart
```

Ponovna izgradnja sučelja ili ponovno pokretanje poslužitelja nisu potrebni osim ako se promijenio sam `rade_helper.py`.

*Testirano na Ubuntu 24.04 (x86\_64) i Raspberry Pi 4 / Raspberry Pi OS Bookworm (ARM64, Python 3.11). PhantomSDR-Plus fork: [sv1bt/PhantomSDR-Plus](https://github.com/sv1bt/PhantomSDR-Plus). RADE su razvili David Rowe VK5DGR i FreeDV tim — <https://freedv.org>*

## POGLAVLJE 11

# Uređivanje varijanti

EDITING\_VARIANTS.md

## PhantomSDR-Plus — Kako uređivati varijante frontenda

(nakon što su četiri datoteke `App__*_smeter_.svelte` spojene u jednu `App.svelte`)

### 1. Što su varijante

Dva neovisna izbora koji daju četiri kombinacije:

```
smeter = "analog" instrument s pomičnom kazaljkom
 = "digital" segmentirani stupčasti instrument

layout = "v1" izbornik načina rada gore, izbornik pojasa dolje
 = "v2" izbornik pojasa gore, izbornik načina rada dolje
 (pomiče i red gumba za fino ugađanje na mobitelu)
```

Sve četiri žive u **jednoj** izgradnji, koja se poslužuje na `/`. Birač  u gornjem desnom kutu prebacuje između njih unutar stranice koja radi: postavlja propove `smeter/layout` u `App.svelte`, pa se ništa ne učitava ponovno, a zvuk, slap i dekoderi nastavljaju raditi. Odabir se sprema po pregledniku u `localStorage` pod ključem `phantom.variant` i pri sljedećem posjetu ima prednost pred zadanom vrijednošću izgradnje.

Varijanta koju sysop odabere u `recompile.sh` stoga je **početna** varijanta — ono što posjetitelj vidi prvi put, prije nego dotakne izbornik.

URL	što je to
<code>/</code>	stranica za računalo; sve četiri varijante, zamjenjive u radu
<code>/mobile/</code>	<i>zasebna stranica, gradi je <code>build-mobile.sh</code> — nije varijanta</i>

Do 10. 8. 2026. svaka je varijanta bila zasebna izgradnja pod `/analog/`, `/digital/`, `/v2-analog/` i `/v2-digital/`, a birač je navigirao među njima — zato je prebacivanje ponovno učitavalo stranicu i prekidalo zvuk. Tih izgradnji više nema; na svakoj od četiri putanje ostaje samo mali `index.html` koji preusmjerava na `/`, a piše ga `frontend/make-redirect-stubs.sh` pri svakoj izgradnji kako bi stare oznake i dalje radile.

Varijanta se **ne** upisuje ni u jednu izvornu datoteku. `src/main.js` je nepromjenjiva ulazna točka koja komponenti predaje dvije vrijednosti definirane u vrijeme izgradnje:

```
const app = new App({
 target: document.getElementById('app'),
 props: { smeter: __PHANTOM_SMETER__, layout: __PHANTOM_LAYOUT__ }
})
```

`vite.config.js` ih popunjava pri izgradnji, uzimajući prvo od navedenog što je postavljeno:

1. **PHANTOM\_SMETER / PHANTOM\_LAYOUT** — varijable okoline. Otkad nema izgradnji po varijanti, ništa ih više ne postavlja; i dalje rade za jednokratnu izgradnju s drugom početnom varijantom.
2. **frontend/variant.json** — zadana vrijednost stranice, koju upisuje `recompile.sh`. Izgradnja korijena (`/`) ne prosljeđuje varijantu pa završava ovdje.
3. **analog / v1** — pričuvna vrijednost.

Te su dvije vrijednosti samo početna vrijednost. `App.svelte` ih pri inicijalizaciji prepisuje iz `localStorage` ako je posjetitelj već odabrao varijantu, a birač ☼ ih ponovno postavlja pri svakom prebacivanju.

Unutar `App.svelte` propovi se pretvaraju u dvije zastavice u retcima 20-21:

```
$: isAnalog = smeter !== "digital";
$: isV2 = layout === "v2";
```

Koristite te dvije zastavice — nikada `smeter/layout` izravno — kako bi se cijela datoteka čitala na isti način.

## 2. Koja datoteka što radi

### `frontend/src/main.js` — 12 redaka

Ulazna točka. Ništa osim instanciranja App-a i dviju `PHANTOM_*` vrijednosti koje prosljeđuje kao propove. **Nepromjenjiva** je — ni `recompile.sh` ni skripte za izgradnju je više ne prepisuju, a upravo to omogućuje istovremenu izgradnju varijanti.

**Ne stavljajte ovdje kod aplikacije.** To je ulazna datoteka, a ne mjesto za logiku.

### `frontend/src/App.svelte` — ~17130 redaka

Cijela aplikacija: slap, ugađanje, dekoderi, oznake, ploče, raspored za računalo i prilagodljivi raspored za mobitel. Gotovo svaka izmjena koju ćete ikada raditi ide ovdje.

### `frontend/src/lib/SMeterAnalog.svelte` — 824 retka

Instrument s pomičnom kazaljkom, u cijelosti: crtanje brojčanika, kazaljka, prebacivanje svijetlog/tamnog brojčanika i njegov ključ u `localStorage`, te zaglađivanje kazaljke.

- Ulazni prop: `dbm` — KALIBRIRANA snaga u dBm.
- Prima i: `mobile` — manji atributi canvasa za raspored na mobitelu.
- Podešavanje: `SMOOTH_TIME_MS` u retku 55. Manje = brža kazaljka, više = glađa. Trenutno 16 (jedan okvir pri 60 fps, praktički donja granica).

### Tri izgleda skale — i `smeter_theme.sh`

Analogni instrument ima tri pozadine, sve iscrtane istim kodom:

```
dark izvorni tamni brušeni metal
amber svijetla, blijedosiva skala
vintage topla skala u boji ostarjelog jantara
```

**Nemojte to brkati s varijantom `smeter = analog / digital` iznad.** Varijanta bira koja se komponenta instrumenta prikazuje; izgled bira kako se analogni iscrtava. Pohranjeni su pod različitim `localStorage` ključevima (`phantom.variant` i `smeterTheme`) i postavljaju se na različitim mjestima.

**Strana posjetitelja.** Platno instrumenta je gumb ( `role="button"` , također Enter/Razmaknica s tipkovnice, uz napomenu „click to switch style“ ispod). Svaki klik prolazi `dark` → `amber` → `vintage` i prema izbor u `localStorage.smeterTheme`. Čita se natrag pri svakom učitavanju stranice, pa izbor posjetitelja preživljava osvježavanja i ponovna pokretanja preglednika neograničeno. Namjerno **nije** kolačić: kolačić bi se slao poslužitelju uz svaki zahtjev za slapom i zvukom, za vrijednost koju poslužitelj nikada ne čita.

**Strana sysopa.** Dvije stvari u `SMeterAnalog.svelte` određuju što dobivaju svi ostali:

Redak	Što radi
<code>let smeterTheme = 'dark';</code>	Izgled koji dobiva preglednik <b>bez spremljenog izbora</b> .
<code>const SMETER_PREF_VERSION = 2;</code>	Brojač. Kada je spremljena verzija preglednika iza njega, preglednik <b>jednom</b> zaboravi spremljeni izgled i preuzme gornju zadanu vrijednost. Povećajte ga da pomaknete posjetitelje koji su već birali.

Promjena samo zadane vrijednosti stoga doseže isključivo posve nove posjetitelje. Obje promjene zajedno pomiču sve — upravo zato postoji `smeter_theme.sh` u korijenu repozitorija:

```
./smeter_theme.sh # prikaži trenutnu zadanu vrijednost, zatim odaberi iz izbornika
./smeter_theme.sh vintage # postavi odmah, pa ponudi ponovnu izgradnju
./smeter_theme.sh dark --build # pokreni frontend/build-all.sh bez pitanja
./smeter_theme.sh amber --no-reset # samo novi posjetitelji – postojeće ostavi na miru
```

Prijavljuje trenutnu zadanu vrijednost, postavlja novu, povećava brojač poništavanja (u uvijek prema gore — **nikada ga ne smanjujte**, manji broj samo spriječi da se poništavanje dogodi) i zatim nudi isti izbor izgradnje frontenda kao `recompile.sh`. **Ne** dira backend i **ne** prepisuje `variant.json`.

Promjena doseže spojenog slušatelja tek kad ponovno učitava svoju stranicu — zadana vrijednost je ugrađena u JS paket, a poništavanje se izvodi pri učitavanju. Nema slanja uživo. Obično ponovno učitavanje je dovoljno jer vite dodaje hash u imena datoteka paketa.

Jednokratno brisanje putem brojača nadjačava sve, uključujući posjetitelje koji su namjerno kliknuli vlastiti izgled. To je namjerno: to je jedini način da se cijela stranica vrati na jedan izgled. Upotrijebite `--no-reset` ako ih radije želite ostaviti na miru.

## frontend/src/lib/SMeterDigital.svelte — 184 retka

Segmentirani stupčasti instrument, u cijelosti.

- Ulazni propovi: `rawDb` — SIROVA vrijednost `audio.getPowerDb()`
- `smeterOffset` — `audio.smeter_offset`
- `mobile` — manji atributi canvasa

**NAPOMENA:** ovaj instrument namjerno se napaja SIROVOM snagom plus pomakom, a NE kalibriranom dBm vrijednošću koju koristi analogni brojačnik. Oduvijek je imao vlastito preslikavanje. Davanje kalibrirane vrijednosti pomaknulo bi svako očitavanje na svakoj digitalnoj postaji. Nemojte „ujednačavati“ ova dva ulaza.

## frontend/src/lib/BandSelector.svelte — 69 redaka

Mreža gumba pojaseva. Prikazuje se dvaput u `App.svelte` (jednom po rasporedu).

## frontend/src/lib/ModesSelector.svelte — 75 redaka

Mreža gumba načina rada. Prikazuje se dvaput u App.svelte.

## frontend/src/lib/StatusIndicators.svelte — 122 retka

Statusne žaruljice (mute, squelch, NR, NB, NS, AN, CTCSS). Prima prop wide: digitalni raspored koristi šire žaruljice od analognog. Obje Tailwind klase (w-8 i w-10) napisane su u cijelosti unutar komponente — Tailwindov skener vidi samo doslovna imena klasa, pa ih nikada ne gradite spajanjem znakovnih nizova.

Sve ostale datoteke u frontend/src/lib/ (Spectrogram, PassbandTuner, FrequencyInput, VersionSelector, ...) starije su od ovog posla i nisu mijenjane.

## 3. Izmjena koja utječe samo na neke varijante

Umotajte je u zastavicu. U oznakama:

```

{#if isAnalog}
 ...ovo vide samo analogne postaje...
{:else}
 ...ovo vide samo digitalne postaje...
{/if}

{#if isV2} ... {/if} {#if !isV2} ... {/if}

```

U atributu class:

```
class="p-4 {isAnalog ? 'text-xs' : 'text-sm'} rounded-md"
```

Napišite obje mogućnosti kao cjelovite doslovne nizove, kao gore. Tailwind pretražuje izvorni kod za potpunim imenima klasa; text-{size} ili spojeno ime uklanja se iz CSS-a i stil tiho nestaje.

U bloku <script>: pokušajte uopće ne granati. Skripta namjerno računa ulaze OBAJU instrumenata u svakom ciklusu —

```

smeterDbm = powerDb; // analogni brojčanik
smeterRawDb = audio.getPowerDb(); // digitalni brojčanik
smeterOffset = audio.smeter_offset; // digitalni brojčanik

```

— pa samo oznake moraju birati. Zato u datoteci od 17000 redaka postoji samo 15 prekidača varijanti. Neka tako i ostane; grananje na razini skripte je ono što je razdvojilo četiri stare datoteke.

## 4. Gdje su prekidači varijanti

Petnaest mjesta u App.svelte. Brojevi redaka pomiču se pri uređivanju — pouzdan način da ih sve pronađete je:

```
cd frontend/src && grep -n 'isAnalog\|isV2' App.svelte
```

U trenutku pisanja:

Redak	Zastavica	Što prebacuje
20		deklaracija <code>isAnalog</code>
21		deklaracija <code>isV2</code>
8062	<code>isV2</code>	računalo: izbornik pojasa prvi (poredak v2)
8090	<code>isV2</code>	računalo: veličina naslova ploče
8686	<code>isV2</code>	računalo: izbornik načina rada prvi (poredak v1)
8717	<code>isAnalog</code>	najmanja širina ploče instrumenta (kazaljka je uža)
8730	<code>isAnalog</code>	samo analogno: redak datuma/vremena iznad instrumenta
8819	<code>isAnalog</code>	samo analogno: statusne žaruljice + razdjelnik
8834	<code>isAnalog</code>	samo digitalno: redak vremena, široke žaruljice i sam izbor <code>&lt;SMeterDigital&gt;</code> / <code>&lt;SMeterAnalog&gt;</code>
8861	<code>isAnalog</code>	okomiti razmaci bloka frekvencije
8864	<code>isAnalog</code>	gornja margina bloka finog ugađanja
12288	<code>isV2</code>	mobitel: red finog ugađanja iznad (poredak v2)
12347	<code>isAnalog</code>	mobitel: koji se instrument prikazuje
12433	<code>isV2</code>	mobitel: red finog ugađanja ispod (poredak v1)
13204	<code>isAnalog</code>	veličina teksta gumba noise gate

Sve ostalo u datoteci zajedničko je svim četirima varijantama.

## 5. Ostale postavke koje vrijedi znati

Mjesto	Konstanta	Svrha
App.svelte redak 5034	<code>const visualGain = 1.1;</code>	Kalibracija analognog instrumenta. Svaki 0,10 otprilike je 5 dBm na očitavanju.
SMeterAnalog.svelte redak 55	<code>const SMOOTH_TIME_MS = 16;</code>	Vremenska konstanta zaglađivanja kazaljke, u milisekundama.
SMeterAnalog.svelte redak 64	<code>let smeterTheme = 'dark';</code>	Zadani izgled analognog instrumenta za preglednik bez spremljenog izbora. Postavite ga s <code>./smeter_theme.sh</code> , ne ručno.
SMeterAnalog.svelte redak 69	<code>const SMETER_PREF_VERSION = 2;</code>	Brojač jednokratnog poništavanja spremljenog izgleda. Povećajte ga da postojeće posjetitelje prebacite na zadanu vrijednost; ide samo prema gore.
SMeterDigital.svelte redak 149	<code>const DIGITAL_BAR_TRIM = 0;</code>	Pomiče traku za cijele segmente.
SMeterDigital.svelte redak 31	<code>const numberOfDots = 35;</code>	Broj segmenata trake.

## 6. Nakon uređivanja — ponovna izgradnja

Najjednostavnije, iz korijena repozitorija:

```
./recompile.sh
```

Stavka izbornika **[2]** ponovno gradi frontend i pita koju varijantu treba vidjeti posjetitelj koji dolazi prvi put. Taj se odabir bilježi u `frontend/variant.json` — nijedna izvorna datoteka ne kopira se niti prepisuje. (Prije je iz heredoca ponovno stvarala i `VersionSelector.svelte`; to je uklonjeno kad je birač prestao navigirati, jer bi pri svakom pokretanju poništilo prebacivanje u radu.) Stavka **[1]** je samo backend, **[3]** je oboje.

Ili izravno, iz `frontend/`:

```
./build-all.sh gradi stranicu za računalo I /mobile/ <-- obično koristite ovo
./build-default.sh samo stranicu za računalo
./build-mobile.sh samo /mobile/
```

`build-default.sh` dopušta viteu da isprazni `dist/`, čime nestaje `dist/mobile/` do sljedećeg `build-all.sh` ili `build-mobile.sh`. Zato je `build-all.sh` uobičajen izbor.

`build-all.sh` ima vlastitu kopiju logike izgradnje (`build_version SMETER LAYOUT NAME OUTDIR BASE LOG`) umjesto da poziva `build-default.sh`. Ako mijenjate način na koji se stranica gradi, promijenite to na OBA mjesta ili će se dva puta razići.

Nekoliko pojedinosti koje vrijedi znati:

- Postavlja `PHANTOM_KEEP_OUTDIR=1` kako Vite ne bi praznio izlazni direktorij, nego sam čisti `dist/` — briše sadržaj, ali zadržava `users.json`, koji `spectrumserver` u pogonu prepisuje pri svakom spajanju i odspajanju slušatelja.
- `/mobile` gradi zadnje, kada je izlaz za računalo gotov.
- Njegov mehanizam paralelne izgradnje (`PHANTOM_BUILD_JOBS`, zadano 3) ostatak je iz vremena pet izgradnji za računalo. S jednom, ništa ne čeka u redu.

**Bitovi izvršavanja:** `recompile.sh` pokreće `chmod +x` na skripti prije nego što je izvrši, a `build-all.sh` namjerno poziva `bash build-mobile.sh` umjesto `./build-mobile.sh`, pa nedostajući bit više ne ruši izgradnju. Prijenos skripte kroz GitHubovo web sučelje ipak je vraća na 644, pa ako neku pokrećete izravno:

```
chmod +x frontend/build-*.sh
```

## 7. Kako provjeriti izmjenu prije nego joj povjerujete

Zeleni vite `build` **NIJE** dovoljan. Izgradnja ne može vidjeti identifikator koji je nestao pri premještanju koda između datoteka — to je `ReferenceError` tijekom izvođenja i stranica se jednostavno ne učitava. Dogodilo se dvaput.

## 1) Provjera nedefiniranih identifikatora (hvata upravo taj kvar)

```
cd frontend
npx eslint --no-eslintrc \
 --parser svelte-eslint-parser \
 --rule '{"no-undef":"error"}' \
 --env browser,es2022 \
 src/App.svelte src/lib/SMeter*.svelte
```

Očekuje se jedan poznati, bezopasni pogodak:

```
'dBmCalOffset' is not defined
```

`typeof x === 'number'` nad nedeklariranim imenom nikada ne baca iznimku, pa taj izraz već daje 0. Sve ostalo je pravi problem.

## 2) Izgradite, pa UČITAJTE STRANICU I PROĐITE KROZ SVE ČETIRI VARIJANTE

Ne samo onu koju ste mijenjali — izmjena pod `isAnalog/isV2` može slomiti ostale tri dok vaša radi.

Otvorite `/`, prođite kroz sve četiri stavke u izborniku , pa otvorite `/mobile/`.

Otvorite konzolu preglednika na svakoj. Prazna ili napola iscrtana stranica s `ReferenceError` u konzoli potpis je gornje greške.

## 3) Nikada ne ostavljajte `frontend/dist/` u djelomičnom stanju

Goli `npx vite build` prazni izlazni direktorij i sa sobom odnosi `dist/mobile/` te živi `dist/users.json` poslužitelja — mobilna stranica vraća 404 do sljedeće uspješne izgradnje. Uvijek gradite kroz skripte: `build-all.sh` postavlja `PHANTOM_KEEP_OUTDIR=1` i sam obavlja čišćenje, čuvajući `dist/users.json`. Ako ste u dvojbi, jednostavno ponovno pokrenite `./build-all.sh`.

## POGLAVLJE 12

# Korisnički vodič

USER\_GUIDE.md

## Korisnički vodič za PhantomSDR-Plus

Dobrodošli u PhantomSDR-Plus! Ovaj će vam vodič pomoći da izvučete najviše iz slušanja putem WebSDR-a.

### Sadržaj

1. Uvod
2. Početak rada
3. Pregled sučelja
4. Osnovne radnje
5. Napredne značajke
6. Načini demodulacije
7. Digitalni dekoderi
8. Tipkovni prečaci
9. Oznake
10. Upotreba na mobilnim uređajima
11. Savjeti i dobre prakse
12. Rješavanje problema
13. Često postavljana pitanja

### Uvod

#### Što je PhantomSDR-Plus?

PhantomSDR-Plus je softverski definirani radio (SDR) koji radi u web-pregledniku i omogućuje vam slušanje radijskih signala preko interneta. S vaše strane nisu potrebni ni poseban softver ni hardver — dovoljan je moderan web-preglednik!

#### Što možete slušati?

Ovisno o konfiguraciji WebSDR-a, možete se ugoditi na:

- **Amaterski radio:** radioamatere diljem svijeta
- **Radiodifuzijske postaje:** AM/FM radio, kratkovalne emisije
- **Zrakoplovstvo:** kontrolu zračnog prometa, komunikacije zrakoplova
- **Pomorstvo:** veze brod-obala, pomorsku meteorologiju
- **Meteorološke satelite:** NOAA, METEOR-M
- **Digitalne načine rada:** FT8, RTTY, PSK31 i druge
- **Uslužne postaje:** signale točnog vremena, vojne, državne

## Sistemske zahtjevi

- **Preglednik:** Chrome/Edge (preporučeno), Firefox, Safari
  - **Veza:** širokopojasni internet (preporučuje se 1 Mbit/s ili više)
  - **Zvuk:** ispravni zvučnici ili slušalice
  - **Neobavezno:** miš s kotačićem za lakše ugađanje
- 

## Početak rada

---

### 1. Pristupite WebSDR-u

Otvorite preglednik i idite na adresu WebSDR-a koju vam je dao operater.

Primjer: `http://websdr.example.com:9002`

### 2. Prvo učitavanje stranice

Kada se stranica učitava, vidjet ćete:

- šareni prikaz slapa koji pokazuje radijsku aktivnost
- upravljačku ploču s prikazom frekvencije i gumbima
- S-metar koji pokazuje jakost signala
- brojač korisnika

### 3. Počnite slušati

1. **Kliknite na signal** u prikazu slapa
  2. **Zvuk će se automatski pokrenuti**
  3. **Podesite glasnoću** pomoću regulatora glasnoće preglednika ili klizača na zaslonu
  4. **Fino ugodite frekvenciju** klikom točno na signal
-

## Pregled sučelja

### Glavni dijelovi

WebSDR <callsign>, located in <QTH>      ⌘ [ Analog S-Meter ▼ ] email · Keyboard Shortcuts · Users · Stats · Other servers 1 2 3 Frequency Search: MW SW HamDash PSK Reporter WSPRnet · Callsign Search [ Open Additional Info ]			
Spectrum + Waterfall band-plan strip along the bottom			
Audio & Buffer volume · SQ · zoom AGC Auto Fast Med Slow Compressor Equalizer	Frequency 7,120.00 band · VFO	S-Meter dBm dBµV SNR · NF	Waterfall Controls Min · Max · zoom colour-map strip Zoom In Out Max Min
Filters NR NB NS AN CTCSS Modes selector ... RADEL RADEU USB LSB CW AM QUAM FM Band selector	Fine Tuning (kHz) VFO A · Modes · Bands · IF Filters Wheel Tuning Steps Decoders FT8 FT4 FT2 JS8 CW WSPR FAX SSTV NAVTEX RTTY ...	Waterfall Spectrum Auto Adj. Height [ profile ▼ ] [ Bookmarks ]	

Tri ploče ispod slapa pomiču se kao jedna stranica; u uskom se prozoru slažu jedna ispod druge umjesto jedna uz drugu. **RADEL** i **RADEU** stoje na samom retku naslova *Modes selector*, njemu zdesna.

### 1. Prikaz slapa

Slap je vizualni prikaz radijskih signala:

- **Vodoravna os:** frekvencija
- **Okomita os:** vrijeme (pomiče se prema dolje)
- **Boje:** jakost signala
- **Tamnoplava/crna:** nema signala (razina šuma)
- **Zelena/žuta:** slabi do umjereni signali
- **Narančasta/crvena:** jaki signali
- **Bijela:** vrlo jaki signali

### 2. Prikaz frekvencije

Prikazuje trenutačno ugođenu frekvenciju u raznim formatima:

- **MHz:** 7.100.000 MHz (KV pojasevi)
- **kHz:** 14200 kHz
- **Hz:** 145500000 Hz (VHF/UHF)

### 3. S-metar (analogni ili digitalni)

Prikazuje jakost signala:

- **S1-S9:** standardna ljestvica jakosti signala
- **+dB:** signali iznad S9 (npr. S9+20dB)
- **Analogni ili digitalni:** ovisno o konfiguraciji

## 4. Gumbi načina rada

Odaberite način demodulacije:

- **AM**: amplitudna modulacija
- **FM**: frekvencijska modulacija
- **USB**: gornji bočni pojas
- **LSB**: donji bočni pojas
- **CW**: Morseov kod (neprekinuti val)
- **WBFM**: širokopojasni FM (radiodifuzija)
- **QUAM**: C-QUAM AM stereo — natpis postaje zelen kada se otkrije stereo pilot

Uz naslov **Modes selector** nalaze se gumbi **RADEL** i **RADEU**, koji jednim pritiskom pokreću RADE v1 digitalni glas — pogledajte [Digitalni dekoderi](#).

## 5. Upravljačka ploča

Dodatne kontrole:

- **AGC**: automatska regulacija pojačanja
- **NR**: smanjenje šuma (spektralno)
- **NB**: prigušivač impulsnog šuma
- **NS**: potiskivanje pozadinskog šuma
- **AN**: automatski notch
- **CTCSS**: squelch podtonom (FM). Pritisak otvara pokraj gumba mali prozor s *Any tone* i 49 standardnih podtonova (67,0–254,1 Hz); nakon odabira gumb prikazuje ton (na primjer 88.5 ili ANY), a zvuk prolazi samo dok se taj podton prima — otvaranje traje oko 0,7 s. Ponovni pritisak isključuje CTCSS; Escape ili klik izvan prozora zatvara ga bez promjene. Oznake pamte odabrani ton.
- **SQL**: squelch
- **AI**: smanjenje šuma umjetnom inteligencijom (samo govor)
- **Zoom**: razina uvećanja slapa
- **Wheel Tuning Steps**: korak ugađanja kotačićem miša
- **Decoders**: gumbi koji jednim pritiskom pokreću i zaustavljaju svaki dekodir

## 6. Preklop plana pojaseva

Obojene trake koje prikazuju dodjelu frekvencija:

- različite boje za različite službe
- pomažu prepoznati što je dopušteno na kojoj frekvenciji

## Osnovne radnje

### Ugađanje na frekvenciju

#### Način 1: klik na slap

1. Kliknite izravno na signal u slapu
2. Prijamnik će se ugoditi na tu frekvenciju
3. Zvuk će se početi reproducirati

## Način 2: upis frekvencije

1. Kliknite na prikaz frekvencije
2. Upišite željenu frekvenciju
3. Pritisnite Enter

Primjeri:

- 7100 → 7.100 MHz
- 14200.0 → 14.200 MHz
- 145.500 → 145.500 MHz

## Način 3: kotačić miša

1. Postavite pokazivač iznad prikaza frekvencije
2. Kotačić prema gore povećava frekvenciju
3. Kotačić prema dolje smanjuje frekvenciju

## Način 4: gumbi koraka

1. Koristite gumbе ▲ i ▼ pokraj frekvencije
2. Veličina koraka ovisi o načinu rada:
3. **AM/FM**: koraci od 1 kHz
4. **SSB**: koraci od 100 Hz
5. **CW**: koraci od 10 Hz

## Skener

Skener vodi prijemnik preko raspona kanala i zaustavlja se na prvom na kojem ima signala. Nalazi se u retku **Fine Tuning (kHz)**, skroz desno:

7 152.0 ◀ ■ ▶ ○ 30 dB · 4 ▼

Kontrola	Što radi
◀ ▶	Pretraga dolje / gore. Dok stoji na signalu, strelica nastavlja pretragu.
■	Zaustavljanje.
○	Isključi ovaj kanal iz pretrage — za trajni nosilac ili smetnju.
30 dB · 4 ▼	Prag, a pokraj njega razina koja se upravo mjeri. Otvara postavke.

Tekst lijevo pokazuje što skener radi: **Scanner** u mirovanju, frekvenciju do koje je došao tijekom pretrage, ● 3s dok odbrojava do nastavka, ● 30·22s dok je kanal još zauzet, ● hold kad ostaje na mjestu.

### Postavke

- **Range** — *Scan Band* pretražuje pojas u kojem pretraga počinje; *Scan Visible* pretražuje točno ono što slap prikazuje i prati ga ako zumirate ili ga pomaknete.
- **Scan** — *Every channel* ugađa i sluša svaki kanal redom; *Skip empty* čita spektar i skoči ravno na signale.
- **Stop at** — koliko iznad šuma pojasa kanal mora biti da bi zaustavio pretragu, u dB. Šum se prati neprekidno pa ista postavka vrijedi danju i noću. Prazan kanal ne pokazuje 0 dB — pogledajte na trenutak vrijednost na gumbu i postavite prag iznad nje.
- **Resume after** — koliko dugo kanal mora biti tih prije nego što pretraga krene dalje. Stanke u govoru je ne pokreću ponovno. *Hold* ostaje dok ne pritisnete gumb.

- **Max stay** — kreće dalje nakon ovog vremena i ako je signal još tu, da trajni nosilac ne bi zadržao pretragu zauvijek.

Korak prati način rada — 1 kHz u SSB, 0,1 kHz u CW, 5 kHz u AM, 9 ili 10 kHz na srednjim valovima i 9 kHz na dugim — a zaustavljanja padaju na mrežu kanala. Pretraga ostaje unutar svojeg raspona i unutar onoga što prijemnik može ugoditi; na rubu se vraća na drugi kraj i nastavlja.

Postavke i isključeni kanali pamte se u vašem pregledniku.

## Odabir načina demodulacije

Odaberite način prikladan signalu:

### Za govorne komunikacije:

- **AM**: zrakoplovstvo, AM radiodifuzija, dio amaterskog prometa
- **FM**: VHF/UHF repetitori, FM radiodifuzija
- **USB**: KV amaterski radio (20 m, 17 m, 15 m, 12 m, 10 m)
- **LSB**: KV amaterski radio (160 m, 80 m, 40 m, 30 m)

### Za podatke/digitalne načine:

- **USB**: većina digitalnih načina (FT8, PSK31, RTTY)
- **LSB**: neki digitalni načini na nižim KV pojasevima

### Za Morseov kod:

- **CW**: telegrafski/Morseovi signali

## Podešavanje glasnoće

- **Klizač na zaslonu**: povucite klizač glasnoće
- **Glasnoća preglednika**: koristite medijske kontrole preglednika
- **Glasnoća sustava**: podesite glasnoću računala
- **Tipkovnica**: koristite tipke + i - (ako su podržane)

## Upotreba S-metra

S-metar prikazuje jakost signala:

- **S0-S3**: vrlo slab signal, teško ga je pratiti
- **S4-S6**: slab do osrednji signal
- **S7-S9**: dobar do jak signal
- **S9+**: iznimno jak signal

**Savjet**: za najbolji zvuk ugodite se na signale koji pokazuju S7 ili više.

**Promjena izgleda instrumenta**: na analognom (kazaljčnom) S-metru kliknite na sam instrument — ili ga fokusirajte i pritisnite Enter ili Razmaknicu — da biste prošli kroz tri pozadine: tamni brušeni metal, svijetlu blijedosivu skalu i toplu jantarnu skalu u vintage stilu. Vaš preglednik pamti izbor, pa ostaje i nakon osvježavanja ili ponovnog pokretanja. Vrijedi po pregledniku i po adresi: otvaranje prijamnika preko imena računala i preko IP-a daje dvije odvojene postavke, a privatni prozor uvijek kreće od zadane vrijednosti stranice.

**Iznad 30 MHz** S-metar slijedi VHF standard IARU Regije 1: ondje je S9 na  $-93$  dBm umjesto  $-73$  dBm na kratkom valu, i dalje sa šest dB po S-jedinici. Kazaljka i traka same se prebacuju prema ugođenoj frekvenciji; vrijednosti u dBm i dBμV uvijek pokazuju izmjerenu razinu. Neke stanice svoje VHF/UHF prijemnike podese i tako da instrumenti **miruju na 0 dok je kanal prazan**, kao na VHF/UHF

primopredajniku: iznad 60 MHz kazaljka i traka tada se dižu tek kad signal izađe iz šuma. Vrijednosti dBm, dBμV, SNR i NF uvijek pokazuju stvarne vrijednosti.

## Odabir prijemnika

Neke stanice imaju više od jednog prijemnika — na primjer jedan za kratki val i jedan za opseg od 2 m. U zaglavlju stranice prikazuju redak *Receivers*: s jednim gumbom po prijemniku; onaj na kojem ste prikazan je žuto. Pritiskom na drugi gumb otvara se stranica tog prijemnika, s vlastitim vodopadom, opsezima, chatom i popisom slušatelja. Na stranici /mobile isti gumbi nalaze se u gornjoj traci. Stanica s jednim prijemnikom ne prikazuje takav redak.

## Napredne značajke

### Automatska regulacija pojačanja (AGC)

AGC automatski podešava razine zvuka:

- **Off**: bez automatske regulacije pojačanja
- **Slow**: postupne promjene razine (najbolje za SSB)
- **Medium**: uravnotežen odziv
- **Fast**: brzo podešavanje (najbolje za AM)

**Preporuka:** počnite s „Fast” za AM i „Slow” za SSB.

### Četiri regulatora šuma

NR, NB, NS i AN zasebni su prekidači, svaki protiv druge vrste šuma. Neovisni su — uključivanje jednoga ne uključuje drugi — i mogu se slobodno kombinirati.

Nijedan ne doseže dekodere: FT8, CW, WSPR, SSTV, FAX, NAVTEX, RTTY/PSK31/Olivia i QRSS grabber čitaju zvuk *prije* ovih filtara, pa ih možete namještati isključivo po sluhu bez utjecaja na dekodiranje. Vidjeti [priručnik za dekodere](#).

### Smanjenje šuma (NR)

Spektralno smanjenje šuma. Procjenjuje razinu šuma u svakom dijelu zvučnog spektra i te dijelove stišava, a ono što se izdiže iznad šuma ostavlja na miru.

**Koristite kada:** čujete ravnomjerno siktanje ili bijeli šum iza signala.

Postojani tonovi — CW nota, nosilac — prepoznaju se kao signal i štite, pa NR ne pojede CW signal kako bi to učinio naivan filtar. Tipičan učinak: 10 dB manje šuma između riječi, uz gubitak od nekoliko desetinki dB na samom govoru.

### Prigušivač impulsnog šuma (NB)

Uklanja impulsni šum: klikove, praskove, atmosferske izboje, paljenje motora i smetnje s dalekovoda. Prati ovojnicu zvuka i utišava samo uzorke koji daleko iskaču iznad nje — oko jedne milisekunde po prasku, s mekim ulazom i izlazom kako samo prigušivanje ne bi kliknulo.

**Koristite kada:** čujete klikove ili praskove s dalekovoda, motora, grmljavine ili paljenja vozila.

Notch filtri mrežnog bruma na 50 Hz i 60 Hz slijede ovu tipku.

## Potiskivanje pozadinskog šuma (NS)

Kroz nekoliko sekundi mjeri vlastiti prag šuma pojasa i stalnim iznosom stišava sve što leži na njemu, dublje na visokim nego na niskim zvučnim frekvencijama. Dok NR reagira iz trenutka u trenutak, NS je spora, mirna ruka: spušta šum pojasa ne mijenjajući kako signal zvuči.

**Koristite kada:** pojas je miran ali šuman, a želite spustiti šum bez „podvodnog“ prizvuka agresivnog NR-a.

Djeluje samo u USB, LSB i AM — CW, FM i digitalni načini ostaju netaknuti. Prag mjeri iznova pri svakoj promjeni frekvencije ili načina rada i smiri se unutar nekoliko sekundi.

## Automatski notch (AN)

Sam pronalazi i uklanja postojeće ometajuće tonove — heterodine, nosioce, zvižduke — bez ručnog postavljanja notcha. Neprekidno se prilagođava, pa istodobno uklanja više tonova i drži prigušenim ton koji klizi.

**Koristite kada:** čujete zvižduk ili ton preko željenog signala.

Zaobiđen u CW-u, gdje je željeni signal *upravo* postojan ton.

**Napomena o kašnjenju:** NR i NS dok su uključeni dodaju svaki oko 40 ms kašnjenja zvuka (oba zajedno oko 80 ms). NB i AN ne dodaju ništa. To se tiče samo slušanja, nikada dekodera.

## Automatski squelch (SQL)

Utišava zvuk kada nema signala:

- **Off:** uvijek se čuje (uz šum)
- **Auto:** automatski postavlja prag
- **Manual:** prag se podešava ručno

**Koristite kada:** nadzirete frekvenciju u očekivanju prometa.

## Smanjenje šuma umjetnom inteligencijom (AI)

Tipka **AI**, s klizačem jačine pokraj nje, nalazi se na ploči Audio & Buffer odmah ispod SQ. Na stranici / mobile nalazi se na kartici Audio, ispod Squelch. Uklanja šum pojasa iz govora pomoću RNNoise, male neuronske mreže naučene na govoru.

Sve se odvija u vašem pregledniku: ništa se ne šalje na vanjski poslužitelj, nije potreban račun, a prijamnik nema nikakav dodatni posao.

- **Uključivanje:** kliknite **AI**; tipka postaje plava. Prvi klik preuzima modul (oko 1,3 MB), a tipka pulsira dok se učitava.
- **Jačina:** određuje koliko snažno djeluje dok netko govori — niže vrijednosti zadržavaju više izvornog zvuka. Gornji kraj ljestvice namjerno je blaži od punog rezanja, kako govor ne bi bio oštećen ni na 100 %, što i dalje čisti više nego 50 %. Između riječi sama djeluje jače, pa šum ondje pada još više. Zadana vrijednost je 50 %. Možete je postaviti i dok je AI isključen.
- **Oznaka stanja:** oznaka AI ispod prikaza frekvencije svijetli cijan dok AI radi, a prikazuje se napola prigušena kad je AI uključen, ali trenutni način nije govorni.

**Koristite kad:** slušate SSB ili AM govor na pojasu s puno šuma. Šum između riječi obično padne za oko 10 dB uz zadanih 50 %, a na 100 % više, do oko 20 dB, dok govor zadržava svoju razinu.

Samo govorni načini — USB, LSB, AM i SAM. U CW, FM, digitalnim načinima i C-QUAM zvuk prolazi nepromijenjen, jer mreža CW ton ili glazbu smatra šumom. Kao ni četiri regulatora šuma, nikad ne dolazi do dekodera. Kod slabih stanica (ispod oko +6 dB SNR-a) sama mjeri omjer signala i šuma i jače izglađuje svoje odluke, kako govor ne bi poprimio žuboreći, „vodenasti“ zvuk koji takav filter može dati;

jake stanice obrađuju se točno kao prije. Ako vrlo slab signal i dalje zvuči obrađeno, smanjite klizač. Dodaje oko 50 ms kašnjenja zvuka.

## Zvučni spremnik i kašnjenje

Zvuk stiže u preglednik u malim paketima, kod tipičnog prijamnika po 35 ms, a preglednik zadržava malu zalihi kako paket koji kasni ne bi izazvao prekid. Ta zaliha je **spremnik** (buffer) i čini najveći dio kašnjenja između prijamnika i vašeg zvučnika.

- **Desktop:** u panelu Audio & Buffer tipka s ikonom mreže povećava spremnik za jedan korak po kliku (postaje zelena iznad najnižeg koraka), a klizač pokraj nje izravno bira korak od 1 do 6. Najniži korak odgovara žičanoj vezi; telefoni počinju na koraku 2.
- **/mobile:** tipka **BUF** i klizač **Buffer** u kartici Audio rade isto.

Spremnik se sam prilagođava. Stranica mjeri koliko nepravilno vaša veza isporučuje zvuk i drži samo onoliko zalihe koliko je posljednja minuta pokazala da treba, plus sigurnosnu rezervu odabranog koraka: 20 ms na koraku 1, 50 ms na koraku 2, više na višim koracima. Nakon trenutnog problema s mrežom dodatno kašnjenje ponovno nestaje za minutu-dvije, a mala razlika između takta vaše zvučne kartice i takta prijamnika više ne izaziva prekid svakih nekoliko minuta. Prilagodbe su po nekoliko milisekundi, rade se ondje gdje se zvuk ponavlja i ne čuju se. Dekoderi i vaše snimke nikad nisu pogođeni: dobivaju svaki uzorak.

Ako se zvuk prekida, povećajte spremnik za jedan korak. Na telefonu preko kućnog Wi-Fija kašnjenje od prijamnika do zvučnika obično je oko 100–150 ms; mreža između vas i prijamnika dodaje se na to.

## Chat

Kad je operater uključio chat, on ima vlastiti panel na desktop stranici i karticu **Chat** na /mobile. Upišite svoje ime ili pozivni znak, napišite poruku i pošaljite je.

Da biste odgovorili na određenu poruku, kliknite ➔ **Reply** pokraj njezina vremena. Redak iznad polja za unos pokazuje kome i na što odgovarate; pišite i šaljite kao i obično, ili pritisnite × ili Esc za odustajanje. Odgovor se pojavljuje uvučen ispod poruke na koju odgovara, s kratkim retkom citata. Odgovor na odgovor ide u istu nit. Poslužitelj čuva zadnjih 20 poruka; kad je izvorna poruka već ispala iz te povijesti, odgovor prikazuje samo ime i vrijeme na koje odgovara.

## Funkcija uvećanja

Uvećava prikaz slapa:

- **1x:** uobičajen prikaz (široka pokrivenost)
- **2x:** uvećanje ×2
- **4x:** uvećanje ×4
- **8x:** uvećanje ×8

**Koristite kada:** trebate jasnije vidjeti signale ili se precizno ugoditi.

## Upravljanje primopredajnikom (CAT)

Vaš primopredajnik i stranica prijemnika mogu ostati na istoj frekvenciji: okrenite gumb za ugađanje na uređaju i slap ga slijedi, ili kliknite signal na slapu i uređaj se ugodi na njega. To se odnosi samo na vašu sesiju slušanja; nitko drugi na prijemniku to ne primjećuje.

**S aplikacijom Desktop PhantomSDR+ (4.0 ili novija).** Desktop aplikacija ima izbornik **Rig**. *Rig* → *Rig control...* otvara prozor u kojem birate svoj uređaj i način povezivanja, a sam izbornik uključuje i isključuje sinkronizaciju. Sinkronizira frekvenciju, način rada i širinu filtra, u jednom ili oba smjera, i može

utišati prijemnik dok odašiljete. Osim PhantomSDR-Plusa na isti način upravlja i prijemnicima **KiwiSDR**, **PA3FWM WebSDR** i **UberSDR**. Do uređaja dolazi na jedan od četiri načina:

- **Ugrađeni upravljači** — bez drugog softvera: Icom (CI-V), Yaesu (novi CAT i FT-817/857/897), Kenwood, Elecraft, FlexRadio SmartSDR CAT, QRP Labs i ostali uređaji kompatibilni s Kenwoodom.
- **Hamlib** — svaki uređaj koji Hamlib poznaje, više od 300, odabran s popisa s pretraživanjem. Instalacijski programi za Windows sadrže Hamlib; na Linuxu instalirajte `libhamlib-utils`.
- **rigctld na mreži** — `rigctld` koji je već pokrenut.
- **flrig** — za uređaj koji flrig već dijeli s fldigijem, WSJT-X-om ili dnevnikom veza.

Aplikacija, njezini instalacijski programi i cijeli priručnik nalaze se na [Desktop PhantomSDR+ \(Dropbox\)](#).

**S web preglednikom.** [CATsync Tool for WebSDRs](#) (Windows) povezuje uređaj sa stranicom prijemnika otvorenom u vašem pregledniku. Sinkronizira frekvenciju i način rada.

**Širina filtra i utišavanje pri odašiljanju** rade na prijemnicima KiwiSDR, WebSDR i UberSDR te na PhantomSDR-Plus prijemniku s inačicom 4.1.0 ili novijom. Na starijem PhantomSDR-Plusu frekvencija i način rada i dalje se sinkroniziraju; filter ne.

Cijeli priručnik — svaka postavka, podržani uređaji, kako se sprječava da se strane „svađaju” i rješavanje problema — nalazi se u [Upravljanje primopredajnikom](#).

**Za programere.** Svaka stranica prijemnika nudi ove funkcije na `window`, a njih koriste oba alata:

Funkcija	Što radi
<code>catsync_getFrequency()</code> / <code>catsync_setFrequency(hz)</code>	Ugođena frekvencija, u Hz
<code>catsync_getMode()</code> / <code>catsync_setMode(mode)</code>	USB, LSB, CW, CW-L, AM, FM, WBFM ...
<code>catsync_getBandwidth()</code> / <code>catsync_setBandwidth(hz)</code>	Ukupna širina propusnog pojasa, u Hz. Postavite je nakon načina rada: promjena načina vraća propusni pojas na zadano
<code>catsync_getMute()</code> / <code>catsync_setMute(on)</code>	Utišavanje, preko gumba za utišavanje na stranici
<code>catsync_ready</code>	<code>true</code> čim su funkcije postavljene

Posljednja tri retka nova su u inačici 4.1.0; prije poziva provjerite postoji li funkcija.

## Načini demodulacije

### AM (amplitudna modulacija)

**Koristi se za:**

- zrakoplovne komunikacije
- AM radiodifuziju
- dio amaterskog radija
- pomorske komunikacije

**Značajke:**

- široki pojas (obično 10 kHz)
- osjetljiva na šum
- lako se ugađa (dovoljno je kliknuti na signal)

**Dobre prakse:**

- koristite brzi AGC
- uključite prigušivač impulsnog šuma ako čujete klikove
- ugodite se na vrh signala u slapu

**QUAM (C-QUAM AM stereo)****Koristi se za:**

- Srednjovalne postaje koje emitiraju AM stereo

**Značajke:**

- Širina 10 kHz, dekodira se kao stereo par umjesto mono AM-a
- Natpis na gumbu sam postaje **zelen** kada je prisutan stereo pilot od 25 Hz, pa vidite koje postaje doista emitiraju u stereu prije nego što ga odaberete
- Iz AM-a novi pritisak prebacuje na sinkronu detekciju na nosiocu; natpis postaje **žut** i prikazuje **SAM**. Treći pritisak vraća na obični AM
- C-QUAM zvuk prenosi se Opusom; svi ostali načini koriste FLAC

**Preporuke:**

- Noću tražite zeleni natpis na jakim srednjovalnim signalima
- Ako stereo zvuči nestabilno, **SAM** je mirniji izbor uz slab nosilac

**FM (frekvencijska modulacija)****Koristi se za:**

- VHF/UHF amaterske repetitore
- javne službe (policija, vatrogasci, hitna pomoć)
- komercijalni dvosmjerni radio
- neke satelitske komunikacije

**Značajke:**

- uski pojas (obično 12,5 ili 25 kHz)
- izvrsna otpornost na šum
- „efekt zahvata“ (pobjeđuje najjači signal)

**Dobre prakse:**

- ugodite se precizno na sredinu signala
- koristite squelch za utišavanje u mirovanju
- isključite smanjenje šuma (nije potrebno)

**USB (gornji bočni pojas)****Koristi se za:**

- KV amaterski radio (iznad 10 MHz)
- većinu digitalnih KV načina rada
- pomorske komunikacije (iznad 8 MHz)

**Značajke:**

- uski pojas (obično 2.4 kHz)

- učinkovito iskorištenje spektra
- zahtijeva precizno ugađanje

**Dobre prakse:**

- koristite spori AGC
- ugodite se na donji rub signala u slapu
- po potrebi uključite smanjenje šuma

**LSB (donji bočni pojas)****Koristi se za:**

- KV amaterski radio (ispod 10 MHz)
- neke digitalne KV načine rada
- pomorske komunikacije (ispod 8 MHz)

**Značajke:**

- isto kao USB, ali zrcalno
- konvencija: LSB na nižim KV pojasevima

**Dobre prakse:**

- koristite spori AGC
- ugodite se na gornji rub signala u slapu
- po potrebi uključite smanjenje šuma

**CW (neprekinuti val / Morseov kod)****Koristi se za:**

- amatersku telegrafiju
- navigacijske svjetionike
- postaje signala točnog vremena

**Značajke:**

- vrlo uski pojas (100-500 Hz)
- velika učinkovitost
- za razumijevanje je potrebno poznavanje Morseova koda

**Dobre prakse:**

- koristite uski filter (400-500 Hz)
- ugodite se precizno na sredinu tona
- uključite audiofilter za bolji ton

**CW-L (CW, donji bočni pojas)**

Isti Morseov filter od  $\pm 250$  Hz kao **CW**, ali se ton uzima ispod nosioca umjesto iznad njega. Korisno kada se signal bolje prima s donje strane ili kada smetajući nosilac leži tik iznad željenoga.

**Na stolnoj stranici nema gumba CW-L.** Njezin red načina rada je **USB · LSB · CW · AM · QUAM · FM**. CW-L se nudi na pojednostavljenoj stranici /mobile, a na stolnoj se i dalje može postaviti poveznicom ili knjižnom oznakom koja ga imenuje.

## WBFM (širokopojasni FM)

### Koristi se za:

- FM radiodifuziju (88-108 MHz)
- neke satelitske silazne veze

### Značajke:

- vrlo široki pojas (200 kHz)
- izvrsna kvaliteta zvuka
- visoka vjernost

### Dobre prakse:

- ugodite se točno na središnju frekvenciju
- za radiodifuziju squelch nije potreban
- uživajte u zvuku visoke kvalitete!

## Digitalni dekoderi

PhantomSDR-Plus uključuje ugrađene dekodere za digitalne načine rada. Za potpuni vodič pogledajte [Dekoderi](#).

Najbrži je put red gumba **Decoders** na glavnoj ploči, odmah ispod **Wheel Tuning Steps**: deset gumba — **FT8, FT4, FT2, JS8, CW, WSPR, FAX, SSTV, NAVTX, RTTY** — od kojih svaki jednim pritiskom pokreće svoj dekodera, uključuje glavni prekidač Decoder i dovodi prozor dekodera na zaslon. Gumb ostaje plav dok dekodera radi; pritisnite ga ponovno da zaustavite dekodera i zatvorite njegov prozor. Padajući izbornik **Decoder Options** radi točno kao i prije te ostaje usklađen s gumbima.

**RADEL** i **RADEU** (RADE v1 digitalni glas) imaju vlastite gumbе uz naslov **Modes selector** te u skočnim prozorima **Modes** i **Bands**, i rade na isti način: pritisak za pokretanje, ponovni pritisak za zaustavljanje.

Dekoderi za SSTV, HF FAX, NAVTEX, FSK/RTTY i CW rade svaki u vlastitoj pozadinskoj dretvi, pa uključivanje ili isključivanje dekodera nikada ne prekida zvuk, a slap ostaje gladak tijekom dekodiranja. Dobivaju zvuk uzet prije AGC-a, smanjenja šuma i utišavanja — možete utišati prijamnik, a dekodiranje se nastavlja bez smetnji.

**Dok dekodera radi, drži način rada i propusni pojas.** Uobičajeno način rada slijedi plan opsega — ugodite u odsječak označen kao LSB ili AM i prijamnik se prebaci. Dekodera koji radi nadjačava to i zadržava način rada i uski propusni pojas koji mu treba, čak i ako odete na drugi opseg. Vlastiti način rada opsega vraća se čim isključite dekodera, a gumbi načina rada rade u svakom trenutku ako želite preuzeti upravljanje. CW dekodera je iznimka: dekodira u načinu rada u kojem slušate.

## FT8 i FT4 dekodera

### Što su FT8 i FT4?

- popularni digitalni načini rada u amaterskom radiju
- komunikacija slabim signalima
- emisije od 15 sekundi za FT8 i 7,5 sekundi za FT4

### Kako se koristi:

1. Ugodite se na FT8 ili FT4 frekvencije
2. Odaberite način USB
3. Uključite FT8 dekodera — pritisnite gumb **FT8** ili ga odaberite iz izbornika

#### 4. Promatrajte kako se pojavljuju dekodirane poruke

##### Uobičajene FT8 frekvencije (USB):

- 160 metara: 1.840 MHz
- 80 metara: 3.573 MHz
- 60 metara: 5.357 MHz
- 40 metara: 7.074 MHz
- 30 metara: 10.136 MHz
- 20 metara: 14.074 MHz
- 17 metara: 18.100 MHz
- 15 metara: 21.074 MHz
- 12 metara: 24.915 MHz
- 10 metara: 28.074 MHz
- 6 metara: 50.313 MHz (50.323 MHz za DX)

##### Uobičajene FT4 frekvencije (USB):

- 80 m: 3.575 MHz
- 40 m: 7.0475 MHz
- 30 m: 10.140 MHz
- 20 m: 14.080 MHz
- 17 m: 18.104 MHz
- 15 m: 21.140 MHz
- 12 m: 24.919 MHz
- 10 m: 28.180 MHz
- 6 m: 50.318 MHz

## JS8 dekodер

### Što je JS8?

- FT8-ov mehanizam za slabe signale, korišten za razgovor s tipkovnice na tipkovnicu
- Slobodan tekst umjesto fiksnih poruka: duge poruke stižu kroz nekoliko ciklusa
- Pet brzina; **Normal** (15 s) je pozivna brzina i nosi gotovo sav promet

### Kako se koristi:

1. Ugodite se na JS8 frekvenciju
2. Odaberite način **USB**
3. Uključite JS8 dekodер — tipka **JS8** ili iz izbornika
4. Ostavite **Speed** na *Normal*, a sync offset na **Auto**

### Kako čitati ploču:

- Poruke koje još pristižu prikazuju se na vrhu zeleno, s trepćućim pokazivačem — na Normal poruka može trajati punu minutu
- Dovršene se navode ispod kao Mode | Hz | dB | Message, s **pozivnim znakovima u zelenom**
- Blijed, kurzivni redak istekao je prije posljednjeg okvira: tekst je stvaran, ali može biti skraćen

##### Uobičajene JS8 frekvencije (USB):

- 160m: 1.842 MHz

- 80m: 3.578 MHz
- 40m: 7.078 MHz
- 30m: 10.130 MHz
- 20m: 14.078 MHz
- 17m: 18.104 MHz
- 15m: 21.078 MHz
- 12m: 24.922 MHz
- 10m: 28.078 MHz

JS8 je mnogo tiši od FT8 — jedna emisija svakih nekoliko minuta je normalna, a tišina nije kvar. Potpuni vodič: [Dekoderi](#).

## CW dekode

Samo pritisnite gumb CW i pojavit će se dekodirani tekst. Pritisnite gumb CW još jednom da očistite prozor i ponovno pokrenete dekode.

## WSPR

WSPR (Weak Signal Propagation Reporter, izgovara se „whisper“) način je rada svjetionika s iznimno slabim signalima koji kartira putove KV prostiranja diljem svijeta. Svaka emisija traje otprilike 110 sekundi i stane u kanal širine 200 Hz. Dekoder čeka potpuni dvominutni interval usklađen s UTC-om prije dekodiranja. Uključite dekode i s padajućeg popisa odaberite **WSPR**. Ili jednostavno pritisnite gumb **WSPR**.

## HF FAX / WEFAX

HF radiofaks (poznat i kao WEFAX) koriste obalne straže i meteorološke službe diljem svijeta za emitiranje vremenskih karata, karata stanja mora i prizemnih analiza na kratkim valovima. Dekoder rekonstruira sliku redak po redak kako je prima.

## NAVTEX

NAVTEX je međunarodni pomorski sustav emitiranja obavijesti o sigurnosti u priobalju — navigacijskih upozorenja, vremenskih prognoza te obavijesti o traganju i spašavanju.

## FSK / RTTY, PSK31 i Olivia

Univerzalni dekode za uskopojasne tekstualne načine rada, s pet inačica u jednom prozoru: pomorski FSK (SITOR), meteorološki RTTY, amaterski RTTY, **PSK31** (fazno ključanje, 31,25 bauda) i **Olivia** (viševalni FSK s korekcijom pogrešaka). Ploča se prilagođava inačici — kontrole za shift, baud i okvir nestaju za PSK31 i Oliviju, a Olivia dodaje birač Mode i klizač squelcha.

Amaterski i meteorološki RTTY imaju klizač **Squelch (SNR)** koji sprječava da se šum ispisuje kao nasumična slova, a sve varijante osim Olivije prikazuju pravi SNR na istoj ljestvici od 3 kHz. Dvije stvari treba znati: PSK31 sam ispravlja pogrešku ugađanja u rasponu od otprilike  $\pm 25$  Hz, pa se dovoljno samo približiti; Olivia traži da **Mode** (tonovi / širina pojasa) točno odgovara emisiji, ne šalje preambulu i stoga joj treba nekoliko sekundi za sinkronizaciju prije nego što se pojavi tekst. Ploča se otvara na Olivia **8 / 250**.

## QRSS grabber

QRSS je CW poslan tako sporo da jedna točka traje sekundama, i gleda se umjesto da se sluša — grabber crta trag na vlastitom prikazu. Nije u izborniku dekodera; ima vlastiti odjeljak **QRSS** i može raditi istodobno s dekodером.

Pritisnite **Show**, zatim odaberite prozor s popisa **Band** i pritisnite **Tune**: prijamnik odlazi na tu frekvenciju u **CW**-u s propusnim pojasom skrojenim za taj način rada, a trag pada na središnju crtu prikaza. Najprometniji je prozor na 30 m (10.140,00 kHz). Postavite **Speed** prema svjetioniku — **QRSS 10** ako ne znate — i budite strpljivi: pozivnom znaku može trebati deset minuta da prijeđe zaslon. Sve pojedinosti u [Dekoderi](#).

## SSTV

Televizija sporog raščlanjivanja šalje nepomične slike običnim SSB kanalom, redak po redak. Uključite dekodер, odaberite **SSTV** i ugodite se na SSTV frekvenciju — 14.230 MHz glavna je međunarodna pozivna frekvencija. Ostavite **Mode** na **Auto**: dekodер čita VIS zaglavlje emisije, a ako ste se ugodili nakon zaglavlja, način rada prepoznaje prema vremenskim odnosima sinkronizacije. Podržani su načini Martin, Scottie i Robot, a slika se gradi redak po redak kako pristiže. Ili jednostavno pritisnite gumb **SSTV**.

## Automatska prijava spotova i grafikoni poslužitelja

Dekodiranja FT8, FT4 i WSPR može slati i sam poslužitelj — FT8/FT4 na PSK Reporter, WSPR na WSPRnet — pomoću autorun daemona koji sysop pokreće s administratorske ploče. Neovisan je o dekodерima u vašem pregledniku: radi bez obzira sluša li tko ili ne, a ništa što dekodirate u pregledniku ne prijavljuje se.

Sysop ga prati preko dva brojača koja je lako zamijeniti: pločice po dekodерu broje spotove poslane od zadnjeg pokretanja daemona, dok je broj uz svaki potvrdni okvir pojasa/načina rada ukupan zbroj tog mjesta od početka i preživljava ponovna pokretanja. Ista ploča ima stranicu **Grafikoni** koja crta frekvenciju procesora, opterećenje, temperaturu i broj korisnika kroz zadnjih 15 minuta do 24 sata. Oboje je opisano u [vodiču za administratorsku ploču](#).

## Tipkovni prečaci

Tipkovni prečaci za brži rad.

## Upravljanje frekvencijom

Prikaz frekvencije je niz znamenki kojima upravljate izravno. **Najprije kliknite znamenku** — time je odabirete, a sve niže djeluje na odabir.

- **Strelica lijevo / desno**: pomiče odabir na susjednu znamenku (osam znamenki, od 100 MHz do 10 Hz)
- **Strelica gore / dolje**: povećava ili smanjuje *odabranu* znamenku za njezinu mjesnu vrijednost — 1 MHz na znamenki MHz, 10 Hz na zadnjoj
- **0 - 9**: upisuje tu znamenku izravno na odabrano mjesto

**Kotačić miša iznad prikaza frekvencije**: pomiče se korakom ugađanja pojasa (1 kHz osim ako plan pojasa kaže drukčije). Držite **Shift** za 1 kHz, **Alt** za 10 kHz.

**Kotačić miša iznad slapa**: zumira. Držite **Ctrl** (ili **Cmd**) odnosno **Shift** za ugađanje umjesto zumiranja, a **Shift + Ctrl** zajedno za poravnanje na najbliži cijeli kHz.

Prečaca Page Up / Page Down nema.

## Oznake

Spremite omiljene frekvencije za brzi pristup. Popis oznaka možete i izvesti te spremiti lokalno, a zatim ga uvesti u bilo koji drugi PhantomSDR.

Kada se oznake i markeri preklapaju:

Plave se oznake prikazuju iznad

Žuti se markeri prikazuju ispod

Klikovi na oznake imaju prednost

### Dodavanje oznake

1. Ugodite se na željenu frekvenciju
2. Kliknite gumb „Bookmarks“
3. Kliknite „Add Bookmark“
4. Unesite opis
5. Kliknite „Save“

			Bookmark name   Label (optional)   [Add]
--	--	--	------------------------------------------

### Kako se koristi:

#### Dodavanje oznake:

- Naziv: „Local News Station“
- Oznaka: „NEWS“
- Kliknite Add

#### Prikaz u slapu:

- uvećavajte dok se markeri ne pojave
- vidjet ćete tamnoplavi okvir s natpisom „NEWS“ podebljano žutim

#### Klik na oznaku:

- ugađa na frekvenciju
- postavlja način demodulacije
- radi jednako kao klik na marker

### Upravljanje oznakama

- **Uređivanje:** kliknite ikonu olovke pokraj oznake
- **Brisanje:** kliknite ikonu koša pokraj oznake
- **Izvoz:** preuzmite oznake kao JSON datoteku
- **Uvoz:** učitajte oznake iz JSON datoteke

### Dijeljenje oznaka

1. Kliknite „Export Bookmarks“
2. Podijelite JSON datoteku s drugima

3. Primatelji kliknu „Import Bookmarks“
4. Odaberu vašu datoteku

## Upotreba na mobilnim uređajima

PhantomSDR-Plus izvrsno radi na mobilnim uređajima!

### Dva mobilna prikaza

Prijamnik se na telefonu može koristiti na dva načina:

- **`http://vaš_poslužitelj:PORT/mobile`** — pojednostavljena stranica. Bez slapa, pa troši otprilike upola manje podataka. Unos frekvencije, koraci ugađanja, načini rada, S-metar, pojasevi, oznake, korisnici i čavrljanje.
- **Prošireni prikaz** — telefonski raspored glavnog sučelja, sa slapom i svim kontrolama.

Prebacuje se gumbima na dnu `/mobile` (**Mobile extended view, Full desktop view**) i gumbom **Simplified mobile** u proširenom prikazu.

**Vaša frekvencija ide s vama.** Promjena prikaza ostavlja vas na istom signalu — frekvencija i način rada putuju u poveznici, pa više ne završavate na zadanoj frekvenciji prijamnika. I adresna traka prati vaše ugađanje, pa ponovno učitavanje stranice, oznaka ili poveznica poslana nekome vraćaju točno na tu frekvenciju.

**Način rada slijedi plan pojaseva na obje stranice.** Ugodite se na segment označen kao AM, LSB, USB ili CW — upisivanjem frekvencije, koracima ili gumbom pojasa — i prijamnik prelazi na taj način rada, na pojednostavljenoj stranici jednako kao i u punom sučelju. Način koji odaberete ručno ostaje dok se krećete unutar istog segmenta, a izvan definiranih pojaseva vaš se način ne dira. Dok **RADE** (RADEL/RADEU) radi, on zadržava prijamnik, pa ga ugađanje ne prekida.

Pri prebacivanju između dva prikaza trenutačni način rada ide s vama; ako prikaz u koji prelazite nema odgovarajući, o njemu odlučuje plan pojaseva za tu frekvenciju — radiodifuzijska frekvencija stiže u AM, 40 m u LSB, CW segment u CW. SAM s pojednostavljene stranice postaje AM sa sinkronim detektorom u proširenom prikazu, i obratno. RADEL i RADEU postoje samo na pojednostavljenoj stranici: napuštanjem jednog od njih frekvencija se čuva, a način rada bira plan pojaseva.

Frekvencija izvan pokrivenosti prijamnika povlači se na najbliži rub, pa vas stara poveznica nikada ne može ostaviti izvan pojasa.

### Značajke posebno za mobilne uređaje

- **Kontrole prilagođene dodiru:** veliki gumbi i klizači
- **Povlačenje za ugađanje:** povucite lijevo/desno po slapu
- **Stiskanje za uvećanje:** stisnite slap za približavanje/udaljšavanje
- **Vodoravni način:** okrenite uređaj za bolji pregled

### Savjeti za mobilne uređaje

1. **Koristite Wi-Fi:** prijenos zvuka troši podatkovni promet
2. **Vodoravna orijentacija:** bolji pogled na slap
3. **Slušalice:** bolja kvaliteta zvuka
4. **Spremite omiljene postaje kao oznake:** lakše im se vraćate
5. **Zatvorite druge aplikacije:** osigurava gladak rad

## Preporučeni mobilni preglednici

- **Android:** Chrome ili Samsung Internet
  - **iOS:** Mozilla
  - **Oba:** pobrinite se da je preglednik ažuriran
- 

## Savjeti i dobre prakse

---

### Za najbolji prijam

1. **Birajte jake signale:** tražite narančastu/crvenu boju u slapu
2. **Ugodite se precizno:** kliknite točno na sredinu signala
3. **Odaberite ispravan način rada:** uskladite ga s vrstom signala
4. **Podesite AGC:** brzi za AM, spori za SSB
5. **Koristite NR/NB:** pomažu u uvjetima s puno šuma

### Pronalaženje aktivnosti

1. **Promatrajte slap:** boje pokazuju jakost signala
2. **Slušajte popularne frekvencije:**
3. 40 m: 7.100-7.300 MHz (LSB)
4. 20 m: 14.200-14.350 MHz (USB)
5. 2 m: 145.200-145.600 MHz (FM)
6. **Provjerite preklap plana pojaseva:** prikazuje dodjelu frekvencija
7. **Koristite oznake:** brz pristup aktivnim frekvencijama

### Razumijevanje uvjeta prostiranja

#### KV danju:

- viši pojasevi rade bolje (20 m, 15 m, 10 m)
- moguća je veza na velike udaljenosti (DX)
- čuju se radiodifuzijske postaje

#### KV noću:

- niži pojasevi rade bolje (80 m, 40 m)
- drukčiji obrasci prostiranja
- čuju se druge postaje

#### VHF/UHF:

- uglavnom optička vidljivost
- lokalne komunikacije
- ujednačeniji uvjeti

### Bonton

1. **Nemojte dugo zauzimati prijamnik:** i drugi žele slušati
2. **Koristite razgovor s poštovanjem:** budite pristojni prema drugim korisnicima
3. **Prijavite probleme:** pomozite operateru u održavanju postaje

4. **Nemojte tražiti tehničku podršku:** ovo je platforma za slušanje. Za pomoć pošaljite poruku SysOpu.

## Rješavanje problema

### Nema zvuka

#### Mogući uzroci:

1. Preglednik je utišan → provjerite regulaciju glasnoće u pregledniku
2. Sustav je utišan → provjerite glasnoću računala
3. Slab signal → ugodite se na jači signal (S7+)
4. Pogrešan način rada → isprobajte druge načine demodulacije

#### Rješenja:

1. Kliknite na jak signal (narančast/crven u slapu)
2. Provjerite da preglednik nije utišan (ikona utišavanja na kartici)
3. Isprobajte drugu frekvenciju
4. Ponovno učitajte stranicu (F5)

### Izobličen zvuk

#### Mogući uzroci:

1. Preupravljan signal → signal je prejak
2. Pogrešan način rada → AM signal slušan u SSB načinu itd.
3. Smetnje → prelijevanje susjednih signala

#### Rješenja:

1. Smanjite glasnoću
2. Isprobajte drugi način demodulacije
3. Upotrijebite užu pojas filtra
4. Odmaknite se od ometajućih signala

### Zvuk se prekida ili kasni

#### Mogući uzroci:

1. Nepravilna veza → paketi stižu kasno i spremnik se isprazni
2. Zauzet telefon ili računalo → preglednik ne stiže

#### Rješenja:

1. Povećajte spremnik za jedan korak (vidi *Zvučni spremnik i kašnjenje*)
2. Radije koristite Wi-Fi ili kabel nego mobilne podatke
3. Zatvorite ostale kartice i aplikacije

Za detaljniji uvid otvorite stranicu s dodanim `?diag=1` na adresi (na primjer `http://your_server:PORT/mobile?diag=1`). Stranica `/mobile` tada ispod svojih kontrola prikazuje kašnjenje, koliko je veza nepravilna i kako stoji spremnik; desktop stranica isti redak upisuje u konzolu preglednika (F12) svakih deset sekundi. Korisno kad operateru stanice prijavljujete problem.

## Slap se ne osvježava

### Mogući uzroci:

1. Problem s mrežom → spora ili prekinuta veza
2. Performanse preglednika → previše otvorenih kartica
3. Preopterećenje poslužitelja → previše korisnika

### Rješenja:

1. Provjerite internetsku vezu
2. Zatvorite nepotrebne kartice preglednika
3. Ponovno učitajte stranicu (F5)
4. Pokušajte kasnije, kada je manje korisnika na vezi

## Nije moguće ugoditi se na frekvenciju

### Mogući uzroci:

1. Frekvencija izvan raspona → SDR ne pokriva tu frekvenciju
2. Pogrešan format upisa → koristite ispravan format (npr. „14200“, a ne „14.200.000“)

### Rješenja:

1. Provjerite frekvencijsku pokrivenost SDR-a (prikazana na stranici)
2. Koristite navedene primjere formata frekvencije
3. Umjesto toga kliknite na slap

## Isprekidan zvuk

### Mogući uzroci:

1. spora internetska veza
2. veliko opterećenje poslužitelja
3. problemi s performansama preglednika

### Rješenja:

1. Zatvorite druge aplikacije koje troše propusnost
2. Pokušajte izvan vršnih sati
3. Zatvorite nepotrebne kartice preglednika
4. Koristite žičanu vezu umjesto Wi-Fija

---

## Često postavljana pitanja

### Opća pitanja

**P: Trebam li posebnu opremu za korištenje WebSDR-a?** O: Ne! Samo računalo ili mobilni uređaj s pristupom internetu.

**P: Je li korištenje WebSDR-a besplatno?** O: Da, većina WebSDR-ova je besplatna. Vode ih volonteri.

**P: Mogu li odašiljati putem WebSDR-a?** O: Ne, WebSDR služi samo za prijam. Odašiljanje nije moguće.

**P: Koje frekvencije mogu slušati?** O: Ovisi o konfiguraciji WebSDR-a. Pogledajte podatke o postaji.

**P: Mogu li snimati zvuk?** O: Neki preglednici omogućuju snimanje. Provjerite mogućnosti svojeg preglednika.

## Tehnička pitanja

**P: Koju frekvenciju uzorkovanja koristi SDR?** O: Razlikuje se od postaje do postaje. Pogledajte stranicu s podacima o postaji.

**P: Kolika je latencija?** O: Obično nekoliko desetinki sekunde između radijskog signala i vaših zvučnika: obrada u prijamniku, mreža i spremnik reprodukcije preglednika, koji se prilagođava vašoj vezi (oko 100-150 ms na telefonu preko kućnog Wi-Fija). Vidi *Zvučni spremnik i kašnjenje*.

**P: Mogu li koristiti više instanci?** O: Obično da, ali to može opteretiti poslužitelj. Budite obzirni.

**P: Radi li izvan mreže?** O: Ne, WebSDR zahtijeva internetsku vezu.

**P: Koji su preglednici podržani?** O: Chrome, Firefox, Edge, Safari (sve novije verzije)

## Pitanja o korištenju

**P: Koliko ljudi može slušati istodobno?** O: Ovisi o kapacitetu poslužitelja. Često 50-200+ korisnika.

**P: Mogu li vidjeti što drugi slušaju?** O: Ako je omogućeno, da. Potražite oznake „drugi korisnici“.

**P: Mogu li razgovarati s drugim slušateljima?** O: Ako ga je operater uključio. Potražite okvir za chat; ↪ **Reply** odgovara na određenu poruku (vidi *Chat* gore).

**P: Zašto se na nekim frekvencijama ništa ne vidi?** O: Na toj frekvenciji trenutno nema signala. Isprobajte druge!

**P: Što su obojene trake u slapu?** O: Preklop plana pojaseva koji prikazuje dodjelu frekvencija.

## Izvori

### Saznajte više o radiju

- **Planovi pojaseva:** potražite „amateur radio band plan“ + svoju regiju
- **Prostiranje:** naučite o prostiranju KV radiovalova
- **Digitalni načini rada:** istražite FT8, PSK31, RTTY
- **Amaterski radio:** razmislite o polaganju za amatersku dozvolu!

### Pronalaženje drugih WebSDR-ova

- **WebSDR imenik:** <http://sdr-list.xyz>
- **WebSDR.org:** <http://websdr.org>
- **KiwiSDR:** <http://kiwisdr.com/public/>

### Kako doći do pomoći

1. **Operater postaje:** pogledajte kontaktne podatke na stranici
2. **Razgovor korisnika:** pitajte druge slušatelje (ako je dostupan)
3. **Mrežni forumi:** potražite zajednice WebSDR-a
4. **Dokumentacija:** pogledajte ovaj vodič!

## Dodatak: uobičajene frekvencije

### KV amaterski pojasevi

Pojas	Frekvencijski raspon	Način rada	Aktivnost
160 m	1.800-2.000 MHz	LSB	Noć/lokalno
80 m	3.500-4.000 MHz	LSB	Noć/regionalno
40 m	7.000-7.300 MHz	LSB	Dan/noć/DX
30 m	10.100-10.150 MHz	USB	Samo podaci/CW
20 m	14.000-14.350 MHz	USB	Danju/DX
17 m	18.068-18.168 MHz	USB	Danju/DX
15 m	21.000-21.450 MHz	USB	Danju/DX
12 m	24.890-24.990 MHz	USB	Danju/DX
10 m	28.000-29.700 MHz	USB	Sporadično/DX

### VHF/UHF amaterski pojasevi

Pojas	Frekvencijski raspon	Način rada	Aktivnost
6 m	50.000-54.000 MHz	USB/FM	Sporadično
2 m	144.000-148.000 MHz	FM	Vrlo aktivno
70 cm	420.000-450.000 MHz	FM	Aktivno

### Radiodifuzijski pojasevi

Služba	Frekvencijski raspon	Način rada
AM radio	530-1710 kHz	AM
Kratki valovi	2.3-26.1 MHz	AM
FM radio	88-108 MHz	WBFM

### Zrakoplovstvo

Služba	Frekvencijski raspon	Način rada
Kontrola zračnog prometa	118-137 MHz	AM
ACARS (podaci)	130-136 MHz	Podaci

### Pomorstvo

Služba	Frekvencijski raspon	Način rada
Pomorski VHF	156-162 MHz	FM
Pomorski KV	2-22 MHz	USB

## Pojmovnik

---

**AGC:** automatska regulacija pojačanja – automatski podešava razine zvuka

**AM:** amplitudna modulacija – govorni način rada u zrakoplovstvu i radiodifuziji

**Širina pojasa:** frekvencijski raspon signala

**CW:** neprekinuti val – signali Morseova koda

**DX:** veza na veliku udaljenost

**FFT:** brza Fourierova transformacija – pretvara vremensku domenu u frekvencijsku

**FM:** frekvencijska modulacija – govorni način rada za VHF/UHF

**HF:** kratki valovi (3-30 MHz) – pojasevi za velike udaljenosti

**kHz:** kilohertz (1.000 Hz)

**LSB:** donji bočni pojas – govorni način rada za niže KV pojaseve

**MHz:** megahertz (1.000.000 Hz)

**NB:** prigušivač impulsnog šuma – uklanja impulsni šum

**NR:** smanjenje šuma – smanjuje pozadinski šum

**PSK:** fazno pomično ključanje – digitalni način rada

**RTTY:** radioteleprinter – digitalni tekstualni način rada

**S-metar:** mjerač jakosti signala

**SDR:** softverski definirani radio

**SQL:** squelch – utišava zvuk kada nema signala

**SSB:** jednobočni pojas (USB ili LSB)

**USB:** gornji bočni pojas – govorni način rada za više KV pojaseve

**VHF:** vrlo visoke frekvencije (30-300 MHz) – optička vidljivost

**UHF:** ultravisoke frekvencije (300-3000 MHz) – optička vidljivost

**Slap (waterfall):** vizualni prikaz radijskog spektra kroz vrijeme

---

**Uživajte u istraživanju radijskog spektra uz PhantomSDR-Plus!**

**73 (srdačni pozdravi) de SV1BTL & SV2AMK**

Za upute o instalaciji pogledajte [INSTALLATION.md](#). Za tehničke pojedinosti pogledajte [PROJECT\\_STRUCTURE.md](#).

## POGLAVLJE 13

# Emulacija KiwiSDR klijenta

Aether\_config.md

## Emulacija KiwiSDR klijenata

### Povezivanje AetherSDR-a, kiwiclienta i drugog KiwiSDR softvera na PhantomSDR-Plus

Od v4.1.0 PhantomSDR-Plus može odgovarati i na **KiwiSDR protokol**, pa se softver pisan za KiwiSDR — **AetherSDR**, **kiwiclient** i ostali — spaja izravno na vaš prijamnik. Riječ je o mostu unutar istog poslužiteljskog procesa, na istom računalu i istom portu koji već objavljujete: bez drugog servisa, bez drugog porta, bez posrednika.

**Isključen je dok ga ne uključite.** Ništa od mosta ne radi — ni utičnica, ni provjera — dok `[kiwi_emulation]` nedostaje ili je `false`, a to je stanje svježe instaliranog stabla.

## Sadržaj

1. Što dobiva Kiwi klijent
2. Instalacija mosta
3. Uključivanje
4. Pregled postavki
5. Spajanje klijenta
6. Razina zvuka
7. S-metar
8. Slap i spektar
9. Ako nešto ne izgleda dobro
10. Dokle je provjereno

## 1. Što dobiva Kiwi klijent

<b>Zvuk</b>	<code>ws://&lt;host&gt;:&lt;port&gt;/kiwi/&lt;id&gt;/SND</code> — demodulirani zvuk u Kiwijevoj SND okviru
<b>Slap</b>	<code>ws://&lt;host&gt;:&lt;port&gt;/kiwi/&lt;id&gt;/W/F</code> — spektar, u Kiwijevoj W/F okvirima
<b>Ugađanje</b>	stvarno, ne kozmetičko: <code>SET mod=...</code> s klijenta mijenja frekvenciju, bočni pojas i propusni pojas na PhantomSDR strani, jednako kao što bi to učinio slušatelj u pregledniku
<b>S-metar</b>	pogonjen snagom po bloku samog demodulatora, na istoj skali kao i web S-metar

Ugađanje je podržano za **realni** ulaz (`signal = "real"` u vašem `.toml`). Za IQ ulaz odnos između binova i frekvencije je drukčiji i nije implementiran, pa će IQ prijamnik davati zvuk i slap, ali neće pratiti ugađanje Kiwi klijenta.

## 2. Instalacija mosta

Most je skup zakrpa na izvorni kod pozadinskog dijela plus jedno novo zaglavlje, `src/kiwi_bridge.h`. Postoje dva načina da ih ugradite.

**Instalacijskim programom.** `install.sh` (te `install_arch.sh`, `install_fedora.sh`, `install_opensuse.sh`) imaju vlastiti korak za to — korak 17, *Kiwi client emulation*. Nudi se, ne nameće: odgovorite `n` i ništa se ne zakrpa. Za nenadzirano izvođenje odlučuje `PHANTOM_KIWI=y|n`, a zadano je da.

**Na stablu koje je već instalirano** pokrenite skriptu koja dolazi u korijenu repozitorija:

```
cd ~/PhantomSDR-Plus
./kiwi_install.sh
./recompile.sh # odaberite [1] Samo pozadinski dio
```

`kiwi_install.sh` najprije spremi svaku datoteku koje se dotakne u `backup_kiwi_bridge_<vremenska-oznaka>/` i primjenjuje svaku zakrpu točnim podudaranjem teksta — ako sidro nije ondje gdje ga očekuje (lokalno izmijenjeno stablo, druga inačica), tu se zaustavi, imenuje datoteku i ostavi je netaknutom. Idempotentan je: na stablu koje već ima most prijavljuje svaku zakrpu kao već primijenjenu i ništa ne mijenja.

**Ako ste upravo primijenili arhivu ažuriranja koja već sadrži most, ne morate ga pokretati.** Zakrpani izvorni kod je u paketu; `kiwi_install.sh` bi vam samo rekao da je svaka zakrpa već na mjestu.

Datoteke koje zakrpava su `src/client.h`, `src/signal.cpp`, `src/waterfall.cpp`, `src/spectrumserver.h`, `src/spectrumserver.cpp`, `src/websocket.cpp` i `src/http.cpp`, a `kiwi_bridge.h` kopira u `src/`.

**Instalira most, ne nadograđuje ga.** Ako zakrpa ikada *padne* na stablu na kojem most već radi, izvorni kod je otišao ispred skripte — i skriptu treba osvježiti, ne vaše stablo.

## 3. Uključivanje

Dodajte ovo u `.toml` s kojim vaš prijamnik zaista radi — istu datoteku koju predajete `spectrumserveru`, onu koju imenuje vaš `start-<radio>.sh`. Ne `config.example.*`:

```
[kiwi_emulation]
enabled = true
```

Zatim ponovno pokrenite prijamnik:

```
./stop-websdr.sh
./start-<vaš-radio>.sh
```

`kiwi_install.sh` dodaje blok umjesto vas, s `enabled = true`, u one od `config.toml`, `config-rx888mk2.toml`, `config-airspyhf.toml`, `config-rtl.toml`, `config-rspla.toml`, `config-fobos.toml`, `config-fobos-hf.toml` i `config-hackrf.toml` koje postoje u korijenu repozitorija. Ne može znati za konfiguraciju koju držite drugdje ili pod drugim imenom, a **archive ažuriranja nikada ne isporučuju** `.toml`, pa je nakon ažuriranja blok na vama da ga dodate ručno. Isporučene `config.example.*.toml` nose ga dokumentiranog i postavljenog na `false`.

## 4. Pregled postavki

Sve što most čita nalazi se u jednom odjeljku `[kiwi_emulation]`. Obavezan je samo `enabled`; sve četiri fine postavke imaju kalibriranu vrijednost kao zadanu, pa je blok koji sadrži samo `enabled = true` već ispravan.

Ključ	Zadano	Što radi
<code>enabled</code>	<code>false</code>	Uopće odgovarati na KiwiSDR protokol. Isključeno = potpuno neaktivno.
<code>audio_gain</code>	<code>0</code>	dB. Izlazno pojačanje, samo za Kiwi klijente. Vidi <a href="#">Razina zvuka</a> .
<code>smeter_offset</code>	<code>input.analog_smeter_offset</code>	dB. Nadjačava pomak S-metra, samo za Kiwi klijente. Vidi <a href="#">S-metar</a> .
<code>wf_cal</code>	<code>0</code>	dB. Prikazna korekcija za slap i spektar. Vidi <a href="#">Slap i spektar</a> .
<code>wf_fps_max</code>	<code>23</code>	Sličica slapa u sekundi. Vidi <a href="#">Slap i spektar</a> .

Potpuno ispunjen blok, s vrijednostima s kojima radi prijamnik ovog projekta:

```
[kiwi_emulation]
enabled = true
audio_gain = 60.0 # dB, samo Kiwi klijenti. 0 = netaknuto.
wf_cal = -10.0 # dB, stvar ukusa. 0 = slaže se sa S-metrom.
wf_fps_max = 28 # sličica/s. 23 je maksimum protokola.
smeter_offset = 5.0 # dB. Zadano je input.analog_smeter_offset.
```

**Svaki se ovdje navedeni ključ čita pri pokretanju.** Promjena bilo kojeg od njih je **ponovno pokretanje**, a ne ponovno prevođenje — `./stop-websdr.sh`, pa vaša skripta za pokretanje. Za njihovo podešavanje nikada ne morate ponovno prevoditi.

Prihvaćaju se i cjelobrojni i decimalni oblik: `28` i `28.0` čitaju se jednako. Samo ključ koji zaista nedostaje pada na svoju zadanu vrijednost.

## 5. Spajanje klijenta

Usmjerite klijent na istu adresu i port koje koriste vaši slušatelji — bez staze, bez prefiksa. AetherSDR traži računalo i port; `kiwiclient` uzima `-s` i `-p`:

```
kiwiclient, snimanje 30 sekundi na 7100 kHz LSB
python3 kiwirecorder.py -s vas.prijamnik.example -p 8073 -f 7100 -m lsb --tlimit=30
```

Klijent se ugađa, slap se puni, S-metar pokazuje. Ako se baš ništa ne događa, most je gotovo sigurno još uvijek isključen — vidi [odjeljak 3](#).

## 6. Razina zvuka

Prvo što većina primijeti jest da Kiwi klijent zvuči tanje od PhantomSDR web stranice. Nije da most nešto gubi: jedan međuspremnik hrani svaki koder, a ista frekvencija s istim propusnim pojasom, izmjerena kroz preglednikov vlastiti audio put i kroz `/SND`, daje istovjetne uzorke. Web stranica je glasna jer `audio.js` iznova gradi zvuk u pregledniku — naglašavanje basova, pojasni filter, podizanje prisutnosti,

kompresor s nadoknadnim pojačanjem i klizač glasnoće. Kiwi klijent nema ništa od toga i ne može to dobiti.

`audio_gain` zatvara tu razliku, samo za Kiwi klijente:

```
[kiwi_emulation]
enabled = true
audio_gain = 55.0 # dB, samo Kiwi klijenti. 0 = netaknuto.
```

**Počnite oko 55-60 dB.** 55 dB nije račun: to je vrijednost pri kojoj su Kiwi klijent i PhantomSDR web stranica izmjereni na istoj razini, na jakom lokalnom signalu na 729 kHz. U praksi se često voli malo više — ovaj se prijamnik zaustavio na 60 — pa birajte po sluhu.

Time ne možete pokvariti zvuk. **Graničnik vrhova s uvidom unaprijed** stoji između pojačanja i 16-bitnog odsijecanja: svaki se uzorak kasni 4 ms dok se pojačanje računa iz zvuka koji još nije poslan, pa glasan odsječak dočeka pojačanje koje je već sišlo zbog njega, a vrhovi se preklapaju umjesto da budu spljošteni. Potjeran na 60 pa i 70 dB — daleko iznad rezerve koju samo pojačanje ima — nije odsječen ni jedan uzorak od četvrt milijuna. Ispod praga graničnika pojačanje je točno 1,0, pa prijamnik koji ne tjera zvuk dobiva uzorke netaknute.

Previsoka vrijednost stoji glasnoće, ne kvalitete. Iznad otprilike 60 dB dodatno se pojačanje jednostavno ograniči: izmjereno ovdje, razina je porasla oko 1 dB od 55 do 60 i jedva se pomaknula od 60 do 70. Nakon toga kupujete kompresiju, a ne glasnoću.

Ništa od toga ne dira web stranicu i ništa ne pomiče S-metar — on dolazi iz snage demodulatora, a ne iz audio uzoraka, pa pojačavanjem glasnoće ne možete natjerati instrument da laže.

## 7. S-metar

Kiwijev S-metar nije zasebno kalibriran: on ponavlja ono što prikazuje vaša web stranica, stupanj po stupanj. Snaga po bloku iz demodulatora polazište je za oboje, a stranica zatim primijeni `input.analog_smeter_offset`, raširi rezultat oko -130 dBm i doda vlastiti prikazni pomak. Most radi isto, pa Kiwi klijent i stranica pokazuju isti signal jednake jakosti — provjereno na jakoj lokalnoj postaji, gdje su oba pokazala -37 dBm.

Ako — i samo ako — želite da Kiwi klijenti pokazuju drukčije od vaše stranice, dajte mostu vlastiti pomak:

```
[kiwi_emulation]
enabled = true
smeter_offset = 5.0 # zamjenjuje input.analog_smeter_offset samo za Kiwi klijente
```

**Uspoređujte dva instrumenta na istom filtru, ili nikako.** Očitavanje dolazi iz snage u propusnom pojasu, pa širi filter skuplja više šuma i pokazuje više — na mirnom pojasu ista je frekvencija izmjerena 5,6 dB jače pri 9 kHz nego pri 2,4 kHz. Kiwi klijent sa 6 kHz filtrom naspram web stranice na 2,4 kHz razlikovat će se za nekoliko dB koliko god oba bila kalibrirana. Najprije uskladite način rada i širinu pojasa.

**Ovo je skala stranice, a ne fizička.** Web instrument širi svoj raspon radi čitljivosti, pa se 10 dB stvarne promjene signala prikazuje kao otprilike 11. Uskladiti se s njom znači da se Kiwi klijenti slažu s vašim prijamnikom i razlikuju od pravog KiwiSDR-a za iznos koji raste s jakošću signala. To je ispravan kompromis kada je vaša stranica mjerilo s kojim svi uspoređuju; ako biste radije da most ostane fizički pošten, ovo je odjeljak koji treba promijeniti.

## 8. Slap i spektar

### dB skala — wf\_cal

Svaki se bin slapa prije slanja pretvara u stvarne dBm, na istu skalu na koju je kalibriran S-metar. Nosilac stoga pokazuje **istu razinu pri svakom zumu**, što daje i pravi KiwiSDR, a spektar se slaže s vlastitim S-metrom web stranice preko istog propusnog pojasa.

```
[kiwi_emulation]
enabled = true
wf_cal = 0.0 # dB, samo Kiwi klijenti
```

**wf\_cal nije kalibracija.** 0 je vrijednost pri kojoj se spektar slaže s vašim S-metrom, i ondje ga ostavite ako želite da brojke nešto znače. Postoji zato što je pitanje ukusa — i onoga što vaš klijent radi s rasponom — koliko *jako* spektar treba izgledati: prijamnik ovog projekta radi na -10 jednostavno zato što tako izgleda dobro u AetherSDR-u. Jedna vrijednost pomiče spektar i slap zajedno: to su isti bajtovi na vodu, a Kiwi protokol ih ne može skalirati odvojeno. Želite li da se razlikuju, to mora doći iz klijentovih vlastitih prikaznih kontrola.

Za provjeru skale usporedite slap zbrojen preko propusnog pojasa sa S-metrom na **mirnoj** frekvenciji. Na nosiocu SSB propusni pojas isključuje sam nosilac i to dvoje nije usporedivo.

### Brzina sličica — wf\_fps\_max

Kiwi klijent traži najbrži slap koji može dobiti (SET wf\_speed=4) i ravna svoje klizanje — te usrednjavanje spektra — prema brzini koju poslužitelj obeća isporučiti. Ako se to dvoje ne poklapa, prikaz izgleda tromo iako zapravo ništa ne kasni.

Prijamnik proizvodi  $2 \times \text{sps} / \text{fft\_size}$  spektara u sekundi. Kiwi klijenti se poslužuju iz svakog od njih i prorjeđuju na brzinu koju su tražili; web stranica zadržava vlastiti, sporiji ritam i nije zahvaćena.

```
[kiwi_emulation]
enabled = true
wf_fps_max = 23 # sličica u sekundi
```

23 je maksimum koji objavljuje sam KiwiSDR protokol i sigurna zadana vrijednost. **Uvijek pobjeđuje manja od wf\_fps\_max i vlastite brzine prijarnika**, pa povećanje čini nešto samo ako je vaš FFT dovoljno brz da sličica pretekne. Primjer, za prijamnik na 60 Muzoraka/s s FFT-om od 4194304 točke —  $2 \times 60000000 / 4194304 = 28,6$  sličica u sekundi:

wf_fps_max	isporučeno
nema (zadano 23)	23,0 fps
28	28,0 fps
40	28,6 fps — pobjeđuje vlastita granica prijarnika

Poštuje se i SET wf\_speed s klijenta: 0 isključeno, 1 = 1 fps, 2 četvrtina maksimuma, 3 polovica, 4 maksimum. Klijent na slaboj vezi tako može tražiti manje.

**Hoće li povećanje pomoći odlučuje klijent, ne poslužitelj.** Klijent koji crta svaku sličicu čim stigne dat će vam glatkije klizanje; klijent koji se ravna prema 23 koje očekuje jednostavno će višak sličica staviti u red, a kašnjenje će rasti. Isprobajte, i ako slap djeluje tromije umjesto glatkije, vratite na 23.

## Što određuje ostatak kašnjenja

Ako lovite kašnjenje, većina ga nije u mostu. Prozor analize širok je `fft_size / sps` — 70 ms uz FFT od 4194304 točke na 60 Muzoraka/s — a uzorak uz to čeka do polovice toga da se korak napuni. Prepoloviti `fft_size` prepolovljuje oboje i udvostručuje brzinu sličica, ali i prepolovljuje vašu frekvencijsku razlučivost, od koje žive uskopojasni dekoderi (WSPR, FT8, CW). Na većini prijamnika to je loša zamjena za nekoliko desetaka milisekundi. To je odluka u `[input]`, a ne nešto što most može promijeniti.

## 9. Ako nešto ne izgleda dobro

Svaka naredba koju Kiwi klijent pošalje zapisuje se u `/tmp/kiwi_retune.log` — zahtjevi za ugađanjem, promjene načina rada, prijava i sve što most nije prepoznao. Ne košta ništa mjerljivo i namijenjeno je da ostane na mjestu. Kada se klijent ponaša čudno, ta datoteka pokazuje što je zapravo tražio, a to je obično cijeli odgovor.

Simptom	Pogledajte
Klijent se spoji i odmah ispadne	u konfiguraciji s kojom je poslužitelj <b>pokrenut</b> nedostaje <code>[kiwi_emulation] enabled = true</code>
Nema zvuka, nema slapa, nema redaka u zapisniku	pozadinski dio nije ponovno preveden nakon <code>kiwi_install.sh</code> — pokrenite <code>./recompile.sh</code> , opcija <code>[1]</code>
<code>kiwi_install.sh</code> stane na zakrpi	poruka imenuje datoteku i sidro; stablo je odstupilo od onoga što zakrpa očekuje, a ta je datoteka ostala netaknuta
Zvuk postoji, ali je tanak i tih	očekivano — postavite <code>audio_gain</code> , vidi <a href="#">odjeljak 6</a>
Slap klizi tromo	<code>wf_fps_max</code> , vidi <a href="#">odjeljak 8</a>
Ugađanje u klijentu ne radi ništa	<code>IQ</code> ulaz ( <code>signal = "iq"</code> ); ugađanje je izvedeno samo za realne ulaze
Promijenjena postavka nije imala učinka	ti se ključevi čitaju pri pokretanju — ponovno pokrenite prijamnik

**AGC je na strani klijenta.** `SET agc=` s Kiwi klijenta prihvaća se i zanemaruje — zvuk stiže s primijenjenim PhantomSDR-ovim vlastitim AGC-om, a klijentove AGC kontrole djeluju na ono što primi.

## 10. Dokle je provjereno

Format na vodu provjeren je naspram živog prijamnika i pročitani redak po redak naspram izvornog koda obaju klijenata kojima je namijenjen — AetherSDR-ovog `KiwiSdrProtocol.cpp` i kiwiclientovog `kiwi/client.py`. Raspored i veličina okvira, redni brojevi, redoslijed bajtova, skala S-metra, ugađanje i slap — sve se slaže. Za most je također **potvrđeno da radi sa samom AetherSDR desktop aplikacijom**, od strane održavatelja, naspram ovog prijamnika.

Izmjereno na tom prijamniku, s testnim klijentom na živim utičnicama: ugađanje se izvrši u 35–50 ms, zvuk teče u stvarnom vremenu bez zanošenja, a slap isporučuje postavljenu brzinu bez zastoja i bez izgubljenih sličica.

Ako ipak nešto izgleda pogrešno na vašoj instalaciji, `/tmp/kiwi_retune.log` je prvo mjesto za pogledati, a povratna je informacija dobrodošla u svakom slučaju.

**Vidi i:** [Vodič za instalaciju](#) · [Struktura projekta](#) · [Korisnički vodič](#)

## POGLAVLJE 14

# Diverziti prijam

RECEIVE\_DIVERSITY.md

## PhantomSDR-Plus — Diverziti prijam (Receive Diversity)

Povežite svoj prijamnik s **drugim prijamnikom na drugoj lokaciji** i slušajte onaj koji u tom trenutku ima bolji signal. Kad jedna lokacija upadne u fading, druga obično nije, pa prijam teče dalje.

Drugi prijamnik može biti još jedan PhantomSDR-Plus, KiwiSDR, UberSDR ili WebSDR. Za prva tri sve se izvodi u pregledniku: bez izmjena na poslužitelju, bez konfiguracijskih datoteka, bez ičega za instalirati i bez posebnog pristupa drugom prijamniku. WebSDR je iznimka i traži mali program na vašem vlastitom poslužitelju — vidi [Korištenje WebSDR-a](#) niže.

### Što radi, a što ne

Diverziti ovdje znači **odabir**: u svakom trenutku slušate jednu ili drugu lokaciju, uz kratko preklapanje pri promjeni. Nije riječ o faznom nizu i dva se signala ne zbrajaju.

To je namjeran izbor, a razlog je fizika, ne softver. Dva prijamnika udaljena stotinama kilometara čuju isti prijenos preko različitih ionosferskih putova. Dva zvučna vala stižu s nepovezanim fazom i malo drugačijim Dopplerom, pa njihovo zbrajanje zvuči šuplje i češljasto filtrirano — poznati „phasey” učinak. Pravo koherentno spajanje traži dva prijamnika na **zajedničkom taktu**, poravnata do uzorka; dva neovisna prijamnika preko interneta to nikada ne mogu pružiti.

Dobitak je stoga **kontinuitet, a ne jačina signala**:

- Na stabilnom putu gdje je jedan prijamnik jednostavno bolji, čut ćete njega i nećete dobiti ništa. To je normalno.
- Na putu s dubokim QSB-om, fadinzi na dvije udaljene lokacije uglavnom su nekorelirani. Lokacija A nestane na tri sekunde, lokacija B ne, i zvuk se nastavlja. Očekujte „*presta sam gubiti riječi*”, a ne „*iz S3 je postalo S7*”.
- Druga korist često je **lokalni šum i QRM**. Dvije lokacije imaju različite susjede, različite električne mreže i različiti splatter. Na bučnom opsegu to može značiti više od fadinga.

Ima svoju cijenu: slušate **jednu do tri sekunde iza stvarnog vremena**, jer raniji tok treba odgoditi da se poklopi s kasnijim. Nevažno za slušanje i dekodiranje, nezgodno ako pokušavate raditi sa stanicom.

#### NAPOMENA

Digitalni dekoderi (FT8, JS8, WSPR, RADE i ostali) namjerno i dalje koriste vaš **lokalni** prijamnik, a ne kombinirani zvuk. Ti načini rada integriraju koherentno kroz cijeli vremenski odsječak, a promjena lokacije usred odsjeka prekid je faze koji vas može stajati upravo onog dekodiranja koje je diverziti trebao spasiti. Diverziti služi zvučniku; dekoderi zadržavaju neprekinuti lokalni tok.

### Brzi početak

1. Otvorite ploču **Receive Diversity** — nalazi se odmah ispod prozora dekodera, sklopljena, s natpisom `Receive Diversity – off`. Kliknite da se raširi.

2. Odaberite što je drugi prijamnik: **PhantomSDR+**, **KiwiSDR**, **UberSDR** ili **WebSDR**.
3. Upišite njegovu adresu — ili odaberite jednu od spremljenih — i pritisnite **Start**.
4. Ugađajte svoj prijamnik normalno. Drugi ga **automatski slijedi** — frekvenciju, način rada i propusni pojas — pri svakom ugađanju.

To je cijela postava. Na drugom prijamniku nema se što podesiti niti postoji zasebno ugađanje: uvijek vas prati.

## Adrese

Drugi prijamnik	Što upisati	Napomene
PhantomSDR+	host:8900	Uobičajeni port prijamnika
KiwiSDR	host:8073	Vlastiti port KiwiSDR-a
UberSDR	host	Njegov <b>uobičajeni web port</b> , ne 8073
WebSDR	host:8901	Vlastiti port stanice. Traži relej — vidi niže

Prihvaća se samo naziv poslužitelja, `http://` adresa ili puna `ws://` adresa. Ako se drugi prijamnik poslužuje preko HTTPS-a, upišite naziv poslužitelja i veza će biti sigurna.

## Spremljeni prijammici

Svaka adresa koju upotrijebite pamti se, u vlastitom popisu za svaki tip prijamnika — adresa KiwiSDR-a nikad nije koristan prijedlog kad birač pokazuje WebSDR. Popisi ostaju u vašem pregledniku: ne šalju se nikamo i svaki slušatelj ima svoj.

Gumb  pokraj polja s adresom otvara popis.

- **Imena.** Svaki unos može dobiti ime — `Twente` se čita bolje od `websdr.ewi.utwente.nl:8901`. Imenovani unosi prikazuju se zeleno iznad svoje adrese, a dok tipkate ime stoji uz adresu u padajućem popisu. Unos se prepoznaje po svojoj adresi, pa imenovanje nikad ne stavi istu stanicu dvaput na popis. Dok diverziti radi, prijamnik na koji je spojen prikazuje svoje ime **podebljano i polako trepćuće**, pa popis govori i kojega zapravo slušate.
- **add** dodaje prijamnik ručno, bez prethodnog spajanja na njega.  ispravlja ime i adresu na mjestu,  uklanja jedan unos, a **clear all** prazni popis tipa koji je na zaslonu. Enter sprema, Escape odustaje.
- **\*\*\*\*** otvara vlastitu web stranicu tog prijamnika u novoj kartici. Spremljena je adresa koju poziva softver, pa se najprije pretvara u adresu za pregledavanje — `ws://` postaje `http://`, a putanja `/audio`, `/ws` ili, kod KiwiSDR-a, `/kiwi/.../SND` koja svakoj vrsti prijamnika treba, odbacuje se jer je stranica stanice u korijenu. To je pravi link, pa srednji klik i dugi pritisak rade kao i inače.
- **Redoslijed je vaš.** Pomaknite redak s  , ili ga povucite i pustite ondje gdje mu je mjesto. Tim redoslijedom padajući popis nudi adrese.
- **export i import.** Popisi su u spremištu samog preglednika, koje nestaje kad obrišete podatke stranice — upravo od toga ovo štiti. Export zapisuje sva četiri popisa, zajedno s imenima, u JSON datoteku koju možete sačuvati ili prenijeti u drugi preglednik ili na drugo računalo. Import je čita natrag i pita hoće li se **spojiti** (merge) sa spremljenim ili će se sve **zamijeniti** (replace).
-  **QR.** Crta sve popise kao QR kod na zaslonu, da ih prenesete na mobitel — vidi „Na mobitelu” niže. Popis prevelik za skeniranje vraća se na tekst pokraj njega, koji se može kopirati i zalijepiti bilo gdje.

Broj prijammika koje čuvate nije ograničen.

## Na mobitelu

Stranica `/mobile` ima istu funkciju u kartici **Div**, uz Audio, Bands, Marks, Users i Chat: vrsta prijamnika, adresa, Start i Stop, spremljeni prijammici kao popis koji se bira dodirrom, žive brojke i SNR trim. Svaki

redak nosi i link \*\*\*\* na vlastitu stranicu tog prijamnika. Drugi prijamnik sam slijedi ugađanje mobitela, jednako kao na stolnoj stranici. Nema uređivanja — preimenovanja, presloživanja, brisanja — ono ostaje na stolnoj stranici, gdje za to ima mjesta.

Spremljeni popis pripada jednom pregledniku, pa mobitel počinje s praznim koliko god prijamnika bilo spremljeno na računalu. Zato postoji QR kod: na računalu pritisnite ☰ pa  **QR**, skenirajte kod kamerom mobitela i zalijepite tekst u **Import** u kartici Div. Pita hoće li spojiti ili zamijeniti, a Export na mobitelu šalje popis u suprotnom smjeru.

## Korištenje WebSDR-a

WebSDR — softver Pieter-Tjerka de Boera, PA3FWM, koji radi u Twenteu i na više stotina drugih lokacija — može poslužiti kao drugi prijamnik, ali ne izravno iz preglednika.

Razlog je namjerna provjera na njihovoj strani. WebSDR odbija audio vezu ako zaglavlje `Origin` ne imenuje njegovu vlastitu stranicu, a `Origin` je *zabranjeno ime zaglavlja*: preglednik ga postavlja prema stranici na kojoj jeste i nijedna skripta ga ne smije mijenjati. Zaobilaznice na strani klijenta nema, niti bi je smjelo biti.

Zato vezu uspostavlja mali program na vašem poslužitelju, `websdr_relay.py`. Vaš preglednik razgovara s relejem, a relej s WebSDR-om.

## Instalacija releja

Sva četiri distribucijska instalatera nude ga kao neobavezan korak. Za već postojeću instalaciju:

```
cd ~/PhantomSDR-Plus
./setup_websdr_relay.sh
```

Traži port, preuzima vaš pozivni znak i adresu stanice iz `frontend/site_information.json` te nudi instalaciju systemd usluge kako bi se relej pokretao pri podizanju sustava. Postavke su u `websdr_relay.json` i mogu se promijeniti bilo kada.

**Jedno instalater ne može učiniti umjesto vas: proslijediti port releja na vašem ruteru.**

Preglednici slušatelja spajaju se izravno na relej — ne ide kroz prijamnik — pa bez tog prosljeđivanja WebSDR diverziti radi samo iz vaše vlastite mreže. Ostala tri tipa prijamnika to ne pogađa.

Ako je administracijska ploča instalirana, njezina nadzorna ploča prikazuje karticu **WebSDR Diversity Relay** sa stanjem, portom te brojem sesija i stanicama prema kojima idu.

## Budite dobar gost

Relej se na tuđe prijamnike spaja s vašeg poslužitelja, a ne s adrese svakog slušatelja, pa je građen da se pristojno ponaša:

- **Ograničenje od deset istodobnih sesija prema jednom WebSDR-u**, i šezdeset ukupno. To sprječava da užurban dan kod vas izgleda kao napad na tuđu stanicu.
- **User-Agent koji imenuje vašu stanicu i njezina operatora**, kako bi operator koji to ne želi znao točno kome pisati. Ispunite ga pošteno.
- **Prosljeđuju se samo naredbe ugađanja**. Veza se ne može koristiti za slanje bilo čega drugog.
- **Privatne, loopback i carrier-NAT adrese se odbijaju**, tako da posjetitelj ne može usmjeriti relej na nešto unutar vašeg računala ili lokalne mreže.

**VAŽNO**

Provjera Origin postoji jer autori WebSDR-a nisu htjeli da se njihovim prijateljima upravlja s tuđih stranica. Korištenje releja promišljena je odluka, a ne njihov propust. Ostavite ograničenje gdje jest, držite User-Agent poštenim i prestanite ako vas operator zamoli.

## Što je kod WebSDR-a drugačije

- **Mnoge stanice pokrivaju tek nekoliko uskih isječaka spektra**, a ne neprekinut raspon — primjerice prozor od 256 kHz na 40 m. Diverziti se uključuje samo unutar njih, a ploča čita popis opsega stanice da bi znala gdje su.
- **Frekvencija uzorkovanja zvuka mijenja se** sa širinom filtra i više odstupa od nazivne nego kod ostalih tipova, pa poravnanje može potrajati nešto dulje.
- **CW nije prilagođen** konvenciji WebSDR-a, koja propusni pojas stavlja u cijelosti ispod nosioca. SSB i AM su ispravni.

## Čitanje ploče

Tijekom rada ploča prikazuje malu tablicu:

Redak	Značenje
<b>link</b>	Veza s drugim prijateljem. <code>ready</code> je ono što želite. Pri neuspjehu prikazuje se tekst pogreške drugog poslužitelja i WebSocket kod zatvaranja
<b>aligned</b>	Izmjereno kašnjenje između dvaju tokova nakon zaključavanja, inače <code>searching</code>
<b>corr</b>	Koliko su tokovi korelirani. Ima smisla tek nakon poravnanja
<b>remote audio</b>	Dekodirani zvuk koji stiže s drugog prijatelja. <code>none</code> u jantarnoj boji znači da se ništa ne dekodira
<b>SNR local / remote</b>	Dvije procjene odnosa signal/šum koje se uspoređuju
<b>switches</b>	Koliko je puta promijenio lokaciju

Uz naslov, jedna riječ govori gdje ste:

<code>off</code> (sivo)	ne radi
<b>Please wait...</b> (jantarno)	spaja se, ili je spojen i još se poravnava
<b>Ready</b> (zeleno)	povezan, zaključan i prati bolju lokaciju
poruka pogreške samog prijatelja (crveno)	nije uspjelo i neće se samo oporaviti

Razlika između jantarne i crvene je ona koja štedi vrijeme: jantarna znači nastavite čekati, crvena znači prestanite čekati i pročitajte poruku.

Obojena točka dodaje koju lokaciju zapravo slušate — zelena vaš prijatelj, cijan udaljeni.

## Poravnanje traje oko 15 sekundi

To je normalno. Dva se toka poravnavaju koreliranjem njihovih **zvučnih ovojnica**, a za to treba prozor zvuka. Traženje ide u dva koraka: usko traženje od  $\pm 1,5$  sekunde može krenuti čim se prikupi oko 9 sekundi zvuka, a druga, neovisna mjera pet sekundi kasnije mora se **složiti** s prvom prije nego se poravnanje prihvati. Kad su dvije postaje vremenski razmaknutije od toga — sekunda ili dvije dodatnog

međuspremanja negdje na putu — na oko 14 sekundi preuzima puno traženje od  $\pm 4$  sekunde, a potvrda stiže na 19.

Taj korak potvrde je važan. Kad dvije lokacije blijede u protufazi — nikad istodobno ne nose signal — jedna jedina korelacija spremno se zaključa na uvjerljivo, ali posve pogrešno kašnjenje, što zvuči kao jeka. Zahtjev za dvjema neovisnim mjerama koje se slažu to odbacuje.

Još nešto mora biti točno prije nego zaključavanje izdrži: brzine uzorkovanja dvaju prijamnika. Dva prijamnika znače dva takta i dva lanca decimacije, pa im zvučni tokovi **ne** stižu posve istom brzinom čak ni kad oba prijavljuju 12 kHz — razlika od 2% je uobičajena, dakle 240 uzoraka klizanja u sekundi, daleko više od tolerancije poravnanja. Zato se udaljeni tok neprekidno preuzorkuje na brzinu vašeg prijamnika. Taj se omjer mjeri **brojanjem uzoraka**, a ne koreliranjem, pa mu ne treba vlastito zaključavanje i utvrđen je u prvih nekoliko sekundi — prije nego uopće krene prvo traženje poravnanja. Slab ili isprekidan signal može tražiti dva ili tri pokušaja, pa 30 sekundi nije razlog za brigu; minuta bez ičega jest.

Dok piše `searching`, obje vrijednosti SNR-a pokazuju 0.0. To je namjerno: obje se procjene mjere na **sadržajno poravnatim** uzorcima, pa se ništa ne mjeri prije nego postoji poravnanje. 0.0 / 0.0 znači „još nije zaključano“, a ne „nema signala“.

## Remote SNR trim

Ovaj klizač pristrano usmjerava izbor između dviju lokacija. Dodaje se izmjerenom SNR-u udaljenog prijamnika prije usporedbe — pozitivno pogoduje udaljenom, negativno vašem. Ne mijenja **ništa drugo**: ni razinu zvuka ni spajanje, samo koja lokacija pobjeđuje.

Postoji jer dvije vrijednosti SNR-a nisu uvijek usporedive. Objе se mjere jednako — kao raspon percentila zvuka — ali prijamnik čiji kodek podiže vlastiti prag šuma pokazuje manje nego što zaslužuje. KiwiSDR primjenjuje snažno pojačanje i limiter, što sažima raspon; kodek s gubicima postavlja prag šuma ispod kojeg signal nikad ne siđe.

### Kako ga podesiti:

1. Ugodite oba prijamnika na signal koji je pouzdano prisutan i pričekajte `aligned`.
2. Promatrajte obje vrijednosti SNR-a pola minute. Zapamtite uobičajenu razliku.
3. Krajnje položaje iskoristite kao slušni test: **+15** prisilno uključuje udaljenu lokaciju, **-15** vašu. Poslušajte svaku nekoliko sekundi. To je jedini način da čujete jednu lokaciju zasebno.
4. Ako udaljeni pokazuje, recimo, 5 dB manje ali zvuči jednako dobro, postavite **+5**. Time postizete da se dva očitavanja slažu kad obje lokacije zvuče jednako.
5. Pratite brojač **switches** kroz deset minuta slušanja. Neprestano prebacivanje znači da je vrijednost preblizu izjednačenju; nikakvo prebacivanje iako vaš prijamnik čujno blijedi znači da je otišla predaleko u drugom smjeru.

Polazne vrijednosti: **0** za drugi PhantomSDR-Plus, koji ima istovjetan zvučni put i usporediv je po samoj izvedbi; mala **pozitivna** vrijednost za KiwiSDR ili UberSDR.

### SAVJET

Ako vam treba više od otprilike  $\pm 8$  dB, prestanite podešavati. U tom je trenutku pošteno objašnjenje obično da je jedan prijamnik stvarno lošiji za taj put, a pristranost preko stvarne razlike znači samo slušanje slabije lokacije.

Postavka djeluje odmah i pamti se u vašem pregledniku.

## Odabir drugog prijamnika

**Udaljenost je važna.** Fadinzi se dekoreliraju s udaljenošću — nekoliko stotina kilometara na KV dobar je cilj. Dva prijamnika u istom gradu blijede zajedno i ne donose ništa. Predaleko, pa drugi prijamnik možda uopće ne čuje vaš signal.

**Oba moraju stvarno čuti signal.** Taj zahtjev nijedan softver ne zaobilazi. Ako drugi prijamnik ne čuje ono što vi slušate, korelacija ostaje niska i neće se zaključati — namjerno, jer pogrešno poravnanje zvuči gore nego nikakav diverziti.

**Pokrivenost.** Drugi PhantomSDR-Plus ili KiwiSDR objavljuju frekvencijski raspon koji pokrivaju, pa vas ploča upozorava ako ugodite izvan njega. WebSDR objavljuje svoje opsege, koji su često uski isječci umjesto neprekinutog raspona. UberSDR ne objavljuje pokrivenost, pa je ondje izostanak zaključavanja vaš jedini znak.

### VAŽNO

Javne prijamnike održavaju volonteri i imaju ograničen broj slušateljskih mjesta. Sesija diverzitija zauzima jedno dok traje, jednako kao i ljudski slušatelj. Budite obzirni prije nego ostavite vezu otvorenu neograničeno i pitajte operatora ako namjeravate njegov prijamnik koristiti intenzivno.

## Rješavanje problema

**Link nikad ne napusti connecting ili se stalno spaja i zatvara** Vjerojatno su adresa ili port pogrešni. Pogledajte tablicu gore — posebno, UberSDR koristi svoj uobičajeni web port, ne 8073. Ploča prikazuje WebSocket kod zatvaranja, a konzola preglednika (F12) osnovnu pogrešku.

**Link prikazuje pogrešku s tekstom** Taj tekst dolazi od drugog prijamnika i obično je konkretan: odbijena sesija, pun poslužitelj ili prijamnik koji ne prihvata vanjske klijente.

**KiwiSDR odbija vezu** Neki KiwiSDR-ovi traže lozinku, a na UberSDR-u je pristupna točka kompatibilna s KiwiSDR-om prema zadanim postavkama isključena. Za UberSDR upotrijebite vrstu **UberSDR** — taj je put uvijek dostupan.

**WebSDR: the WebSDR relay is not reachable** Relej ne radi ili vaš preglednik ne doseže njegov port. Mora biti dostupan na **istom imenu poslužitelja s kojeg se poslužuje stranica prijamnika**, jer se preglednik spaja izravno na njega, a ne kroz prijamnik. Otvorite `http://<to ime poslužitelja>:<port releja>/status` u istom pregledniku da vidite o čemu se radi. Ako odgovara na lokalnoj mreži, ali ne izvana, port nije proslijeđen na ruteru.

**WebSDR: pogreška koja spominje odbijenu vezu** Ta je stanica zaključana strože od uobičajene provjere `Origin`, ili je njezin operator blokirao ovu stanicu.

**Link je ready, ali remote audio pokazuje none** Zvuk se ne dekodira. Osvježite stranicu; ako se nastavi, konzola preglednika pokazat će koji je dekodir zakazao.

**Zvuk stiže, ali se nikad ne poravna** Dva prijamnika ne čuju istu stvar. Pokušajte s jakim radiodifuzijskom postajom ili opsegom na kojem obje lokacije imaju dobru propagaciju. Miran opseg sa samo šumom na obje strane nikad se neće korelirati — i ne bi ni trebao.

**Stalno se prebacuje amo-tamo** Dvije su lokacije preblizu po SNR-u. Malo pomaknite trim tako da jedna bude dosljedno povlaštena.

**Nikad ne koristi drugi prijamnik** Provjerite raste li `remote audio` i djeluju li vrijednosti SNR-a razumno. Ako udaljeni pokazuje mnogo manje nego što zvuči, upravo za to služi trim.

## Ograničenja

---

- **Nije koherentno.** Bez fazriranja, bez poništavanja, bez radiogoniometrije. Poništavanje izvora šuma traži dvije antene na istom taktu na istoj lokaciji; to je druga tehnika za drugi problem.
- **Jedna do tri sekunde kašnjenja**, neizbježno.
- **Samo dva prijamnika.**
- **Mono.** Ako je vaš prijamnik u stereo načinu poput C-QUAM-a, zvuk prolazi nepromijenjen i diverziti se ne uključuje.
- **Poravnanje treba signal.** Ispod otprilike 10 dB SNR-a na jednoj od dviju lokacija očekujte da ostane na `searching`.
- **WebSDR traži relej** na vašem vlastitom poslužitelju i proslijeđen port. Ostala tri tipa prijamnika ne trebaju ni jedno ni drugo.

## POGLAVLJE 15

# Upravljanje primopredajnikom

RIG\_CONTROL.md

## PhantomSDR-Plus — Upravljanje primopredajnikom (CAT)

Držite **vlastiti primopredajnik** i **PhantomSDR-Plus prijemnik** na istoj frekvenciji, načinu rada i filtru. Okrenite gumb za ugađanje na uređaju i slap ga slijedi; kliknite signal na slapu i uređaj se ugodi na njega. Kad odašiljete, prijemnik može utihniti kako vam ne bi vraćao vlastiti signal.

Radi s prijemnicima **PhantomSDR-Plus**, **KiwiSDR**, **PA3FWM WebSDR** i **UberSDR**, vašima ili tuđima, i uvijek pomiče samo *vašu* sesiju slušanja — nitko drugi na prijemniku ništa ne čuje ni ne vidi. Operater prijemnika ne mora ništa instalirati ni podešavati.

### Što vam treba

Dva načina povezivanja uređaja i jedan uvjet na strani prijemnika:

Dio	Što je	Sinkronizira
<b>Desktop PhantomSDR+ 4.0 ili noviji</b>	Desktop aplikacija s izbornikom <b>Rig</b> . Linux (PC i Raspberry Pi) i Windows.	Frekvenciju, način rada, širinu filtra, utišavanje pri odašiljanju — u jednom ili oba smjera
<b>CATsync Tool for WebSDRs</b>	Zaseban Windows program koji povezuje uređaj sa stranicom prijemnika u vašem pregledniku.	Frekvenciju i način rada
<b>TCI-CAT na stranici prijemnika</b>	Gumb na samoj PhantomSDR-Plus stranici, u bilo kojem pregledniku. Govori TCI s ExpertSDR-om, AetherSDR-om ili Thetisom, ili s bilo kojim uređajem koji podržava Hamlib preko malog mosta. Samo na PhantomSDR-Plus prijemnicima s inačicom 4.1.0 ili novijom.	Frekvenciju, način rada i širinu filtra u oba smjera; utišavanje pri odašiljanju
<b>Prijemnik</b>	PhantomSDR-Plus, KiwiSDR, PA3FWM WebSDR ili UberSDR. PhantomSDR-Plus prijemnik treba <b>4.1.0</b> ili noviju za širinu filtra i utišavanje.	Stariji PhantomSDR-Plus i dalje sinkronizira frekvenciju i način rada

Veći dio ovog priručnika opisuje Desktop PhantomSDR+. Gumb TCI-CAT ima vlastiti odjeljak, [TCI-CAT na stranici prijemnika](#). CATsync Tool ima vlastitu dokumentaciju na svojoj web stranici.

### Podržani prijemnici

Aplikacija prepoznaje vrstu prijemnika u prozoru stanice i upravlja njime preko vlastitih kontrola te stanice. Prozor Rig control prikazuje pronađenu vrstu pokraj naziva stanice.

Prijemnik	Frekvencija i način rada	Širina filtra	Utišavanje pri odašiljanju
PhantomSDR-Plus	Da	S 4.1.0 ili novijom	S 4.1.0 ili novijom
KiwiSDR (uključujući Web-888)	Da	Da	Da
PA3FWM WebSDR	Da, uz promjenu opsega na stranicama s više opsega	Da	Da
UberSDR	Da	Da	Da

Web prijemnik ima manje načina rada od većine uređaja, pa neki načini uređaja dijele jedan način prijemnika: KiwiSDR i WebSDR imaju jedan CW za CW i CW-R. Načini prijemnika navedeni u ovom priručniku su PhantomSDR-Plusovi; ostali prijemnici koriste najbliži ekvivalent. WebSDR drži CW filtar ispod 1 kHz, a ostale filtre na 1 kHz ili šire, jer tako sama stranica prepoznaje CW, a frekvencija izvan svih opsega WebSDR stranice ostavlja se na miru. UberSDR podešava filtar u koracima klizača, pa širina može odstupati do oko 50 Hz od širine na uređaju. Svaka druga vrsta stranice — primjerice OpenWebRX — prikazuje *not a receiver this app can drive* i ništa se ne sinkronizira.

## Što radi, a što ne

- Sinkronizira **jedan uređaj s jednim prozorom prijemnika** istodobno.
- Čita obje strane nekoliko puta u sekundi i, kad se razlikuju, jednu podesi prema drugoj. **Ne** odašilje, ne uključuje PTT i nikamo ne šalje zvuk.
- Pomicanje prijemnika mijenja samo vašu sesiju. Ostali slušatelji istog prijemnika to ne osjete, a operater to ne mora dopustiti.
- Serijski port može otvoriti samo **jedan program istodobno**. Ako WSJT-X, dnevnik veza ili alat proizvođača već drži port, upotrijebite izbor **flrig** ili **rigctld on network** kako biste dijelili uređaj umjesto da se programi otimaju za port.

## Brzi početak

1. Otvorite stanicu u Desktop PhantomSDR+ kao i obično.
2. **Rig → Rig control...**
3. Pod **Connection** odaberite **Built-in** ako je vaš uređaj na popisu, inače **Hamlib (all rigs)**.
4. Odaberite uređaj, serijski port i brzinu na koju je podešen CAT ili CI-V izbornik uređaja.
5. Pod **Sync** ostavite odabrano **Both directions**.
6. Pritisnite **Connect**. Dva prikaza na vrhu — primopredajnik i prijemnik — trebali bi unutar sekunde pokazati istu frekvenciju.

Svaka se postavka sprema čim je promijenite. Sljedeći put dovoljno je **Rig → Connect**, ili označite **Connect when the app starts**.

## Odabir načina povezivanja

Izbor	Koristite kada	Treba
<b>Built-in</b>	Vaš je uređaj na popisu ispod.	Ništa drugo
<b>Hamlib (all rigs)</b>	Vaš je uređaj bilo što drugo — Hamlib poznaje više od 300. Aplikacija sama pokreće Hamlibov <code>rigctld</code> na privatnom lokalnom portu i zaustavlja ga pri odspajanju.	Windows: ništa, Hamlib je uključen. Linux: <code>sudo apt install libhamlib-utils</code>
<b>rigctld on network</b>	Već radi <code>rigctld</code> , na ovom ili drugom računalu u vašoj mreži.	Host i port (zadano 4532)
<b>flrig</b>	flrig već upravlja uređajem za fldigi, WSJT-X ili dnevnik veza.	Pokrenut flrig s njegovim XML-RPC portom (zadano 12345)

## Uređaji s ugrađenim upravljačem

Prikazane brzine i CI-V adrese tvorničke su vrijednosti koje aplikacija upisuje. **To je samo polazište — postavite ih prema izborniku svog uređaja.**

Obitelj	Uređaji	Zadana brzina	Napomene
<b>Icom CI-V</b>	IC-7300, IC-7610, IC-705, IC-9700, IC-905, IC-7760, IC-7851, IC-7100, IC-7410, IC-9100, IC-7600, IC-7200, IC-7700, IC-7000, IC-7800, IC-756PROIII, IC-756PROII, IC-R8600 i svaki drugi CI-V uređaj	19200	CI-V adresa upisana po modelu (IC-7300 94, IC-705 A4, IC-9700 A2, IC-7610 98 ...)
	IC-746PRO, IC-718, IC-R75	9600	
<b>Xiegu (CI-V)</b>	G90, X6100	19200	Adresa 70; provjerite izbornik
<b>Yaesu novi CAT</b>	FTDX101D/MP, FTDX10, FT-710, FT-991/A, FT-891, FTDX5000, FTDX3000, FTDX1200, FT-950, FT-2000, FT-450/450D	38400	
<b>Yaesu klasični CAT</b>	FT-817/818, FT-857/857D, FT-897/897D	38400	2 stop bita; ugađa u koracima od 10 Hz
<b>Kenwood</b>	TS-990S, TS-890S, TS-590S/SG	115200	
	TS-480, TS-2000, TS-870S	57600	
<b>Elecraft</b>	K4, K3/K3S, KX3, KX2	38400	Sinkronizira se i širina filtra
<b>Kompatibilni s Kenwoodom</b>	FlexRadio SmartSDR CAT (virtualni port), QRP Labs QMX/QMX+/QDX, (tr)uSDX, Lab599 Discovery TX-500, ostali Kenwood-kompatibilni uređaji	9600–38400	

Uređaj koji bi trebao biti kompatibilan, a ne razgovara s ugrađenim upravljačem, obično radi s **Hamlibom**, koji podnosi mnogo više inačica.

## Postavke serijskog porta

Postavka	Što upisati
<b>Serial port</b>	Port uređaja. USB adapteri i uređaji s USB portom navedeni su prvi. <b>Other / network address...</b> prima port kojeg nema na popisu — COM7, /dev/ttyUSB1 — ili tcp://host:port za serijski port koji preko mreže posluhuje ser2net ili slično.
<b>Speed (baud)</b>	Točno ono što piše u CAT / CI-V izborniku brzine uređaja. Pogrešna brzina izgleda kao uređaj koji nikad ne odgovara.
<b>Stop bits</b>	1 za gotovo sve; 2 za obitelj FT-817/857/897.
<b>CI-V address</b>	Samo Icom, heksadecimalno (94, ne 148). Mora odgovarati izborniku CI-V adrese.
<b>DTR / RTS</b>	Ostavite <b>isključeno</b> osim ako ih vaše sučelje treba. Mnogi CAT kabeli na jednoj od tih linija uključuju odašiljač ili resetiraju uređaj.
<b>Hardware flow control</b>	Isključeno, osim ako priručnik traži RTS/CTS.

Za **Hamlib** iste se postavke proslijeđuju `rigctld`-u. Stop bitovi tamo imaju i izbor *Rig default*, a **Extra rigctld options** prima sve ostalo što `rigctld` razumije, npr. `--set-conf=post_write_delay=10`. **rigctld program** pokazuje na određeni `rigctld` ako ih je instalirano više.

## Sinkronizacija

### Smjer

Izbor	Što se događa
<b>Rig → receiver</b>	Prozor prijemnika slijedi uređaj. Promjena na slapu vraća se na frekvenciju uređaja.
<b>Receiver → rig</b>	Uređaj slijedi prozor prijemnika. Okretanje gumba na uređaju poništava se.
<b>Both directions</b>	Pobjeđuje strana koju ste <b>zadnju</b> dotaknuli. U trenutku spajanja, prije nego što je ijedna dotaknuta, pobjeđuje uređaj.

Smjer se može promijeniti i iz izbornika **Rig** tijekom veze.

### Kako se sprječava da se strane „svađaju”

Svaka vrijednost koju aplikacija upiše pojavi se trenutak kasnije kao promjena na drugoj strani. Kad bi se to shvatilo doslovno, uređaj i prijemnik beskonačno bi se ganjali. Aplikacija to izbjegava na tri načina:

- Stranica prijemnika primjenjuje promjenu odmah, pa se čita odmah nakon upisa i to očitavanje postaje novo polazište.
- Uređaj primjenjuje promjenu nešto kasnije, pa se svaka poslana vrijednost pamti. Kad uređaj javi tu vrijednost, prepoznaje se kao upis aplikacije, a ne kao ruka na gumbu.
- Vrijednost koju uređaj odbije — npr. široki FM na KV uređaju — šalje se **dvaput**, a zatim se ostavlja na miru dok se izvorna strana ne promijeni, umjesto da se ponavlja nekoliko puta u sekundi.

Ponovno ugađanje stranice prijemnika može je navesti da odabere zadani način rada za opseg (npr. LSB ispod 10 MHz). Kad vodi uređaj, aplikacija odmah vraća njegov način rada, pa uređaj u USB-u na 40 m drži i prijemnik u USB-u.

## Koji prozor prijemnika

**Receiver window** bira koja stanica slijedi uređaj:

- **The station window last in front** (zadano) — s dvije otvorene stanice kliknite u jednu i uređaj slijedi nju.
- **Određena stanica** — uređaj ostaje vezan uz nju, bila ona u prvom planu ili ne. Ako ta stanica nije otvorena, ništa se ne sinkronizira dok se ne otvori.

## Brzina osvježavanja

**Update every** određuje koliko se često čitaju obje strane: 150 ms, 300 ms (zadano), 500 ms ili 1 s. Brže se na gumbu osjeća neposrednije; sporije je blaže prema starom uređaju na 4800 ili 9600 bauda, gdje svako čitanje traje stvarno vrijeme na liniji.

## Načini rada

Način na uređaju	Prijemnik sluša u
USB, LSB	USB, LSB
CW	CW
CW-R (obrnuto)	CW-L
AM, sinkroni AM, DSB	AM
FM, uski FM	FM
Široki FM	WBFM
RTTY / FSK	LSB
RTTY-R / FSK-R	USB
Podatkovni načini (USB-D, DATA-U, PKTUSB, DIG)	USB
Podatkovni LSB, podatkovni FM	LSB, FM

Način na prijemniku	Uređaj se postavlja na
USB, LSB	USB, LSB
CW	CW
CW-L	CW-R
AM, QUAM	AM
FM	FM
WBFM	WFM — većina KV uređaja to odbija i ostavljaju se na miru nakon dva pokušaja
RADE (gornji / donji)	USB / LSB

Uređaj u **podatkovnom načinu** u njemu i ostaje: USB prijemnika smatra se usklađenim s USB-D uređaja, pa prijemnik nikad ne izbacuje uređaj iz podatkovnog načina.

## Širina filtra

Označite **Sync filter width** da se propusni pojasevi poklapaju. Razlike manje od 60 Hz smatraju se jednakima jer nikoja dva filtra nemaju iste korake.

Veza s uređajem	Širina filtra
Hamlib	Da, gdje ga Hamlib podržava za taj uređaj
flrig	Da
Ugrađeni Icom CI-V	Da — koraci od 50 Hz do 500 Hz, zatim od 100 Hz do 3,6 kHz; AM u koracima od 200 Hz do 10 kHz; ne u FM-u
Ugrađeni Elecraft	Da, u koracima od 10 Hz
Ugrađeni Kenwood, Yaesu, obitelj FT-817	Ne — ti uređaji biraju filtre iz tablica za pojedini model. Za filter koristite Hamlib

Prijemnici KiwiSDR, WebSDR i UberSDR uvijek imaju tu kontrolu. PhantomSDR-Plus prijemnik mora imati **4.1.0** ili noviju; na starijem se frekvencija i način rada i dalje sinkroniziraju, a prozor Rig control objašnjava zašto filter ne slijedi.

## Utišavanje pri odašiljanju

Označite **Mute receiver while transmitting**. Dok uređaj odašilje, prozor prijemnika je utišan, a kad prestane, zvuk se vraća. Gumb za utišavanje prijemnika to pokazuje i zvuk uvijek možete vratiti ručno.

Ako ste prijemnik već sami utišali, ostaje utišan i nakon toga.

Treba vezu koja javlja stanje odašiljanja — svi ugrađeni upravljači, flrig i Hamlib za većinu uređaja — i, na PhantomSDR-Plus prijemniku, inačicu 4.1.0 ili noviju.

## Pomak frekvencije

**Frequency offset** dodaje se frekvenciji uređaja da bi se dobila frekvencija prijemnika:

frekvencija prijemnika = frekvencija uređaja + pomak

Postava	Pomak
Transverter za 2 m na uređaju za 10 m (144,100 MHz prikazuje se kao 28,100 MHz)	116000000
Transverter za 70 cm na uređaju za 2 m (432 → 144)	288000000
Bez transvertera	0

## Izbornik Rig

Stavka	Što radi
<b>Rig control...</b>	Otvora prozor Rig control
<b>Connect / Disconnect</b> <i>naziv uređaja</i>	Pokreće ili zaustavlja sinkronizaciju; za Hamlib pokreće ili zaustavlja i rigctld
<b>Rig to receiver / Receiver to rig / Both directions</b>	Smjer sinkronizacije
<b>Sync filter width</b>	Da / ne
<b>Mute receiver while transmitting</b>	Da / ne
Statusni redak	<i>Not connected, Connecting..., Connected: naziv</i> , ili posljednja pogreška

Živi prikaz frekvencije nalazi se u prozoru Rig control, a ne u izborniku, koji bi se inače sam zatvarao pri svakoj promjeni.

## Linux

**Dozvola za serijski port.** Serijski portovi pripadaju grupi  `dialout` . Korisnik izvan nje dobiva *Could not open ttyUSB0*. Dodajte se jednom, zatim se odjavite i ponovno prijavite:

```
sudo usermod -aG dialout $USER
```

**Hamlib.** Instalirajte ga iz svoje distribucije:

```
sudo apt install libhamlib-utils
```

Paket `.deb` za Desktop PhantomSDR+ ga preporučuje, pa ga `sudo apt install ./phantomsdr-plus-desktop_4.0.0_amd64.deb` donosi sa sobom; `dpkg -i` ne instalira preporučene pakete. Ugrađeni upravljači i flrig ne trebaju Hamlib.

## Windows

Hamlibov vlastiti `rigctld.exe` uključen je i u 64-bitni i u 32-bitni instalacijski program. COM portovi pojavljuju se na popisu pod svojim nazivom (COM3). Ako uređaj treba USB upravljački program, najprije instalirajte onaj proizvođača — do tada port ne postoji.

## TCI-CAT na stranici prijemnika

Stranica PhantomSDR-Plus prijemnika može pratiti primopredajnik i **sama**, bez desktop aplikacije: preglednik izravno razgovara s **TCI** poslužiteljem na vašem vlastitom računalu. TCI je WebSocket protokol za upravljanje programa ExpertSDR2/ExpertSDR3 (SunSDR), AetherSDR (FlexRadio) i Thetis (Apache Labs ANAN, Hermes). Za uređaj bez TCI-ja, mali most koji dolazi s PhantomSDR-Plus čini da bilo koji Hamlib uređaj izgleda kao TCI poslužitelj.

Potreban je PhantomSDR-Plus prijemnik s inačicom **4.1.0 ili novijom**. Stranice KiwiSDR, WebSDR i UberSDR to nemaju.

## Gumb TCI-CAT

Gumb je u redu s **VFO**, **Modes**, **Bands** i **IF Filters** — **TCI-CAT** u širokom rasporedu, **CAT** u sažetom. Njegova točka je **zelena** dok je TCI poslužitelj spojen, a inače **siva**. Otvara prozor s: Do verzije 4.2.0 gumb se zvao **QRG Sync**.

Kontrola	Što radi
Stanje	<i>Active</i> — TCI (port 50001) kad je spojeno; <i>Inactive</i> s <i>Wait</i> dok traži
<b>CAT Sync</b>	Uključuje ili isključuje sinkronizaciju frekvencije, načina rada i filtra. Utišavanje pri odašiljanju radi u oba slučaja
<b>Host</b>	Računalo na kojem radi TCI poslužitelj: <code>localhost</code> za ovo računalo, inače njegova adresa u lokalnoj mreži, npr. <code>192.168.1.42</code> . Pritisnite Enter ili kliknite izvan polja za ponovno spajanje

Ništa drugo ne treba birati: portovi **50001** (ExpertSDR3, AetherSDR) i **40001** (ExpertSDR2, Thetis) pokušavaju se istodobno i ponovno svakih nekoliko sekundi, pa se stranica spoji čim se program pokrene. **CAT Sync** i **Host** preglednik pamti.

Preglednici koji traže dopuštenje prije nego što web stranica pristupi lokalnoj mreži — to rade novije inačice Chromea i Edgea — pitaju jednom, pri prvom spajanju. Dopustite, inače točka ostaje siva.

## Što se drži usklađenim

	Ponašanje
<b>Uređaj → prijemnik: frekvencija</b>	Pomiče se samo skala. Zoom, svjetlina i kontrast ostaju kako ste ih ostavili; vodopad se pomiče samo kad frekvencija izađe iz prikaza, a plan opsega ne mijenja način rada
<b>Uređaj → prijemnik: način rada</b>	Prati kad se promijeni način rada uređaja — USB, LSB, CW, AM, FM; digitalni načini kao USB ili LSB. Način koji prijemnik nema ostavlja ga nepromijenjenim. Dekoder koji radi zadržava svoj način
<b>Uređaj → prijemnik: širina filtra</b>	Propusni opseg prijemnika preuzima širinu filtra uređaja: promjena filtra ili tipka <b>FIL</b> na uređaju mijenja propusni opseg. Dekoder koji radi zadržava svoj propusni opseg
<b>Prijemnik → uređaj: frekvencija</b>	Upisana frekvencija, oznaka ili knjižna oznaka, klik na vodopad i povlačenje propusnog opsega pomiču VFO uređaja. Pri povlačenju šalju se samo promijenjene frekvencije
<b>Prijemnik → uređaj: način rada</b>	Svaka promjena načina rada na stranici — tipka načina, plan opsega, dekode, knjižna oznaka — postavlja način rada uređaja (USB, LSB, CW, AM, FM; RADE kao USB ili LSB). Uređaj u digitalnom načinu (USB-D) ostaje u njemu
<b>Prijemnik → uređaj: širina filtra</b>	Tipke <b>IF Filters</b> , IF klizač, povlačenje propusnog opsega i promjena načina rada postavljaju filter uređaja; pri povlačenju šalje se samo konačna širina. Na Icom uređajima most umjesto toga odabire FIL1, FIL2 ili FIL3 — vidi <a href="#">Icom filtri</a>
<b>Bez jeke</b>	Promjena koja je došla s uređaja nikad mu se ne vraća, a javljanje prethodnog načina ili filtra uređaja koje stigne odmah nakon promjene na stranici zanemaruje se
<b>Utišavanje pri odašiljanju</b>	Uvijek uključeno, čak i s isključenim CAT Sync: stranica utihne dok uređaj odašilje. Pomicanje klizača glasnoće tijekom odašiljanja ne vraća zvuk; pri prijemu zvuk se vraća na trenutni položaj klizača
<b>Nije podržano</b>	Split, VFO B, RIT/XIT i pomaci transvertera. Prati se samo VFO A prvog prijemnika

Način rada i širina filtra šalju se vlastitim TCI naredbama (`modulation`, `rx_filter_band`), pa bi ih i ExpertSDR, AetherSDR i Thetis trebali pratiti; isproban je samo Hamlib most. Za pomak transvertera

koristite Desktop PhantomSDR+. Ne koristite oboje istodobno na istom uređaju — dva upravljača bore se oko skale.

## S kojim primopredajnicima radi

Primopredajnik	Radi	Kako
SunSDR (ExpertSDR2/3), FlexRadio (AetherSDR), ANAN/Hermes (Thetis)	Da	Izravno — uključite TCI poslužitelj u programu. Još nije isprobano s tim programima; testirano sa simuliranim TCI poslužiteljem
Uređaj s CAT portom koji Hamlib podržava — većina Icom, Yaesu, Kenwood, Elecraft, Xiegu, QRP Labs	Da	Preko Hamlib mosta opisanog dolje. Isprobano s <b>Icom IC-7300</b> na Linuxu i Windowsima; drugi upravljački programi mogu se razlikovati u nazivima načina rada ili javljanju PTT-a
Uređaj bez CAT porta, ili koji Hamlib ne podržava	Ne	Nema odakle pročitati frekvenciju

## Hamlib most (IC-7300 i drugi uređaji bez TCI-ja)

`tci-bridge/tci-rigctld.mjs` u PhantomSDR-Plus stablu pretvara svaki uređaj kojim Hamlib može upravljati u TCI poslužitelj za stranicu. Radi na **vašem** računalu — onom spojenom na uređaj, uz preglednik — ne na prijemniku. Četiri puta u sekundi pita Hamlib za frekvenciju, način rada, širinu filtra i stanje odašiljanja, šalje stranici samo promjene i prosljeđuje uređaju promjene frekvencije, načina rada i filtra sa stranice.

Do Hamliba dolazi na jedan od dva načina:

Način	Lanac	Kada
<code>--rigctl</code> (preporučeno)	primopredajnik → <code>rigctl</code> → <code>tci-rigctld.mjs</code> → stranica	Uobičajeno. Most sam pokreće Hamlibov <code>rigctl</code> : jedan prozor, bez mrežnog porta. <b>Koristite ga na Windowsima</b> , gdje se <code>rigctld.exe</code> često odbija porukom <i>Access is denied</i>
<code>rigctld</code>	primopredajnik → <code>rigctld</code> → <code>tci-rigctld.mjs</code> → stranica	Drugi program, poput WSJT-X-a, mora istodobno koristiti uređaj — vidi <a href="#">Dijeljenje uređaja</a>

## Što mu treba

	Linux	Windows
Hamlib	<code>sudo apt install libhamlib-utils</code> , ili hamlib paket vaše distribucije	<code>hamlib-w64-....zip</code> s <a href="https://github.com/Hamlib/Hamlib/releases">github.com/Hamlib/Hamlib/releases</a> , raspakiran u <code>C:\hamlib</code> — <code>rigctl.exe</code> je u <code>C:\hamlib\bin</code>
Node.js	18 ili noviji	LTS <i>Windows Installer (.msi)</i> s <a href="https://nodejs.org">nodejs.org</a> , sa zadanim opcijama
Paket <code>ws</code>	Unutar PhantomSDR-Plus stabla pronalazi se automatski	Kopirajte <code>tci-rigctld.mjs</code> u mapu poput <code>C:\tci-bridge</code> i ondje jednom pokrenite <code>npm install ws</code>
Port uređaja	<code>ls /dev/serial/by-id/</code> . Jednom se pridružite grupi <code>dialout</code> : <code>sudo usermod -aG dialout \$USER</code> , zatim se odjavite i ponovno prijavite	Instalirajte USB upravljački program proizvođača; COM broj je u <b>Upravitelj uređaja</b> → <b>Priključci (COM i LPT)</b>

Samo jedan program može držati CAT port uređaja. Prije pokretanja mosta zatvorite WSJT-X, flrig, JS8Call, RS-BA1 ili vezu s uređajem u Desktop PhantomSDR+.

### Primjer: Icom IC-7300

Na uređaju: **MENU → SET → Connectors → CI-V — CI-V USB Baud Rate 115200, CI-V Transceive ON**. Hamlibov broj modela za IC-7300 je 3073.

**Linux.** Uređaj je redak koji sadrži IC-7300 u `ls /dev/serial/by-id/`, obično i `/dev/ttyUSB0`.

1. Provjerite da Hamlib dolazi do uređaja — mora ispisati frekvenciju, npr. 14280000: `bash rigctl -m 3073 -r /dev/ttyUSB0 -s 115200 f`
2. Pokrenite most i ostavite ga da radi: `bash cd ~/PhantomSDR-Plus/tci-bridge node tci-rigctld.mjs --rigctl rigctl -m 3073 -r /dev/ttyUSB0 -s 115200`

**Windows.** Uređaj se u Upravitelju uređaja prikazuje kao *Silicon Labs CP210x USB to UART Bridge (COM4)* — koristite svoj COM broj. U naredbenom retku:

1. Provjerite da Hamlib dolazi do uređaja — mora ispisati frekvenciju: `bat C:\hamlib\bin\rigctl.exe -m 3073 -r COM4 -s 115200 f`
2. Pokrenite most i ostavite prozor otvoren: `bat cd C:\tci-bridge node tci-rigctld.mjs --rigctl C:\hamlib\bin\rigctl.exe -m 3073 -r COM4 -s 115200` Redak je dugačak: provjerite da stvarno završava s `-s 115200`, inače se `rigctl` zaustavlja porukom *Type: rigctl --help*.

Na oba sustava most unutar sekunde ispiše:

```
rig answered through rigctl
rig -> page 14280000 Hz
rig -> page mode USB
rig -> page RX
```

Zatim otvorite stranicu prijemnika u pregledniku na istom računalu, otvorite **TCI-CAT** i uključite **CAT Sync**. Točka postaje zelena, a most ispiše `page connected`. Okrenite VFO i prijemnik prati; kliknite na vodopad i uređaj prati (`page -> rig ... Hz`); prijedite na odašiljanje i stranica utihne. **Ctrl+C** zaustavlja most, a s njim i `rigctl`.

Sve iza `--rigctl` je program `rigctl` s vlastitim opcijama — upravo onima koje su radile pri provjeri. To je pravilo za svaki uređaj: **kad `rigctl ... f` ispiše frekvenciju, most radi s istim opcijama.**

### Icom filtri (FIL1, FIL2, FIL3)

Icom uređaji poput IC-7300 ne prihvaćaju bilo koju širinu filtra: imaju tri filtra, **FIL1**, **FIL2** i **FIL3**, svaki sa širinom postavljenom u izborniku uređaja. S nekom širinom Hamlib bi odabrao jedan od njih i ujedno prepisao njegovu širinu — a na IC-7300 ta širina može završiti na prethodno odabranom filtru i pomiješati postavke. Zato na Icom uređajima most nikad ne šalje širinu: uzima filtar čija je referentna širina najbliža propusnom opsegu stranice i samo ga **odabire**, vlastitom CI-V naredbom uređaja. Širine postavljene na uređaju nikad se ne mijenjaju.

Način rada	FIL1	FIL2	FIL3
USB, LSB (i njihovi digitalni načini)	2700 Hz	2400 Hz	1800 Hz
CW	1200 Hz	500 Hz	250 Hz
AM	9000 Hz	6000 Hz	3000 Hz
FM	15000 Hz	10000 Hz	7000 Hz

Na IC-7300 (`-m 3073`) to je automatski. Postavite SSB filtre uređaja odgovarajuće — FIL1 2,7 kHz, FIL2 2,4 kHz, FIL3 1,8 kHz (držite **FIL** na uređaju) — pa odabir 2,7 / 2,4 / 1,8 kHz u **IF Filters** odabire FIL1 / FIL2 / FIL3, a pritisak na **FIL** na uređaju postavlja propusni opseg stranice na širinu tog filtra. Most ispisuje, na primjer, `page -> rig filter 2398 Hz LSB → FIL2 sent`.

Opcija	Namjena
<code>--filters 3000,2400,1800</code>	Druge SSB referentne širine, redom FIL1,FIL2,FIL3; uključuje odabir filtra i za drugi Icom uređaj
<code>--civ A4</code>	CI-V adresa uređaja heksadecimalno, kad nije 94 (IC-705 A4, IC-9700 A2, IC-7610 98)
<code>--filters off</code>	Slati širine preko Hamliba, kao za druge proizvođače

Objekcije idu prije `--rigctl`, npr. `node tci-rigctld.mjs --filters 3000,2400,1800 --civ A4 --rigctl rigctl -m 3085 -r /dev/ttyACM0 -s 19200`. Isprobano samo s IC-7300. Drugi proizvođači — Yaesu, Kenwood, Elecraft i ostali — dobivaju širinu stranice preko Hamliba, koji je postavlja koliko god točno uređaj dopušta.

## Drugi uređaji

Isti koraci vrijede za svaki uređaj koji Hamlib podržava; mijenjaju se samo opcije.

- Pripremite uređaj.** U izborniku zabilježite CAT (ili CI-V) brzinu i uključite postavku poput *CAT preko USB-a* ili *CI-V transceive*, ako postoji. Na Windowsima instalirajte USB upravljački program proizvođača.
- Pronađite Hamlibov broj modela** na popisu koji ispisuje `rigctl -l`:
- Linux: `rigctl -l | grep -i 991`
- Windows: `C:\hamlib\bin\rigctl.exe -l | findstr /i 991`
- Pronađite port.** Linux: `ls /dev/serial/by-id/`. Windows: Upravitelj uređaja. Neki uređaji stvaraju dva porta — Yaesuovi FT-991A, FTDX10 i FT-710 zovu ih *Enhanced* i *Standard*; CAT je na **Enhanced** portu.
- Provjerite, zatim pokrenite most** s `-m <model> -r <port> -s <brzina>`, kao u primjeru za IC-7300: najprije `rigctl -m ... -r ... -s ... f`, zatim iste opcije iza `--rigctl`.

Neki brojevi modela, iz Hamliba 4.5 — potvrdite ih s `rigctl -l`, jer druga inačica Hamliba može drukčije numerirati uređaj:

Uređaj	-m	Uređaj	-m
Icom IC-7300	3073	Yaesu FT-991 / FT-991A	1035
Icom IC-705	3085	Yaesu FTDX10	1042
Icom IC-7610	3078	Yaesu FT-710	1049
Icom IC-9700	3081	Yaesu FT-891	1036
Xiegu G90	3088	Yaesu FT-817	1020
Xiegu X6100	3087	Kenwood TS-590SG	2037
Elecraft K3 / K3S	2029	Kenwood TS-890S	2041
Elecraft KX3	2045	Kenwood TS-2000	2014
Elecraft K4	2047	QRP Labs QCX / QDX	2052

Dodatne opcije, samo kad zatrebaju:

- Icom uređaj s promijenjenom CI-V adresom:** dodajte `-c` s adresom u *decimalnom* obliku — 94h je `-c 148`.
- Postavka koju Hamlib nudi za taj uređaj:** `rigctl -m <model> -L` ih ispisuje; postavite jednu s `-C ime=vrijednost`, npr. `-C post_write_delay=10` za sporo sučelje.

- **Uređaj koji pri otvaranju porta prelazi na odašiljanje ili se ponovno pokreće:** neki CAT kabeli koriste DTR ili RTS za PTT; dodajte `-C dtr_state=OFF -C rts_state=OFF`.

### Primjer: Yaesu FT-991A

Nije isprobano sa stvarnim FT-991A — slijedi Hamlibove postavke za taj uređaj; provjera `rigctl ... f` odmah pokazuje radi li.

Na uređaju postavite **CAT RATE** (izbornik 031) na `38400`. Hamlibov broj modela za FT-991 i FT-991A je `1035`.

USB kabel FT-991A stvara **dva** serijska porta. CAT je na **Enhanced** portu:

- **Linux:** `ls /dev/serial/by-id/` prikazuje dva retka za uređaj; onaj koji završava s `-if00-port0` je Enhanced, obično `/dev/ttyUSB0`.
- **Windows:** Upravitelj uređaja prikazuje *Silicon Labs Dual CP2105 USB to UART Bridge: Enhanced COM Port (COM5)* i *Standard COM Port*; koristite broj Enhanced porta.

#### Linux:

```
rigctl -m 1035 -r /dev/ttyUSB0 -s 38400 f
cd ~/PhantomSDR-Plus/tci-bridge
node tci-rigctld.mjs --rigctl rigctl -m 1035 -r /dev/ttyUSB0 -s 38400
```

#### Windows:

```
C:\hamlib\bin\rigctl.exe -m 1035 -r COM5 -s 38400 f
cd C:\tci-bridge
node tci-rigctld.mjs --rigctl C:\hamlib\bin\rigctl.exe -m 1035 -r COM5 -s 38400
```

Ako provjera ne ispiše ništa korisno, uobičajeni uzrok je Standard port umjesto Enhanced, ili CAT RATE različit od `-s`.

### Primjer: Kenwood TS-590SG

Nije isprobano sa stvarnim TS-590SG — slijedi Hamlibove postavke za taj uređaj; provjera `rigctl ... f` odmah pokazuje radi li.

U izborniku uređaja postavite brzinu **USB** porta na `115200` (broj izbornika je u priručniku za TS-590SG). Hamlibov broj modela je `2037`; stariji TS-590S je `2031`. Na Windowsima najprije instalirajte Kenwoodov upravljački program virtualnog COM porta za USB priključak.

USB kabel stvara **jedan** serijski port:

- **Linux:** redak uređaja u `ls /dev/serial/by-id/`, obično `/dev/ttyUSB0`.
- **Windows:** Upravitelj uređaja → Priključci (COM i LPT), na primjer `COM6`.

#### Linux:

```
rigctl -m 2037 -r /dev/ttyUSB0 -s 115200 f
cd ~/PhantomSDR-Plus/tci-bridge
node tci-rigctld.mjs --rigctl rigctl -m 2037 -r /dev/ttyUSB0 -s 115200
```

#### Windows:

```
C:\hamlib\bin\rigctl.exe -m 2037 -r COM6 -s 115200 f
cd C:\tci-bridge
node tci-rigctld.mjs --rigctl C:\hamlib\bin\rigctl.exe -m 2037 -r COM6 -s 115200
```

Ako provjera ne uspije, uobičajeni uzrok je USB brzina različita od `-s`, ili kabel u RS-232 (COM) utičnici uređaja dok `-r` navodi USB port.

### Još primjera

Nijedan nije isproban na stvarnom uređaju. Svaki redak daje opcije koje idu iza `rigctl` pri provjeri i iza `--rigctl rigctl` za most; u izborniku uređaja postavite istu brzinu. Linux port je uobičajeni — potvrdite ga s `ls /dev/serial/by-id/`. Na Windowsima zamijenite port COM brojem iz Upravitelja uređaja, a `rigctl` s `C:\hamlib\bin\rigctl.exe`.

Uređaj	Izbornik uređaja	Opcije (Linux)	Napomene
Icom IC-705 (USB)	CI-V USB Baud Rate 19200	-m 3085 -r /dev/ttyACM0 -s 19200	Pojavljaju se dva porta; CI-V je prvi (ttyACM0)
Icom IC-7100 (USB)	CI-V USB Baud Rate 19200	-m 3070 -r /dev/ttyUSB0 -s 19200	Hamlibova najveća brzina za ovaj uređaj je 19200
Icom IC-7610	CI-V USB Baud Rate 115200	-m 3078 -r /dev/ttyUSB0 -s 115200	
Icom IC-9700	CI-V USB Baud Rate 38400	-m 3081 -r /dev/ttyUSB0 -s 38400	
Yaesu FTDX101D / FTDX101MP	CAT RATE 38400	-m 1040 -r /dev/ttyUSB0 -s 38400	-m 1044 za FTDX101MP. Dva porta; koristite Enhanced
Yaesu FTDX10	CAT RATE 38400	-m 1042 -r /dev/ttyUSB0 -s 38400	Dva porta; koristite Enhanced, kao kod FT-991A
Yaesu FT-710	CAT RATE 38400	-m 1049 -r /dev/ttyUSB0 -s 38400	Dva porta; koristite Enhanced, kao kod FT-991A
Yaesu FT-891	CAT RATE 38400	-m 1036 -r /dev/ttyUSB0 -s 38400	Dva porta; koristite Enhanced, kao kod FT-991A
Yaesu FT-450D	CAT RATE 38400	-m 1046 -r /dev/ttyUSB0 -s 38400	RS-232 utičnica: koristite USB-serijski adapter
Yaesu FT-817 / FT-818	CAT RATE 38400	-m 1020 -r /dev/ttyUSB0 -s 38400	-m 1041 za FT-818. Treba CAT kabel na ACC utičnici
Yaesu FT-857 / FT-897	CAT RATE 38400	-m 1022 -r /dev/ttyUSB0 -s 38400	-m 1023 za FT-897. Treba CAT kabel
Kenwood TS-890S (USB)	USB brzina 115200	-m 2041 -r /dev/ttyUSB0 -s 115200	
Kenwood TS-990S (USB)	USB brzina 115200	-m 2039 -r /dev/ttyUSB0 -s 115200	
Kenwood TS-590SG / TS-590S (USB)	USB brzina 115200	-m 2037 -r /dev/ttyUSB0 -s 115200	Detaljan primjer gore
Kenwood TS-480	Brzina COM porta 57600	-m 2028 -r /dev/ttyUSB0 -s 57600	RS-232 utičnica: koristite USB-serijski adapter
Kenwood TS-2000	Brzina COM porta 57600	-m 2014 -r /dev/ttyUSB0 -s 57600	RS-232 utičnica; Hamlibova najveća brzina za ovaj uređaj je 57600
Elecraft K4 (USB)	RS232 brzina 115200	-m 2047 -r /dev/ttyUSB0 -s 115200	
Elecraft K3 / K3S	RS232 brzina 38400	-m 2029 -r /dev/ttyUSB0 -s 38400	K3S: USB; K3: serijski port ili KUSB kabel
Elecraft KX3	RS232 brzina 38400	-m 2045 -r /dev/ttyUSB0 -s 38400	KXUSB kabel
Elecraft KX2	RS232 brzina 38400	-m 2044 -r /dev/ttyUSB0 -s 38400	KXUSB kabel
Xiegu G90	CI-V brzina 19200	-m 3088 -r /dev/ttyUSB0 -s 19200	Hamlib koristi zadanu CI-V adresu G90

Uređaj	Izbornik uređaja	Opcije (Linux)	Napomene
Xiegu X6100	CI-V brzina 19200	-m 3087 -r /dev/ttyUSB0 -s 19200	Hamlibova najveća brzina za ovaj uređaj je 19200
Lab599 TX-500	—	-m 2050 -r /dev/ttyUSB0 -s 9600	Hamlib koristi samo 9600
ELAD FDM-DUO	—	-m 33001 -r /dev/ttyUSB0 -s 115200	
QRP Labs QDX	—	-m 2052 -r /dev/ttyACM0 -s 9600	USB serijski port; brzina nije bitna, ali se mora navesti

Potpuna Linux naredba za IC-705, kao primjer čitanja retka:

```
rigctl -m 3085 -r /dev/ttyACM0 -s 19200 f
node tci-rigctld.mjs --rigctl rigctl -m 3085 -r /dev/ttyACM0 -s 19200
```

## Uređaji kojima upravlja drugi program

Nekim uređajima već upravlja program koji može obaviti posao — ponekad bez ikakvog mosta. Nijedan od ovih načina nije isproban.

Uređaj i program	Što učiniti
<b>SunSDR</b> s ExpertSDR2/3, <b>FlexRadio</b> s AetherSDR, <b>Apache Labs ANAN</b> / <b>Hermes</b> s Thetisom	Bez mosta. Uključite <b>TCI poslužitelj</b> programa i koristite <b>TCI-CAT</b> izravno — portovi 50001 i 40001 pronalaze se automatski
<b>FlexRadio</b> sa SmartSDR (Windows)	Dodajte port u <b>SmartSDR CAT</b> . Govori Kenwood CAT, pa serijski port ondje radi kao TS-2000: -m 2014 -r COM8 -s 57600. TCP port radi s Hamlibovim FlexRadio modelom: -m 2036 -r 127.0.0.1:<port>
<b>Bilo koji uređaj kojim upravlja flrig</b>	Ostavite flrig da radi i koristite Hamlibov flrig model; uređaj ostaje dijeljen s fldigi, WSJT-X-om i programima za dnevnik: -m 4 -r 127.0.0.1:12345
<b>Pokrenuti rigctld</b> , ili program s <i>Hamlib NET rigctl</i> poslužiteljem	Pokrenite most bez --rigctl, s tom adresom: node tci-rigctld.mjs 50001 127.0.0.1:4532

Kao i svugdje, opcije idu iza `rigctl` pri provjeri i iza `--rigctl rigctl` za most. Mrežna adresa poput 127.0.0.1:12345 ne treba `-s`.

## Dijeljenje uređaja: put preko rigctld

Kad WSJT-X ili program za dnevnik mora koristiti uređaj dok most radi, pokrenite Hamlibov poslužitelj `rigctld` s istim opcijama, ostavite ga da radi i spojite oba programa na njega:

```
rigctld -m 3073 -r /dev/ttyUSB0 -s 115200
node tci-rigctld.mjs
```

Bez `--rigctl` most traži `rigctld` na 127.0.0.1:4532 i ispiše `rigctld connected`; `node tci-rigctld.mjs 50001 192.168.1.50:4532` koristi `rigctld` na drugom računalu. U WSJT-X-u odaberite uređaj *Hamlib NET rigctl* na istoj adresi. Na Windowsima je poslužitelj `C:\hamlib\bin\rigctld.exe` s istim opcijama; ako Windows odgovori *Access is denied*, koristite `--rigctl` i zatvorite drugi program dok most radi.

**Portovi.** Most nudi TCI na portu 50001; drugi port stavlja se prvi u naredbeni redak, npr. `node tci-rigctld.mjs 40001 --rigctl ...`. Ako su most i preglednik na različitim računalima, u **Host** upišite adresu računala s mostom.

## Rješavanje problema s TCI-CAT

Simptom	Vjerojatni uzrok	Što učiniti
Točka ostaje siva	TCI poslužitelj ili most ne radi; pogrešan <b>Host</b> ; odbijeno je dopuštenje preglednika za lokalnu mrežu	Uključite TCI u ExpertSDR/Thetis/AetherSDR ili pokrenite most; provjerite <b>Host</b> ; dopustite pristup lokalnoj mreži u postavkama stranice
<code>rigctl ... f</code> javlja pogrešku ili čeka	Pogrešan port ili brzina; nema dopuštenja <code>dialout</code> (Linux) ili USB upravljačkog programa (Windows); drugi program drži port	Uskladite <code>-s</code> s izbornikom uređaja; provjerite port ( <code>ls /dev/serial/by-id/</code> , Upravitelj uređaja); zatvorite druge programe za uređaj
Most ispisuje <code>rigctl stopped ... Type: rigctl --help</code>	Naredba je nepotpuna — obično nedostaje brzina iza <code>-s</code>	Ponovno upišite cijeli redak
Most ispisuje <code>The value after -s is missing</code>	Naredbeni redak skraćen je pri lijepljenju	Ponovno upišite kraj retka, npr. <code>-s 115200</code>
Filtar ne prati u jednom ili drugom smjeru	<b>CAT Sync</b> je isključen, na stranici radi dekodir, ili (IC-7300) FIL širine uređaja razlikuju se od 2700 / 2400 / 1800 Hz	Uključite CAT Sync; zaustavite dekodir; postavite FIL širine na uređaju ili navedite <code>--filters</code> sa širinama uređaja
Most ispisuje <code>rigctl stopped ... rig_open: error</code>	<code>rigctl</code> ne može otvoriti port	Kao za <code>rigctl ... f</code> gore
Most ispisuje <code>the rig is not answering</code>	Port se otvara, ali uređaj ne odgovara: pogrešna brzina ili model, CAT isključen u izborniku, ili promijenjena Icom CI-V adresa	Ponovite provjeru <code>rigctl ... f</code> ; za promijenjenu CI-V adresu dodajte <code>-c</code>
Most ispisuje <code>rigctld not reachable</code>	Pokrenut bez <code>--rigctl</code> , a <code>rigctld</code> ne radi	Dodajte <code>--rigctl ...</code> , ili najprije pokrenite <code>rigctld</code>
<code>rigctld.exe</code> javlja <code>Access is denied</code> (Windows)	Windows ga odbija pokrenuti	Koristite <code>--rigctl</code> — treba mu samo <code>rigctl.exe</code>
Spojeno, ali se prijemnik ne pomiče	<b>CAT Sync</b> je isključen	Uključite ga — utišavanje pri odašiljanju radi i bez njega
Način rada uređaja ne prati	Način bez odgovarajućeg na prijemniku, ili Hamlib upravljački program javlja neobičan naziv	Frekvencija se svejedno sinkronizira; način rada postavite na stranici
Nema utišavanja pri odašiljanju	Uređaj ili njegov Hamlib upravljački program ne javlja stanje odašiljanja	Na stranici nema što podesiti
Skala skače naprijed-natrag	Desktop PhantomSDR+ ili drugi program također sinkronizira uređaj	Koristite samo jedan upravljač odjednom

## Za operatere prijemnika

Nema se što podešavati. Upravljanje primopredajnikom koristi malo JavaScript sučelje koje svaka PhantomSDR-Plus stranica već ima; ne treba postavka poslužitelja, otvoreni port ni administratorska ovlast. Funkcije filtra i utišavanja stigle su s inačicom 4.1.0 — nakon primjene ponovno izgradite frontend (`./recompile.sh`, opcija 2); prijemnik ne treba zaustavljati. Ni prijemnici KiwiSDR, WebSDR i UberSDR ne trebaju ništa: aplikacija koristi kontrole koje njihove stranice već imaju.

Gumb **TCI-CAT** stigao je također s inačicom 4.1.0, opet samo uz ponovnu izgradnju frontenda. Ne otvara nijedan port na prijemniku: veza ide iz preglednika svakog slušatelja prema njegovom vlastitom računalu. Hamlib most, `tci-bridge/tci-rigctl.d.mjs`, slušatelji pokreću kod kuće; prijemnik ga ne koristi.

## Za programere: sučelje stranice

I Desktop PhantomSDR+ i CATsync Tool koriste ove funkcije, koje svaka PhantomSDR-Plus stranica postavlja na `window` nakon učitavanja (stranicama KiwiSDR, WebSDR i UberSDR upravlja se preko njihovih vlastitih, drugačijih kontrola):

Funkcija	Vraća / radi
<code>catsync_ready</code>	<code>true</code> čim su funkcije ispod postavljene
<code>catsync_getFrequency()</code>	Ugođena frekvencija, Hz
<code>catsync_setFrequency(hz)</code>	Ugađa na <code>hz</code>
<code>catsync_getMode()</code>	USB, LSB, CW, CW-L, AM, QUAM, FM, WBFM, RADEU, RADEL
<code>catsync_setMode(mode)</code>	Postavlja način rada; vraća propusni pojas na zadano za taj način
<code>catsync_getBandwidth()</code>	Ukupna širina propusnog pojasa, Hz
<code>catsync_setBandwidth(hz)</code>	Postavlja širinu — raste prema gore u USB-u, prema dolje u LSB-u, inače jednako na obje strane. Pozovite je <b>nakon</b> <code>catsync_setMode</code>
<code>catsync_getMute()</code>	<code>true</code> kad je utišano
<code>catsync_setMute(on)</code>	Utišava ili vraća zvuk, preko gumba za utišavanje na stranici

Posljednje četiri stigle su s inačicom 4.1.0, pa ih provjerite prije poziva:

```
if (window.catsync_ready) {
 window.catsync_setFrequency(7074000)
 window.catsync_setMode('USB')
 if (typeof window.catsync_setBandwidth === 'function') window.catsync_setBandwidth(2400)
}
```

Postavljanje frekvencije ponovno ugađa zvuk, pa setter pozovite samo kad se vrijednost stvarno promijenila — setter koji se stalno poziva s istom vrijednošću čuje se. Stariji ulazi u stilu KiwiSDR/ WebSDR (`setfreq`, `set_mode`, `freqset_complete`) i dalje postoje za alate koji ih očekuju.

## Rješavanje problema

Simptom	Vjerojatan uzrok	Što učiniti
<i>Could not open ttyUSB0</i> (Linux)	Niste u grupi <code>dialout</code> , ili drugi program drži port	<code>sudo usermod -aG dialout \$USER</code> , odjava i prijava; zatvorite WSJT-X, dnevnik veza, alate uređaja
Prikaz prijemnika kaže <i>not a receiver this app can drive</i>	Druga vrsta web prijemnika, ili se stranica još učitava	Podržani su PhantomSDR-Plus, KiwiSDR, PA3FWM WebSDR i UberSDR; spoj stranici dajte nekoliko sekundi
<i>The rig did not answer</i>	Pogrešna brzina, pogrešan tip uređaja, pogrešna CI-V adresa, uređaj isključen	Uskladite brzinu s izbornikom uređaja; provjerite CI-V adresu; pokušajte Hamlib
<i>Hamlib is not installed</i>	<code>rigctld</code> nije pronađen	Linux: <code>sudo apt install libhamlib-utils</code> . Ili upišite putanju u <b>rigctld program</b>
<i>rigctld stopped: ...</i>	Hamlib nije mogao otvoriti uređaj — slijedi njegova poruka	Obično port ili brzina; tekst nakon dvotočke Hamlibov je razlog
<i>flrig is not running at ...</i>	flrig je zatvoren, ili je njegov XML-RPC port drugačiji	Pokrenite flrig; provjerite port u njegovim postavkama
Spojeno, ali se prijemnik ne pomiče	Nije otvoren prozor stanice, ili je <b>Receiver window</b> vezan uz zatvorenu stanicu	Otvorite stanicu ili odaberite <i>The station window last in front</i>
Uređaj odašilje pri spajanju	DTR ili RTS preko sučelja uključuje odašiljač	Uklonite oznake <b>DTR on</b> i <b>RTS on</b>
Filtar ne slijedi	Prijemnik stariji od 4.1.0, ili ugrađeni Kenwood/Yaesu upravljač	Frekvencija i način rada i dalje se sinkroniziraju; za filtar na Kenwoodu/Yaesuu koristite Hamlib
Utišavanje pri odašiljanju ne radi ništa	Prijemnik bez ažuriranja, ili uređaj ne javlja stanje odašiljanja	Kao gore
<i>Lost the rig ... reconnecting</i>	Izvučen kabel, isključen uređaj ili je <code>rigctld</code> prestao raditi	Ništa — pokušava ponovno svake 3 sekunde i nastavlja kad se uređaj vrati
Obje strane stalno skaču	Dva programa istodobno upravljaju uređajem	Neka samo jedan program podešava uređaj, ili ga dijelite preko flriga

## Poznata ograničenja

- Jedan uređaj, jedan prozor prijemnika istodobno.
- Prijemnici osim PhantomSDR-Plus, KiwiSDR, PA3FWM WebSDR i UberSDR — primjerice OpenWebRX — nisu podržani.
- Ugrađeni upravljači prate objavljene protokole proizvođača i testirani su na simuliranim uređajima i stvarnom Hamlibu; za uređaj koji se ponaša drugačije, rezervno rješenje je Hamlib.
- Split, VFO B, RIT/XIT i memorijski kanali ne sinkroniziraju se — samo frekvencija aktivnog VFO-a.
- TCI-CAT na stranici prijemnika ne sinkronizira split, VFO B, RIT/XIT ni pomake transvertera, a isproban je samo s IC-7300 preko Hamlib mosta.
- Na Linuxu paketi koriste Hamlib iz distribucije; vlastiti nije uključen.

## POGLAVLJE 16

# Struktura projekta

---

PROJECT\_STRUCTURE.md

## Struktura projekta PhantomSDR-Plus

Ovaj dokument daje sveobuhvatan pregled strukture direktorija PhantomSDR-Plusa, organizacije datoteka i odnosa među komponentama.

---

### Sadržaj

---

1. Stablo direktorija
  2. Korijenski direktorij
  3. Izvorni kod ( `src/` )
  4. Sučelje ( `frontend/` )
  5. Popisi frekvencija ( `frequencylist/` )
  6. Konfiguracijske datoteke
  7. Sustav izgradnje
-

## Stablo direktorija

---

## PhantomSDR-Plus

```

├─ add-receiver.sh # dodaje još jedan prijemnik: upravljački program, instances/<ime>/,
receivers.toml – vidi docs/MULTI_RECEIVER.md
├─ admin_config.json
├─ admin_server.py
├─ autorun
│ ├─ audiotap.js
│ ├─ bandplan.js
│ ├─ decodeworker.js
│ ├─ index.js
│ ├─ manager.js
│ ├─ package.json
│ ├─ pool.js
│ ├─ probe-ft8.js
│ ├─ probe-js8.js
│ ├─ pskreporter.js
│ ├─ spotparse.js
│ ├─ wasm-shim.js
│ └─ wsprnet.js
├─ cf32_to_real.c # Fobos HF: zadržava I iz CF32 od rx_sdr, skaliran na s16 (gradi ga
setup-fobos.sh)
├─ chat_history.txt
├─ check-go.sh # staro: nadzornik lanca go.sh
├─ config-airspyhf.toml
├─ config.example.hackrf.toml
├─ config.example.rtlsdr.toml
├─ config-fobos-hf.toml
├─ config-fobos.toml
├─ config-hackrf.toml
├─ config-rspla.toml
├─ config-rtl.toml
├─ config-rx888mk2.toml
├─ config.toml
├─ connection_impl.hpp
├─ docs
│ ├─ ADMIN_PANEL_SETUP.md
│ ├─ Aether_config.md
│ ├─ CONNECTION_LIMITS.md
│ └─ de
│ ├─ ADMIN_PANEL_SETUP.md
│ ├─ Aether_config.md
│ ├─ CONNECTION_LIMITS.md
│ ├─ DECODERS.md
│ ├─ EDITING_VARIANTS.md
│ ├─ INSTALLATION.md
│ ├─ MULTI_RECEIVER.md
│ ├─ PhantomSDR-Plus-Documentation-DE.pdf
│ ├─ PROJECT_STRUCTURE.md
│ ├─ RADE_General_INSTALL_MANUAL_LINUX.md
│ ├─ RADE_README.md
│ ├─ README.md
│ ├─ RECEIVE_DIVERSITY.md
│ ├─ RIG_CONTROL.md
│ ├─ THERMAL_GUARD.md
│ └─ USER_GUIDE.md
├─ DECODERS.md
├─ EDITING_VARIANTS.md
├─ el
│ ├─ ADMIN_PANEL_SETUP.md
│ ├─ Aether_config.md
│ ├─ CONNECTION_LIMITS.md
│ ├─ DECODERS.md
│ ├─ EDITING_VARIANTS.md
│ └─ INSTALLATION.md

```

- └─ MULTI\_RECEIVER.md
- └─ PhantomSDR-Plus-Documentation-EL.pdf
- └─ PROJECT\_STRUCTURE.md
- └─ RADE\_General\_INSTALL\_MANUAL\_LINUX.md
- └─ RADE\_README.md
- └─ README.md
- └─ RECEIVE\_DIVERSITY.md
- └─ RIG\_CONTROL.md
- └─ THERMAL\_GUARD.md
- └─ USER\_GUIDE.md
- └─ es
  - └─ ADMIN\_PANEL\_SETUP.md
  - └─ Aether\_config.md
  - └─ CONNECTION\_LIMITS.md
  - └─ DECODERS.md
  - └─ EDITING\_VARIANTS.md
  - └─ INSTALLATION.md
  - └─ MULTI\_RECEIVER.md
  - └─ PhantomSDR-Plus-Documentation-ES.pdf
  - └─ PROJECT\_STRUCTURE.md
  - └─ RADE\_General\_INSTALL\_MANUAL\_LINUX.md
  - └─ RADE\_README.md
  - └─ README.md
  - └─ RECEIVE\_DIVERSITY.md
  - └─ RIG\_CONTROL.md
  - └─ THERMAL\_GUARD.md
  - └─ USER\_GUIDE.md
- └─ fr
  - └─ ADMIN\_PANEL\_SETUP.md
  - └─ Aether\_config.md
  - └─ CONNECTION\_LIMITS.md
  - └─ DECODERS.md
  - └─ EDITING\_VARIANTS.md
  - └─ INSTALLATION.md
  - └─ MULTI\_RECEIVER.md
  - └─ PhantomSDR-Plus-Documentation-FR.pdf
  - └─ PROJECT\_STRUCTURE.md
  - └─ RADE\_General\_INSTALL\_MANUAL\_LINUX.md
  - └─ RADE\_README.md
  - └─ README.md
  - └─ RECEIVE\_DIVERSITY.md
  - └─ RIG\_CONTROL.md
  - └─ THERMAL\_GUARD.md
  - └─ USER\_GUIDE.md
- └─ hr
  - └─ ADMIN\_PANEL\_SETUP.md
  - └─ Aether\_config.md
  - └─ CONNECTION\_LIMITS.md
  - └─ DECODERS.md
  - └─ EDITING\_VARIANTS.md
  - └─ INSTALLATION.md
  - └─ MULTI\_RECEIVER.md
  - └─ PhantomSDR-Plus-Documentation-HR.pdf
  - └─ PROJECT\_STRUCTURE.md
  - └─ RADE\_General\_INSTALL\_MANUAL\_LINUX.md
  - └─ RADE\_README.md
  - └─ README.md
  - └─ RECEIVE\_DIVERSITY.md
  - └─ RIG\_CONTROL.md
  - └─ THERMAL\_GUARD.md
  - └─ USER\_GUIDE.md
- └─ INSTALLATION.md
- └─ MULTI\_RECEIVER.md
- └─ PhantomSDR-Plus-Documentation-EN.pdf
- └─ PROJECT\_STRUCTURE.md

```

├── RADE_General_INSTALL_MANUAL_LINUX.md
├── RADE_README.md
├── README.md
├── RECEIVE_DIVERSITY.md
├── RIG_CONTROL.md
├── ru
│ ├── ADMIN_PANEL_SETUP.md
│ ├── Aether_config.md
│ ├── CONNECTION_LIMITS.md
│ ├── DECODERS.md
│ ├── EDITING_VARIANTS.md
│ ├── INSTALLATION.md
│ ├── MULTI_RECEIVER.md
│ ├── PhantomSDR-Plus-Documentation-RU.pdf
│ ├── PROJECT_STRUCTURE.md
│ ├── RADE_General_INSTALL_MANUAL_LINUX.md
│ ├── RADE_README.md
│ ├── README.md
│ ├── RECEIVE_DIVERSITY.md
│ ├── RIG_CONTROL.md
│ ├── THERMAL_GUARD.md
│ └── USER_GUIDE.md
├── sdr-stats
│ ├── package.json
│ ├── readme_de.md
│ ├── readme_el.md
│ ├── readme_es.md
│ ├── readme_fr.md
│ ├── readme_hr.md
│ ├── README.md
│ └── readme_ru.md
├── THERMAL_GUARD.md
├── USER_GUIDE.md
├── websdr2.png
├── websdr3.png
├── websdr.png
├── favicon.ico
├── fftw_wisdom
├── fix_local_geo.py
├── frequencylist
│ ├── 0.TXT
│ ├── A26all00.TXT
│ └── a26allx2.zip
│ # nije u repozitoriju: update-markers.sh ga raspakira iz
├── a26allx2.zip
│ ├── a26allx2.zip
│ ├── admin.txt
│ ├── antenna.txt
│ ├── broadcas.txt
│ ├── curl-output.txt
│ ├── currentUpdateFile.txt
│ ├── fmorg.txt
│ ├── generate-current-shortwave.py
│ ├── language.txt
│ ├── mymarkers.json
│ ├── README.md
│ ├── shortwavestations.json
│ ├── site.txt
│ └── update-markers.sh
├── frontend
│ ├── build-all.sh
│ ├── build-default.sh
│ ├── build-mobile.sh
│ ├── debug-title.sh
│ ├── favicon.ico
│ ├── fix-title-python.py
│ └── index.html

```

```

├─ jsconfig.json
├─ LICENSE
├─ make-redirect-stubs.sh
├─ mobile
│ └─ index.html
├─ package.json
├─ package-lock.json
├─ pnpm-lock.yaml
├─ postcss.config.cjs
├─ .prettierrc.json # Prettier pravila oblikovanja za izvor frontenda
├─ public
│ └─ analyze_users.py
│ └─ decoders
│ └─ ft8_lib.wasm
│ └─ js8_dict.bin # JS8 rječnik riječi, učitava se tek kad zatreba
│ └─ js8.wasm # JS8 dekodir, izgrađen iz jsdsp/js8_wasm
│ └─ logo.jpg
│ └─ stats.html
│ └─ users.html
│ └─ wf-message.json
├─ README.md
├─ site_information.json
├─ src
│ └─ app.css
│ └─ App.svelte
│ └─ assets
│ └─ amateurfrequencies.json
│ └─ background.jpg
│ └─ shortwavestations.json
│ └─ SSTV.png
│ └─ SSTV.svg
│ └─ svelte.png
│ └─ audio.js
│ └─ audio-stream-worklet.js
│ └─ bands-config.js
│ └─ broadcastSchedules.js
│ └─ clientVersion.js
│ └─ cwDecoder.js
│ └─ cw.worker.js
│ └─ cwWorkerProxy.js
│ └─ decoder.worker.js
│ └─ diversity.js
│ └─ diversityList.js
│ └─ eventBus.js
│ └─ events.js
│ └─ fax.js
│ └─ fax.worker.js
│ └─ faxWorkerProxy.js
│ └─ fft.js
│ └─ flacLowLatency.js # FLAC dekodir koji reproducira svaki paket čim stigne
│ └─ fsk.js
│ └─ fsk.worker.js
│ └─ fskWorkerProxy.js
│ └─ kiwiSource.js
│ └─ lib
│ └─ backend.js
│ └─ BandSelector.svelte
│ └─ catsync.js
│ └─ chatReply.js # oznaka i niti odgovora u chatu
│ └─ CheckButton.svelte
│ └─ colormaps.js
│ └─ Counter.svelte
│ └─ DiversityPanel.svelte
│ └─ fftRadix2.js
│ └─ freedv-reporter.js

```

```

├─ FreeDVReporter.svelte
├─ FrequencyInput.svelte
├─ FrequencyMarkers.svelte
├─ FtxSpectrum.svelte
├─ hammeractions.js
├─ LineThroughButton.svelte
├─ Logger.svelte
├─ MagicEyeIndicator.svelte
├─ ModeIdChip.svelte
├─ ModesSelector.svelte
├─ opusMlDecoder.js
├─ PassbandTuner.svelte
├─ Popover.svelte
├─ QrssPanel.svelte
├─ receivers.js # podaci izbornika prijemnika (/receivers.json)
├─ rnnoise.js # učitava RNNoise za smanjenje šuma umjetnom inteligencijom
├─ rx.js # kojem prijemniku stranica pripada (?rx=)
├─ SMeterAnalog.svelte
├─ SMeterDigital.svelte
├─ Spectrogram.svelte
├─ StatusIndicators.svelte
├─ storage.js
├─ sUnits.js # ljestvica S-jedinica prema frekvenciji (S9 = -93 dBm iznad
30 MHz)
├─ Tooltip.svelte
├─ VersionSelector.svelte
├─ VideoAreaSelector.svelte
├─ wrappers.js
├─ main.js
├─ mobile
├─ └─ backend.js
├─ └─ bookmarks.js
├─ └─ main.js
├─ └─ Mobile.svelte
├─ └─ tuning.js
├─ modeId.js
├─ modeId.worker.js
├─ modeIdWorkerProxy.js
├─ modePriors.js
├─ modules
├─ └─ decode.wasm
├─ └─ encode.wasm
├─ └─ ft4.js
├─ └─ ft8.js
├─ └─ js8-decoder.js
├─ └─ js8-format.js
├─ └─ js8.js
├─ └─ js8-reassembler.js
├─ └─ js8-slots.js
├─ └─ js8-tables.js
├─ └─ package.json
├─ └─ phantomsdrdsp_bg_fallback.js
├─ └─ phantomsdrdsp_bg.js
├─ └─ phantomsdrdsp_bg.wasm
├─ └─ phantomsdrdsp_bg.wasm.d.ts
├─ └─ phantomsdrdsp.d.ts
├─ └─ phantomsdrdsp.js
├─ └─ phantomsdrdsp_router.js
├─ └─ wspr.js
├─ olivia.js
├─ playoutControl.js # mjeri vezu, drži spremnik reprodukcije malim
├─ psk31.js
├─ refused.js
├─ remoteSource.js
├─ scanner.js

```

```

| | | | | siteInfo.js # site_information.json, prekriven vlastitom kopijom
prijemnika
| | | | | sstv.js
| | | | | sstv.worker.js
| | | | | sstvWorkerProxy.js
| | | | | uberSource.js
| | | | | unused
| | | | | | AudioProcessor.js
| | | | | | decoder.js
| | | | | | decoding.js
| | | | | | modules-emsripten
| | | | | | | dav1d.js
| | | | | | | dav1dnoWasm.js
| | | | | | | dav1dnoWasm.js.mem
| | | | | | | dav1d.wasm
| | | | | | | decode_ft8.js
| | | | | | | decode_ft8.wasm
| | | | | | | FoxenFlac.js
| | | | | | | jsDSP.js
| | | | | | | jsDSPnoWasm.js
| | | | | | | jsDSPnoWasm.js.mem
| | | | | | | jsDSPnoWasm.wasm
| | | | | | | jsDSP.wasm
| | | | | | | libzstd.js
| | | | | | | LiquidDSP.js
| | | | | | | opus.js
| | | | | | | opusnoWasm.js
| | | | | | | opusnoWasm.js.mem
| | | | | | | opus.wasm
| | | | | | | redsea.js
| | | | | | | redsea.wasm
| | | | | | unused.js
| | | | | | wrappers.js
| | | | | | videoRecorder.js
| | | | | | vite-env.d.ts
| | | | | | waterfall.js
| | | | | | webSdrCodec.js
| | | | | | webSdrSource.js
| | | | | stats.html
| | | | | svelte.config.js
| | | | | tailwind.config.cjs
| | | | | vite.config.js
| | | | go.sh # stari lanac pokretanja, zamijenjen s start-<radio>.sh
| | | | install_arch.sh
| | | | install_fedora.sh
| | | | install_opensuse.sh
| | | | install_rade.sh
| | | | install_rade_ubuntu22.sh
| | | | install.sh
| | | | install-stats-server.sh
| | | | instances # dodatni prijemnici, svaki u svojoj mapi (nije u gitu) – vidi docs/
MULTI_RECEIVER.md
| | | | instructions-for-airspy
| | | | instructions-for-rspla
| | | | jsdsp
| | | | | compilejs.sh
| | | | | configureredsea.sh
| | | | | extract_EXPORTED_FUNCTIONS.js
| | | | | ft8_wasm
| | | | | | build_ft8_wasm.sh
| | | | | | README.md
| | | | | | wasm_wrapper.c
| | | | | include
| | | | | | avif
| | | | | | | avif.h

```

```

├── internal.h
├── liquid
├── liquid.h
├── js8_wasm # JS8 dekodir u C-u, preveden u WebAssembly, s testovima i
referentnom provjerom – vidi njegov README
├── build_js8_wasm.sh
├── check_against_reference.sh
├── extract_js8_constants.py
├── extract_js8_varicode.py
├── js8_ab_test.mjs
├── js8_autorun_test.mjs
├── js8_chain_test.mjs
├── js8_constants.c
├── js8_constants.h
├── js8_decode.c
├── js8_decode.h
├── js8_decode_test.c
├── js8_encode.c
├── js8_encode.h
├── js8_frames_test.mjs
├── js8_osd.c
├── js8_osd.h
├── js8_reassembly_test.mjs
├── js8_roundtrip.c
├── js8_slots_test.mjs
├── js8_subtract.c
├── js8_subtract.h
├── js8_wasm_test.mjs
├── js8_wasm_wrapper.c
├── README.md
├── reference
├── ├── build_in_container.sh
├── ├── build_reference_decoder.sh
├── ├── build_reference_frames.sh
├── ├── Containerfile
├── ├── Containerfile.decoder
├── ├── ref_decode.f90
├── ├── ref_decode.txt
├── ├── ref_frames.cpp
├── └── ref_frames.txt
├── ref_gen.f90
├── run_tests.sh
├── lib
├── ├── ANR.c
├── ├── arm_funcs.h
├── ├── CMSIS_DSP
├── ├── ├── BUILDING.txt
├── └── └── LICENSE.txt
├── ├── dav1d.cpp
├── ├── NB.c
├── ├── NR_spectral.c
├── └── types.h
├── redsea.js
├── redsea.wasm
├── src
├── ├── index.js
├── ├── libzstd.js
├── ├── LiquidDSP.js
├── ├── NoiseProcessing.js
├── └── wbfmpll.cpp
├── kill.sh # staro: gasi procese poslužitelja, poziva ga go.sh
├── kiwi_install.sh # instalira KiwiSDR most u postojeće stablo
├── LICENSE
├── logproxy # rotirane kopije zapisnika ploče/posrednika/autoruna
├── logrotate

```

```

| └─ phantomsdr # logrotate konfiguracija za proxy.log + admin.log (prije
instalacije prilagodite putanje)
| └─ manage_admin.sh
| └─ markers.json
| └─ meson.build
| └─ meson_options.txt
| └─ phantom_fftw_wisdom
| └─ phantomsdr-admin.service # ogledna systemd jedinica za ploču (pokreće se pri dizanju sustava,
ponovno nakon pada)
| └─ phantomsdr-proxy.service # ogledna systemd jedinica za proxy, instalira se zajedno s
jedinicom ploče
| └─ phantomsdr-websdr-relay.service # primjer systemd unita za WebSDR diverziti relej
| └─ proxy.py
| └─ rade_helper.py
| └─ rade_loadtest.csv # izlaz rade_loadtest.py (jedan redak po koraku opterećenja)
| └─ rade_loadtest.py # RADE test opterećenja: koliko istodobnih dekodera stroj podnosi –
vidi docs/RADE_README.md
| └─ rade.sh
| └─ README.md
| └─ recompile.sh
| └─ receivers.toml.example # popis prijemnika za proxy.py – kopirati u receivers.toml
| └─ _relaunch.sh # staro: pomoćnik odgođenog ponovnog pokretanja lanca go.sh
| └─ request.hpp
| └─ setup_admin.sh
| └─ setup-airspyhf.sh # lanac upravljačkog programa Airspy HF+: libairspyhf +
SoapyAirspyHF + rx_sdr + udev
| └─ setup-cpufreq-perms.sh # daje grupi pravo pisanja u scaling_max_freq kako bi zaštita mogla
usporiti bez roota
| └─ setup-firewall.sh # neobvezna nftables zaštita – vidi docs/CONNECTION_LIMITS.md
| └─ setup-fobos.sh # lanac upravljačkog programa Fobos: libfobos + SoapyFobosSDR +
rx_sdr + cf32_to_real + udev
| └─ setup-hackrf.sh # HackRF: paket hackrf iz distribucije + udev (ništa se ne gradi)
| └─ setup-rspla.sh # lanac upravljačkog programa RSP1A: libmirisdr-5 + SoapyMiri +
rx_sdr + crna lista msi2500 + udev
| └─ setup-rtl-sdr.sh # RTL-SDR upravljački program: paket distribucije ili Blog V4 iz
izvornog koda + blokada DVB + udev
| └─ setup-rx888-udev.sh
| └─ setup-sdr-common.sh # zajedničke pomoćne funkcije triju setup skripti iznad
| └─ setup-websdr-relay.sh # instalira WebSDR diverziti relej (port, identitet, systemd)
| └─ smeter_theme.sh
| └─ src
| └─ audio.cpp
| └─ audio.h
| └─ chat.cpp
| └─ chat.h
| └─ client.cpp
| └─ client.h
| └─ compression.cpp
| └─ compression.h
| └─ crash_handler.cpp
| └─ crash_handler.h
| └─ events.cpp
| └─ events.h
| └─ fft.cpp
| └─ fft_cuda.cu
| └─ fft.h
| └─ fft_impl.cpp
| └─ fft_mkl.cpp
| └─ http.cpp
| └─ kiwi_bridge.h # most KiwiSDR protokola – vidi docs/Aether_config.md
| └─ listing
| └─ software_info.cpp
| └─ software_info.h
| └─ samplerreader.cpp
| └─ samplerreader.h

```

```

├─ signal.cpp
├─ signal.h
├─ spectrumserver.cpp
├─ spectrumserver.h
├─ utils
│ ├── audioprocessing.cpp
│ ├── audioprocessing.h
│ ├── dsp.cpp
│ └─ dsp.h
├─ utils.cpp
├─ utils.h
├─ waterfallcompression.cpp
├─ waterfallcompression.h
├─ waterfall.cpp
├─ waterfall.h
├─ websocket.cpp
└─ websocket.h
├─ start-airspyhf.sh
├─ start-all.sh # pokreće svaki prijemnik iz receivers.toml
├─ start-fobos-hf.sh
├─ start-fobos.sh
├─ start-hackrf.sh
├─ start-rsp1a.sh
├─ start-rtl.sh
├─ start-rx888mk2.sh
├─ stop-websdr.sh
├─ subprojects
│ ├── fftw3.wrap
│ ├── flac.wrap
│ ├── glaze.wrap
│ ├── libcds.wrap
│ ├── libflac.wrap
│ ├── libvolk.wrap
│ ├── ogg.wrap
│ ├── opus.wrap
│ └─ tomlplusplus-3.4.0
│ ├── CHANGELOG.md
│ ├── cmake
│ │ ├── install-rules.cmake
│ │ ├── project-is-top-level.cmake
│ │ ├── tomlplusplusConfig.cmake
│ │ ├── tomlplusplusConfig.cmake.meson.in
│ │ ├── tomlplusplusConfigVersion.cmake.meson.in
│ │ └─ variables.cmake
│ ├── CMakeLists.txt
│ ├── CODE_OF_CONDUCT.md
│ ├── CONTRIBUTING.md
│ ├── cpp.hint
│ ├── docs
│ │ ├── images
│ │ │ ├── badge-awesome.svg
│ │ │ ├── badge-C++17.svg
│ │ │ ├── badge-gitter.svg
│ │ │ ├── badge-license-MIT.svg
│ │ │ ├── badge-TOML.svg
│ │ │ ├── banner.ai
│ │ │ ├── banner.png
│ │ │ ├── banner.svg
│ │ │ ├── favicon.ico
│ │ │ ├── logo.ai
│ │ │ └─ logo.svg
│ │ ├── pages
│ │ │ └─ main_page.md
│ │ └─ poxy.toml
│ └─ examples

```

```

├─ benchmark_data.toml
├─ CMakeLists.txt
├─ error_printer.cpp
├─ error_printer.vcxproj
├─ examples.hpp
├─ example.toml
├─ merge_base.toml
├─ merge_overrides.toml
├─ meson.build
├─ parse_benchmark.cpp
├─ parse_benchmark.vcxproj
├─ simple_parser.cpp
├─ simple_parser.vcxproj
├─ toml_generator.cpp
├─ toml_generator.vcxproj
├─ toml_merger.cpp
├─ toml_merger.vcxproj
├─ toml_to_json_transcoder.cpp
├─ toml_to_json_transcoder.vcxproj
├─ include
├─ meson.build
├─ toml++
├─ impl
├─ array.hpp
├─ array.inl
├─ at_path.hpp
├─ at_path.inl
├─ date_time.hpp
├─ formatter.hpp
├─ formatter.inl
├─ forward_declarations.hpp
├─ header_end.hpp
├─ header_start.hpp
├─ json_formatter.hpp
├─ json_formatter.inl
├─ key.hpp
├─ make_node.hpp
├─ node.hpp
├─ node.inl
├─ node_view.hpp
├─ parse_error.hpp
├─ parse_result.hpp
├─ parser.hpp
├─ parser.inl
├─ path.hpp
├─ path.inl
├─ preprocessor.hpp
├─ print_to_stream.hpp
├─ print_to_stream.inl
├─ simd.hpp
├─ source_region.hpp
├─ std_except.hpp
├─ std_initializer_list.hpp
├─ std_map.hpp
├─ std_new.hpp
├─ std_optional.hpp
├─ std_string.hpp
├─ std_string.inl
├─ std_utility.hpp
├─ std_variant.hpp
├─ std_vector.hpp
├─ table.hpp
├─ table.inl
├─ toml_formatter.hpp
├─ toml_formatter.inl

```

```

├── unicode_autogenerated.hpp
├── unicode.hpp
├── unicode.inl
├── value.hpp
├── version.hpp
├── yaml_formatter.hpp
├── yaml_formatter.inl
├── toml.h
├── toml.hpp
├── LICENSE
├── meson.build
├── meson_options.txt
├── README.md
├── src
│ ├── meson.build
│ └── toml.cpp
├── tests
│ ├── at_path.cpp
│ ├── conformance_burntsushi_invalid.cpp
│ ├── conformance_burntsushi_valid.cpp
│ ├── conformance_iarna_invalid.cpp
│ ├── conformance_iarna_valid.cpp
│ ├── cpp.hint
│ ├── for_each.cpp
│ ├── formatters.cpp
│ ├── impl_toml.cpp
│ ├── leakproof.hpp
│ ├── lib_catch2.hpp
│ ├── main.cpp
│ ├── manipulating_arrays.cpp
│ ├── manipulating_parse_result.cpp
│ ├── manipulating_tables.cpp
│ ├── manipulating_values.cpp
│ ├── meson.build
│ ├── odr_test_1.cpp
│ ├── odr_test_2.cpp
│ ├── parsing_arrays.cpp
│ ├── parsing_booleans.cpp
│ ├── parsing_comments.cpp
│ ├── parsing_dates_and_times.cpp
│ ├── parsing_floats.cpp
│ ├── parsing_integers.cpp
│ ├── parsing_key_value_pairs.cpp
│ ├── parsing_spec_example.cpp
│ ├── parsing_strings.cpp
│ ├── parsing_tables.cpp
│ ├── path.cpp
│ ├── settings.hpp
│ ├── tests.cpp
│ ├── tests.hpp
│ ├── user_feedback.cpp
│ ├── using_iterators.cpp
│ ├── visit.cpp
│ └── vs
│ ├── odr_test.vcxproj
│ ├── test_debug_x64_cpplatest_noexcept_unrel.vcxproj
│ ├── test_debug_x64_cpplatest_noexcept.vcxproj
│ ├── test_debug_x64_cpplatest_unrel.vcxproj
│ ├── test_debug_x64_cpplatest.vcxproj
│ ├── test_debug_x64_noexcept_unrel.vcxproj
│ ├── test_debug_x64_noexcept.vcxproj
│ ├── test_debug_x64_unrel.vcxproj
│ ├── test_debug_x64.vcxproj
│ ├── test_debug_x86_cpplatest_noexcept_unrel.vcxproj
│ └── test_debug_x86_cpplatest_noexcept.vcxproj

```

```

├── test_debug_x86_cpplatest_unrel.vcxproj
├── test_debug_x86_cpplatest.vcxproj
├── test_debug_x86_noexcept_unrel.vcxproj
├── test_debug_x86_noexcept.vcxproj
├── test_debug_x86_unrel.vcxproj
├── test_debug_x86.vcxproj
├── test_release_x64_cpplatest_noexcept_unrel.vcxproj
├── test_release_x64_cpplatest_noexcept.vcxproj
├── test_release_x64_cpplatest_unrel.vcxproj
├── test_release_x64_cpplatest.vcxproj
├── test_release_x64_noexcept_unrel.vcxproj
├── test_release_x64_noexcept.vcxproj
├── test_release_x64_unrel.vcxproj
├── test_release_x64.vcxproj
├── test_release_x86_cpplatest_noexcept_unrel.vcxproj
├── test_release_x86_cpplatest_noexcept.vcxproj
├── test_release_x86_cpplatest_unrel.vcxproj
├── test_release_x86_cpplatest.vcxproj
├── test_release_x86_noexcept_unrel.vcxproj
├── test_release_x86_noexcept.vcxproj
├── test_release_x86_unrel.vcxproj
├── test_release_x86.vcxproj
├── windows_compat.cpp
├── toml++.code-workspace
├── toml.hpp
├── toml++.natvis
├── toml++.props
├── toml++.sln
├── toml-test
│ ├── meson.build
│ ├── README.md
│ ├── tt_decoder.cpp
│ ├── tt_decoder.vcxproj
│ ├── tt_encoder.cpp
│ ├── tt_encoder.vcxproj
│ └── tt.hpp
├── toml++.vcxproj
├── toml++.vcxproj.filters
├── tools
│ ├── ci_single_header_check.py
│ ├── clang_format.bat
│ ├── generate_conformance_tests.py
│ ├── generate_single_header.bat
│ ├── generate_single_header.py
│ ├── generate_windows_test_targets.py
│ ├── requirements.txt
│ ├── utils.py
│ └── version.py
├── vendor
│ ├── catch.hpp
│ ├── json.hpp
│ └── README.md
├── tomlplusplus.wrap
├── websocketpp.wrap
├── zlib.wrap
├── zstd.wrap
├── tci-bridge
│ └── tci-rigctld.mjs # TCI poslužitelj za Hamlib uređaje (IC-7300...), radi na računalu
slušačelja – vidi docs/RIG_CONTROL.md
├── thermal_guard.py # Zaštita od pregrijavanja procesora za administratorsku ploču (radi
i samostalno)
├── thermal-guard.service # primjer systemd jedinice za čuvara, bez administratorske ploče
├── tmpfiles
│ └── phantomsdr-logs.conf # drži admin.log + proxy.log u vlasništvu korisnika ploče (prije
instalacije prilagodite putanje)

```

```

├─ update.sh
├─ waterfall.sh
├─ websdr_relay.json.example # predložak konfiguracije (port, ograničenja, identitet stanice)
├─ websdr_relay.py # sam relej – vidi docs/RECEIVE_DIVERSITY.md
├─ websocketpp_asio_connection.hpp # Boost >= 1.87 sloj: ws_post umjesto io_service::post
├─ websocketpp_asio_endpoint.hpp # Boost >= 1.87 sloj: ws_work / ws_restart,
max_listen_connections
├─ websocketpp_asio.hpp # Boost >= 1.87 sloj za websocketpp (io_context,
executor_work_guard)
└─ xgo.sh # staro: pokreće spectrumserver, poziva ga check-go.sh

```

## Korijenski direktorij

### Konfiguracijske datoteke

Datoteka	Svrha	Kada je mijenjati
config.toml	Zadana konfiguracija	Početno postavljanje, testiranje
config-rtl.toml	Konfiguracija specifična za RTL-SDR	Pri korištenju RTL-SDR uređaja
config-rspla.toml	Konfiguracija za SDRplay RSP1A	Pri korištenju RSP1A
config-airspyhf.toml	Konfiguracija za Airspy HF+	Pri korištenju Airspyja
config-fobos-hf.toml	RigExpert Fobos SDR, HF1/HF2 izravno uzorkovanje (0-25 MHz)	Pri korištenju Fobosa na KV
config-fobos.toml	RigExpert Fobos SDR, RF grana (25-6000 MHz)	Pri korištenju Fobosa iznad 25 MHz
config-hackrf.toml	Konfiguracija za HackRF One	Pri korištenju HackRF-a
config-rx888mk2.toml	Konfiguracija za RX888 MK2	Pri korištenju RX888
config.example.hackrf.toml	Primjer za HackRF One	Pri korištenju HackRF-a

### Skripte za pokretanje/zaustavljanje i održavanje

Svaka od donjih `start-*.sh` skripti samostalan je **pokretač + watchdog + zapisivač**: zaustavlja svaku pokrenutu instancu, podiže prijamnik + `spectrumserver`, odvaja se u pozadinu, automatski ponovno pokreće lanac ako se ugasi i zapisuje u `logwebsdr.txt`. Dijeleva jednu skriptu za zaustavljanje i jedno `flock` zaključavanje (istodobno radi samo jedan prijamnik). Uredite samo blok **RECEIVER CONFIGURATION** na vrhu svake (argumenti prijarnika / konfiguracija / naziv procesa). Šest skripti za pokretanje izrađenih iz istog predloška (`start-airspyhf.sh`, `start-rtl.sh`, `start-rspla.sh`, `start-fobos.sh`, `start-fobos-hf.sh`, `start-hackrf.sh`) identične su ispod tog bloka; `start-rx888mk2.sh` je zasebna skripta. Od v5.0.0 više prijemnika može raditi jedan uz drugi: skripta pokrenuta s `INSTANCE=<ime>` daje tom prijemniku vlastitu mapu u `instances/<ime>/` te vlastiti zapisnik, zaključavanje i FIFO — vidi [Više prijemnika](#).

Skripta	Svrha
<code>install.sh</code>	Automatizirana instalacija i izgradnja
<code>add-receiver.sh</code>	Dodaje još jedan prijemnik ovom računalu: pita koji prijemnik i što pokriva, instalira njegov upravljački program, stvara <code>instances/&lt;ime&gt;/</code> i unos u <code>receivers.toml</code> te nudi da ga pokrene i ponovno pokrene proxy. Instalacijske skripte nude ga na kraju — vidi <a href="#">Više prijemnika</a>
<code>start-rtl.sh</code>	Pokretanje + watchdog poslužitelja s RTL-SDR-om ( <code>rtl_sdr</code> )
<code>start-rsp1a.sh</code>	Pokretanje + watchdog poslužitelja sa SDRplay RSP1A ( <code>rx_sdr</code> ); koristi libmirisdr-5, ili SDRplay API kad je instaliran ( <code>RX_DRIVER</code> nameće jedan)
<code>start-airspyhf.sh</code>	Pokretanje + watchdog poslužitelja s Airspy HF+ ( <code>rx_sdr</code> )
<code>start-fobos-hf.sh</code>	Pokretanje + watchdog poslužitelja s RigExpert Fobos SDR, HF1/HF2 izravno uzorkovanje ( <code>rx_sdr \ cf32_to_real</code> )
<code>start-fobos.sh</code>	Pokretanje + watchdog poslužitelja s RigExpert Fobos SDR, RF grana ( <code>rx_sdr</code> )
<code>start-hackrf.sh</code>	Pokretanje + watchdog poslužitelja s HackRF One ( <code>hackrf_transfer</code> )
<code>start-rx888mk2.sh</code>	Pokretanje + watchdog poslužitelja s RX888 MK2 ( <code>rx888_stream</code> )
<code>start-all.sh</code>	Pokreće svaki prijemnik iz <code>receivers.toml</code> njegovom vlastitom skriptom — na stanici s više prijemnika postavite ga kao skriptu za pokretanje u administratorskoj ploči; vidi <a href="#">Više prijemnika</a>
<code>stop-websdr.sh</code>	Zaustavlja poslužitelj i njegov watchdog — zajedničko za sve prijemnike
<code>stop-websdr.sh &lt;instanca&gt;</code>	Zaustavlja samo jedan prijemnik ( <code>main</code> za glavni); bez argumenta zaustavljaju se svi, kao i prije
<code>setup-rx888-udev.sh</code>	Instalira udev pravila da <code>rx888_stream</code> za RX-888 radi bez sudo
<code>setup-rsp1a.sh</code> / <code>setup-fobos.sh</code> / <code>setup-airspyhf.sh</code>	Instaliraju SoapySDR lanac upravljačkog programa za RSP1A, Fobos ili Airspy HF+ — SoapySDR iz distribucije, upravljački program i <code>rx_tools</code> izgrađeni u <code>sdr_drivers/</code> , udev pravilo — na bilo kojoj od četiri podržane distribucije. Pokreću ih opcije 3, 5 i 6 instalacije; rade i samostalno. Zajednički kod u <code>setup-sdr-common.sh</code> — vidi <a href="#">Vodič za instalaciju</a>
<code>setup-hackrf.sh</code>	Instalira paket <code>hackrf</code> iz distribucije i udev pravilo za HackRF One, na bilo kojoj od četiri podržane distribucije; pokreće ga opcija 7 instalacije
<code>setup-rtlsdr.sh</code>	Instalira RTL-SDR upravljački program na bilo kojoj od četiri podržane distribucije — paket <code>rtl-sdr</code> distribucije ili, s <code>RTL_V4=y</code> , upravljački program za RTL-SDR Blog V4 izgrađen iz izvornog koda — blokira DVB-T upravljački program i dodaje udev pravilo; pokreće ga <code>add-receiver.sh</code>
<code>setup-firewall.sh</code>	Neobvezna zaštita od poplave na razini jezgre: učitava nftables tablicu s gornjom granicom istodobnih veza i brzinom po izvorišnoj adresi na priključcima prijemnika, kočnicom za grubo pogađanje SSH lozinki i Windows dijeljenjem datoteka zatvorenim izvan privatnih raspona. Treba root, ne može vas zaključati vani (policy accept, uspostavljene veze prihvaćaju se prve), a <code>--apply</code> se sam poništi ako se ne potvrdi unutar 60 s — vidi <a href="#">Ograničenja veza</a>
<code>setup-cpufreq-perms.sh</code>	Daje grupi <code>cpufreq</code> pravo pisanja na ograničenje frekvencije procesora, kako bi throttle faza čuvara radila bez roota. Instalira <code>tmpfiles.d</code> pravilo da preživi ponovno pokretanje; <code>--revoke</code> sve poništava
<code>update.sh</code>	Ažuriranje instalacije iz objavljenog stabla, bez diranja vaše konfiguracije, oznaka, popisa frekvencija i vlastitih izmjena — vidi <a href="#">Vodič za instalaciju</a>
<code>recompile.sh</code>	Ponovna izgradnja backenda i/ili frontenda te odabir varijante koja se poslužuje na /
<code>smeter_theme.sh</code>	

Skripta	Svrha
	Postavi zadani izgled analognog S-metra (dark / amber / vintage) za sve korisnike i ponudi ponovnu izgradnju frontenda — vidi <a href="#">Uređivanje varijanti</a>
<code>waterfall.sh</code>	Mijenja zadanu minimalnu razinu slapa (dB) u <code>waterfall.js</code> + <code>App.svelte</code> — vidi <a href="#">README</a>
<code>kiwi_install.sh</code>	Instalira emulaciju KiwiSDR klijenata na stablo koje je nema: zakrpava izvorni kod pozadinskog dijela, kopira <code>src/kiwi_bridge.h</code> i dodaje dokumentirani <code>[kiwi_emulation]</code> blok u konfiguracijske datoteke u korijenu. Idempotentan je i sprema kopiju svake datoteke koje se dotakne — vidi <a href="#">Emulacija KiwiSDR klijenata</a>
<code>tci-bridge/tci-rigctld.mjs</code>	Prijemnik ga ne koristi. Mali Node.js program koji slušatelj pokreće uz vlastiti primopredajnik: čita frekvenciju, način rada i stanje odašiljanja iz Hamlibova <code>rigctld</code> i nudi ih kao TCI na portu 50001, pa gumb <b>TCI-CAT</b> na stranici može upravljati uređajem bez vlastitog TCI-ja, poput IC-7300 — vidi <a href="#">Upravljanje primopredajnikom</a>

**Stari lanac pokretanja.** `go.sh`, `xgo.sh`, `check-go.sh`, `kill.sh` i `_relaunch.sh` prethodni su naraštaj skripti za pokretanje, nadzor i zaustavljanje. Sve što su radile sada je unutar svakog `start-<radio>.sh`, i to je ono što treba koristiti. Ostaju na disku jer ih postojeće instalacije referenciraju i više se ne održavaju.

## Podatkovne datoteke

Datoteka	Svrha	Format
<code>markers.json</code>	Frekvencijske oznake i markeri	JSON
<code>chat_history.txt</code>	Poruke korisničkog razgovora	Običan tekst
<code>favicon.ico</code>	Ikona web-mjesta	ICO slika
<code>fftw_wisdom</code>	Podaci za optimizaciju FFT-a	FFTW binarno
<code>phantom_fftw_wisdom</code>	Dodatna optimizacija FFT-a	FFTW binarno

## Administratorska ploča i prijavljivanje spotova

Datoteka	Svrha
<code>admin_server.py</code>	Sama administratorska ploča. Uz stranice za upravljanje pokreće i uzorkivač stranice <b>Grafikoni</b> : pozadinska dretva svake 2 sekunde bilježi frekvenciju procesora, opterećenje, temperaturu i broj korisnika te ih drži samo u memoriji — 1 sat u punoj razlučivosti plus 24 sata prosjeka po 30 sekundi. Ništa se ne zapisuje na disk pa se povijest gubi pri ponovnom pokretanju.
<code>admin_config.json</code>	Postavke administratorske ploče (hash lozinke, portovi, opcije, pragovi toplinske zaštite)
<code>thermal_guard.py</code>	Zaštita od pregrijavanja procesora. Zaustavlja poslužitelj kada procesor postane prevruć i ponovno ga pokreće kad se ohladi, izvedeći pragove iz kritičnog praga koji objavljuje vaš vlastiti procesor umjesto iz fiksnog broja. Nikada ne pokušava prepoznati što nadzire poslužitelj — dok je pregrijan, ponavlja zaustavljanje svake 2 s, pa je watchdog, systemd jedinica ili cron zadatak koji ga oživi poništen dok se stroj ne ohladi. Samo standardna biblioteka; uvozi ga <code>admin_server.py</code> (takt daje uzorkivač Grafikona), a radi i samostalno za instalacije bez ploče. Zadano samo zapisuje, pa ni na što ne djeluje dok se ne uključi — vidi [Administratorska ploča] vidi <a href="#">priručnik Thermal Guard</a>
<code>autorun/</code>	Daemon za prijavljivanje spotova — vidi <a href="#">Autorun Spot Reporter</a>
<code>autorun.json</code>	Pojasevi/načini rada za dekodiranje, identitet i odredišta
<code>autorun-status.json</code>	Trenutačno stanje daemona — puni brojače <b>tekućeg rada</b> (pločice po dekozeru), koje Stop/Start vraća na nulu
<code>autorun-totals.json</code>	<b>Ukupno</b> poslani spotovi po pojasu i načinu rada — broj uz svaki potvrdni okvir; zapisuje ga daemon pa preživljava ponovna pokretanja

## Datoteke sustava izgradnje

Datoteka	Svrha
<code>meson.build</code>	Glavna konfiguracija izgradnje
<code>meson_options.txt</code>	Podesive opcije izgradnje
<code>.gitattributes</code>	Atributi Git repozitorija

## Izvorni kod ( `src/` )

Direktorij `src/` sadrži implementaciju pozadinskog dijela u C++-u.

### Ključne komponente

#### 1. Glavna aplikacija ( `main.cpp` )

- Raščlanjuje argumente naredbenog retka
- Učitava konfiguracijsku datoteku
- Inicijalizira komponente poslužitelja
- Pokreće petlju događaja

#### 2. Poslužitelj spektra ( `spectrumserver.cpp` )

- Koordinira sve komponente
- Upravlja korisničkim vezama
- Distribuiraju podatke spektra

- Obraduje korisničke zahtjeve

### 3. Upravljački programi za SDR (drivers/)

- Apstraktno sučelje za SDR hardver
- Čitanje i oblikovanje podataka uzoraka
- Rukovanje značajkama specifičnima za uređaj

### 4. DSP mehanizam (dsp/)

- Izračun FFT-a (ubrzan CPU-om/GPU-om)
- Demodulacija (AM, FM, SSB, CW itd.)
- Filtriranje i ponovno uzorkovanje zvuka
- AGC i smanjenje šuma

### 5. Web-poslužitelj (server/)

- Komunikacija putem WebSocketsa
- Posluživanje statičkih HTTP datoteka
- Upravljanje korisničkim sesijama
- Prijenos podataka u stvarnom vremenu

### 6. Kodiranje zvuka (audio/)

- FLAC kompresija
- Opus kompresija
- Optimizacija emitiranja

## Sučelje (frontend/)

Web-sučelje izgrađeno pomoću Sveltea i Vitea.

### 1. Glavna aplikacija (App.svelte)

- Komponenta najviše razine
- Struktura rasporeda
- Usklađivanje komponenti
- **Red gumba dekodera** — po jedan gumb za svaki dekodir na glavnoj ploči, odmah ispod izbornika načina rada; jedan pritisak pokreće dekodir i otvara njegov prozor, drugi ga zaustavlja. Zamijenio je dotadašnji red za širinu pojasa. RADEL/RADEU su u `lib/ModesSelector.svelte`, uz izbornik načina rada te u skočnim prozorima **Modes** i **Bands**.

### 2. Prikaz slapa (waterfall.js + lib/)

- Iscrtavanje spektra i slapa na platnu (canvas), palete boja i prilagodljivo automatsko podešavanje — sve u `waterfall.js` (čisti JS, nije komponenta)
- Interaktivno ugađanje putem `lib/PassbandTuner.svelte`
- Preklopi plana pojaseva i oznaka putem `lib/FrequencyMarkers.svelte`
- Audio spektrogram zasebna je komponenta: `lib/Spectrogram.svelte`

### 3. Kontrole (App.svelte + lib/)

- Unos i prikaz frekvencije — `lib/FrequencyInput.svelte`
- Odabir načina rada (AM/FM/SSB/CW) — `lib/ModesSelector.svelte`, promjena pojasa — `lib/BandSelector.svelte`

- AGC/NR/NB i ostale kontrole nalaze se u samom `App.svelte`; zasebna `Controls.svelte` ne postoji

### 3a. Skener (`scanner.js`)

- Pretraga kanala: vodi prijemnik preko raspona i zaustavlja se na prvom kanalu sa signalom. Čisti JS, ne komponenta — `App.svelte` daje VFO, način rada, plan pojaseva i poziv za ugađanje, a natrag prima stanje sučelja kroz jedan povratni poziv
- Prag je u dB iznad šuma pojasa umjesto apsolutne razine i koristi šum koji `waterfall.js` ionako prati (`snrNoiseDb`), pa mu ne treba vlastita kalibracija
- Dva načina prelaska pojasa: ugoditi i poslušati svaki kanal, ili najprije pregledati spektar i ugoditi samo ono što se ne može isključiti. Spektar smije preskočiti kanal, ali nikada stati na njemu — svako zaustavljanje dolazi iz stvarnog slušanja
- Raspon je ili pojas iz plana pojaseva ili točno ono što slap prikazuje; automatski nastavak, najdulje zadržavanje i popis isključenih kanala također su ovdje i čuvaju se u `localStorage`

### 4. Audiosustav (`audio.js`)

- Audiotok putem WebSocketsa
- Dekodiranje FLAC-a/Opusa
- Upravljanje reprodukcijom zvuka
- Distribuirani sirovi PCM (uzet prije AGC-a, smanjenja šuma i utišavanja) dekodirima načina rada
- Smanjenje šuma umjetnom inteligencijom: `lib/rnnoise.js` učitava RNNoise WebAssembly modul (`@jitsi/rnnoise-wasm`) pri prvoj uporabi; `audio.js` ga primjenjuje samo u govornim načinima, iza odvojka za dekodere, pa ga oni nikad ne čuju
- FLAC ide kroz `flacLowLatency.js`, koji dekodira svaki paket u trenutku kad stigne i stvarnu brzinu toka (na primjer 12016,3 Hz) svodi na 12000 Hz resamplerom sa 64 koeficijenta (2,7 ms kašnjenja). WASM dekodirer u `modules/` puštao je zvuk u fiksnim blokovima od 1024 uzorka i zadržavao oko 160 ms; i dalje se koristi za Opus, za izvore diversity prijema i za FLAC kad se stranica otvori s `?flac=wasm`
- Upravljanje spremnikom reprodukcije (`playoutControl.js`): `ArrivalMeter` mjeri koliko nepravilno paketi stižu u odnosu na vlastitu vremensku os toka te odstupanje između takta uzorkovanja prijamnika i zvučne kartice slušatelja; `CushionController` drži najmanju popunjenost spremnika u posljednjoj minuti blizu rezerve odabranog koraka spremnika tako što izreže ili ponovi jednu periodu usklađenu s visinom tona (`spliceOut` / `spliceIn`) u zvuku koji ide prema zvučniku. Dekodirer i snimke dobivaju netaknut tok. Vrijedi za put s planiranim spremnicima koji koristi svaki slušatelj stanice na običnom http-u; `?drain=0` ga isključuje, `?diag=1` prikazuje njegove vrijednosti

#### 4a. Dekodirani načina rada i njihovi workeri

Svaki od zahtjevnih dekodera načina rada izvodi se u vlastitom Web Workeru, pa dekodiranje nikada ne blokira reprodukciju zvuka ni slap. Slijede jedan zajednički obrazac — tri datoteke po dekodireru:

Dekoder	Mehanizam	Worker	Proxy na glavnoj dretvi
SSTV	<code>sstv.js</code>	<code>sstv.worker.js</code>	<code>sstvWorkerProxy.js</code>
HF FAX	<code>fax.js</code>	<code>fax.worker.js</code>	<code>faxWorkerProxy.js</code>
NAVTEX + FSK/RTTY + PSK31 + Olivia	<code>fsk.js</code> , <code>psk31.js</code> , <code>olivia.js</code>	<code>fsk.worker.js</code>	<code>fskWorkerProxy.js</code>
CW	<code>cwDecoder.js</code>	<code>cw.worker.js</code>	<code>cwWorkerProxy.js</code>

- **Mehanizam** je čisti DSP kod koji ništa ne zna o workerima, pa se može pokrenuti i izravno (jedinični testovi ili rezervni rad u istoj dretvi).
- **Worker** drži jednu instancu mehanizma i doslovno prosljeđuje njegove događaje.

- **Proxy** zrcali skup metoda mehanizma, pa ga `audio.js` poziva jednako kao što bi pozvao sam dekođer. Worker stvara odgođeno, pri prvom uključivanju, i prelazi na rad u istoj dretvi ako workeri nisu dostupni.

Dvije su pojedinosti ključne: PCM se **kopira** u novi međuspremnik prije prijenosa workeru (prijenos pogleda na audioakumulator odvojio bi ga i prekinuo reprodukciju), a workerova poruka `init` ponovno primjenjuje konfiguraciju na već pokrenuti mehanizam umjesto da pretpostavlja novi.

`fsk.js` iz jednog mehanizma poslužiće i NAVTEX i FSK/RTTY, uz odabir po instanci pomoću polja `role`; svaka instanca ima vlastito stanje, pa oba mogu raditi neovisno.

Uloga `fsk` dodatno ugošćuje dva dekođera koji uopće nisu FSK. Odabirom inačice `psk31` ili `olivia`, `fsk.js` predaje zvuk modulu `psk31.js` odnosno `olivia.js` umjesto vlastitom diskriminatorskom lancu, ali i dalje posuđuje njegovu konfiguraciju, worker i sustav događaja — pa `fsk.worker.js`, `fskWorkerProxy.js` i `audio.js` ne moraju znati ništa ni o jednom od ta dva načina rada, a sučelje posvuda prima iste događaje `char/status/metrics`.

- `psk31.js` — BPSK31: kompleksni osnovni pojas, prilagođeni filtar, diferencijalna detekcija i varicode, uz spektralno grubo hvatanje i fini AFC raspona približno  $\pm 25$  Hz.
- `olivia.js` — Olivia MFSK: prijenos MFSK prijarnika Pawela Jaloche iz f1digija (`pj_mfsk.h`, GPL-3, kao i ovaj projekt), uključujući Walsh/Hadamard korekciju pogrešaka i slijepo traženje sinkronizacije po fazi bloka i frekvencijskom pomaku.
- `broadcastSchedules.js` — UTC rasporedi koje FAX, NAVTEX i RTTY dekođeri nude kao pripremljene postavke, iz NOAA/NWS rasporeda pomorskog faksimila i objavljenih popisa NAVTEX postaja

#### 4b. Diverziti prijam (`diversity.js`)

- Povezuje lokalni prijamnik s drugim na drugoj lokaciji i prati onu koja trenutačno ima bolji signal. Običan JS, ne komponenta — oslanja se na jedno mjesto u `audio.js`, koje mu predaje lokalni PCM i reproducira ono što vrati
- **Odabir, ne zbrajanje.** Dvije lokacije čuju isti prijenos preko različitih ionosferskih putova, pa im valni oblici imaju nepovezanu fazu; zbrajanje zvuči češljasto filtrirano. Koherentno spajanje tražilo bi zajednički takt, koji dva prijarnika preko interneta ne dijele. Miješa se samo tijekom preklapanja od 30 ms
- Poravnanje korelira dvije **zvučne ovojnice** (logaritamska snaga na 100 Hz), nikad valne oblike — ovojnica preživljava i put i bilo koji kodek. Zaključavanje se prihvaća tek kad se druga, neovisna pretraga složi, čime se odbacuje uvjerljivo ali pogrešno kašnjenje koje bi dale dvije lokacije koje blijede u protufazi
- Udaljeni tok se najprije **zaključava po brzini** na lokalni. Dva prijarnika znače dva takta i dva lanca decimacije, pa im zvuk stiže i do 2% razmaknuto čak i kad oba prijavljuju 12 kHz — 240 uzoraka u sekundi klizanja, što nijedna korelacija ne zadržava. Omjer se mjeri iz toga koliko uzoraka svaka strana stvarno isporuči, a primjenjuje ga preuzorkivač koji nosi svoju razlomljenu fazu preko granica blokova, pa proizvoljan omjer drži neograničeno
- Odabir lokacije koristi percentilni SNR mjeren na **sadržajno poravnatim** uzorcima, uz histerezu, mjerač zadržavanja i brzi izlaz ako aktivna lokacija propadne. Razine se usklađuju šum na šum, pa promjena ne mijenja pozadinski šum
- Dekođeri zadržavaju **lokalni** tok: FT8, JS8, WSPR i RADE integriraju koherentno kroz odsječak, a promjena usred njega prekid je faze koji može stajati dekodiranja

#### 4c. Izvori diverzitija (`remoteSource.js`, `kiwiSource.js`, `uberSource.js`, `webSdrSource.js`)

- Jedan ugovor — `onPcm` / `onState` / `tune` / `canReceive` — tako da `diversity.js` nikad ne saznaje što je s druge strane. Dodavanje vrste prijarnika je jedna nova datoteka
- `remoteSource.js` — još jedan PhantomSDR-Plus preko `/audio` (cbor + FLAC), uz ponovnu upotrebu `createDecoder()` iz `lib/wrappers.js`

- `kiwiSource.js` — KiwiSDR: SND okviri, big-endian PCM, s 10-bajtnim GPS vremenskim žigom koji stereo paket umeće ispred zvuka
- `uberSource.js` — UberSDR preko vlastitog `/ws`: Opus iza 21-bajtnog zaglavlja, ugađanje preko otvorene veze. Identifikator sesije mora se prije prijaviti s `POST /connection` i mora biti UUID
- `webSdrSource.js` — WebSDR, kroz `websdr_relay.py` na ovom poslužitelju: preglednik se ne može spojiti izravno jer WebSDR provjerava zaglavlje `Origin`, a nijedna skripta ga ne smije mijenjati. Ugađanje ide kao tekstni okvir istom vezom; pokrivenost opsega daje relej
- `webSdrCodec.js` — WebSDR-ov audio format: bajtno označen tok čiji komprimirani blokovi pogone leaky-LMS prediktor s 20 odvoda. Prenesen iz WebSDR klijenta i provjeren uzorak po uzorak
- `diversityList.js` — popis spremljenih prijamnika i pravila adresa za četiri vrste, zajednički stolnoj ploči i mobilnoj stranici, tako da jedan format služi objema. Uz to i prijenos — kompaktan blob, njegov QR kod i parser koji prihvaća svaki oblik koji su dvije stranice ikad zapisale — te `browseUrl()`, koja pozvanu adresu vraća u onu koju preglednik može otvoriti
- `lib/DiversityPanel.svelte` — sučelje: adresa, vrsta izvora, SNR trim i stanje uživo, uz spremljene prijamnike — imenovane, slobodnog redoslijeda, s izvozom i uvozom u JSON datoteku, po vrsti izvora u `localStorage`. Sve se uređuje unutar ploče: `prompt()` i `confirm()` blokiraju glavnu dretvu, a upravo se preko nje zvuk gura u playback worklet. Gumb  QR crta popis kao kod za skeniranje, jer `localStorage` pripada jednom pregledniku, a mobitel počinje prazan. `mobile/Mobile.svelte` nosi istu funkciju u kartici **Div**, u vlastitom jednostavnom CSS-u te stranice. Vidi [Diverziti prijam](#)

#### 4d. Prepoznavanje načina rada (`modeId.js`, `modePriors.js`)

- Odgovara na pitanje „što slušam?“. Čita isti sirovi PCM odvod kao i dekoderi te rangira vjerojatne načine rada, kako bi operator odabrao pravi dekodir umjesto da isproba svih deset. Nikada ne dekodira: mjeri fizikalna svojstva signala i boduje ih prema tablici poznatih načina
- Zauzeta širina je povezana širina na  $-15$  dB oko vrha, namjerno ne vrijednost od 99 % snage: klikovi manipulacije ostavljaju duge repove na integralu snage i činili su da svaki uski način izgleda višestruko preširok. Brzina simbola dolazi iz **trenutne frekvencije**, a ne iz energija tonova, jer nijedno vrijeme integracije ne razlučuje istodobno pomak od 170 Hz i simbol od 100 Bd. Brzina manipulacije dolazi iz on/off ovojnice i ujedno služi kao očitavanje brzine CW-a
- `modePriors.js` dodaje jedini trag koji zvuk ne može nositi: gdje ste ugođeni. Signal 100 Bd / 170 Hz na 518 kHz je NAVTEX; isti signal na 14,070 MHz nije. Samo preteže ono što je signal već podupirao i nikada ne izmišlja kandidata
- FT8, JS8 i FT2 namjerno se prijavljuju kao jedna skupina: ne razdvajaju se samo po širini i razmaku tonova, a tvrditi suprotno bilo bi samouvjereno pogrešan odgovor
- Ispod otprilike 10 dB SNR-a šuti umjesto da nagađa
- Radi u vlastitom Web Workeru (`modeId.worker.js` + `modeIdWorkerProxy.js`), po istom obrascu engine/worker/proxy kao i dekoderi iznad; rezultat je oznaka u `lib/ModeIdChip.svelte`

#### 4e. Odbijene veze (`refused.js`, `clientVersion.js`)

- `refused.js` je zajednički rječnik za vezu koju poslužitelj odbija: kôd zatvaranja **4003** (iznad ograničenja po adresi, ili je stranica starija od `[server] min_client_version`) i **4001** (sysopovo izbacivanje). Oba su konačna. `audio.js`, `waterfall.js` i `events.js` svi uvoze `isRefusal()` odande, jer sva tri otvaraju vezu koja može biti odbijena i sva tri tada moraju razriješiti svoje inicijalizacijsko obećanje — inače stranica zauvijek čeka vezu koja nikad neće doći
- Ništa ne pokušava ponovno. Pala `/audio` veza namjerno završava sesiju: `/waterfall` i `/events` nikad se nisu vraćali s ponovnim spajanjem, pa je ponovljena sesija bila živi zvuk na zamrznutom slapu, a pred ograničenjem brzine svaki bi pokušaj produljio upravo ono odbijanje koje je htio zaobići. Stolna stranica objašnjava odbijanje koje se dogodi pri učitavanju i jednostavno stane kad se dogodi usred sesije; `/mobile` prikazuje jedan redak koji slušatelju kaže da osvježi

- `clientVersion.js` sadrži jedan jedini cijeli broj, `CLIENT_VERSION`, koji stranica dodaje svojoj zvučnoj vezi kao `/audio?v=N`. Poslužitelj odbija sve ispod `[server] min_client_version`, i tako stanica prisiljava kartice koje još rade na starijoj inačici da se osvježe — jedina poluga koja postoji, jer poslužitelj ne doseže JavaScript koji već radi u pregledniku. Povećajte ga kad promjenu u frontendu otvorene kartice više ne smiju ignorirati
- Potpuna referenca: [Ograničenja veza](#)

#### 4f. Odgovori u chatu (`lib/chatReply.js`)

- Odgovor je obična poruka u chatu koja počinje oznakom što imenuje poruku na koju odgovara, `[RE:<vremenska oznaka>|<ime>]`; poslužitelj je sprema i prosljeđuje kao svaki drugi redak i ne treba nikakvu promjenu
- `chatReply.js` stvara i čita tu oznaku te grupira odgovore ispod izvorne poruke, jednu razinu duboko. Koriste ga i `App.svelte` i `mobile/Mobile.svelte`, pa obje stranice jednako slažu chat
- Kad je izvorna poruka već ispala iz povijesti od 20 poruka na poslužitelju, odgovor ostaje ondje gdje je stigao i prikazuje samo ime i vrijeme na koje odgovara

#### 5. Upravljanje stanjem (`stores/`)

- Reaktivne pohrane podataka
- Zajedničko stanje aplikacije
- Rukovanje događajima

### Ključne značajke

- Obrada zvuka u stvarnom vremenu
- Dekodiranje digitalnih načina rada (FT8, RTTY itd.)
- Filtriranje zvuka
- Analiza spektra

### Popisi frekvencija (`frequencylist/`)

Oznake frekvencija na slapu. `mymarkers.json` je popis koji prijamnik doista prikazuje; ostalo je sirovina koju `update-markers.sh` pretvara u njega, osvježena iz mrežnih rasporeda. `README.md` u toj mapi objašnjava ažuriranje na svih sedam jezika.

```
frequencylist/
├─ mymarkers.json # oznake koje prijamnik prikazuje – ručno uređivane, preživljavaju
ažuriranje
├─ shortwavestations.json # kratkovalne postaje, generira ih generate-current-shortwave.py
├─ 0.TXT # globalni A26 HF raspored, raspakira ga i obrađuje update-markers.sh
├─ a26allx2.zip # izvorna arhiva koju preuzima update-markers.sh
├─ admin.txt · antenna.txt · broadcas.txt · fmorg.txt · language.txt · site.txt
├─ # pomoćni popisi iz istog izvora (lokacije, jezici, administratori,
antene)
├─ generate-current-shortwave.py
├─ update-markers.sh # osvježava sve navedeno iz mrežnih rasporeda
└─ README.md
```

**Format (mymarkers.json, shortwavestations.json)**

```
[
 { "frequency": 77500, "name": "DCF77", "mode": "CW" },
 { "frequency": 2485000, "name": "Vanuatu Broadcasting", "mode": "AM" }
]
```

**Konfiguracijske datoteke****Konfiguracija poslužitelja (datoteke .toml)**

Struktura konfiguracijskih datoteka:

```
[server]
Web server settings
port = 9002
html_root = "frontend/dist/"
threads = 2
otherusers = 1

[limits]
Ograničenja po adresi – sva su zadano isključena ili na razumnoj vrijednosti,
pa se postava bez njih ponaša kao i prije. Vidi CONNECTION_LIMITS.md.
per_ip = 3 # istodobni slušatelji s jedne adrese
per_ip_rate = 40 # nove veze u minuti s jedne adrese

[websdr]
Online registration
register_online = true
name = "WebSDR Name"
antenna = "Antenna Type"
grid_locator = "AB12cd"
hostname = "domain.com"

[input]
SDR input settings
sps = 2048000 # Sample rate
fft_size = 131072 # FFT size
frequency = 145000000 # Base frequency
signal = "iq" # Signal type: "iq" or "real"
audio_sps = 12000 # Audio sample rate
audio_compression = "opus" # "flac" or "opus"
accelerator = "opencl" # "none", "cuda", "opencl"

[input.driver]
Driver settings
name = "stdin"
format = "u8" # Sample format

[input.defaults]
User interface defaults
frequency = 145500000
modulation = "FM"
```

## Podaci o lokaciji ( `site_information.json` )

```
{
 "siteSysop": "Operator Callsign",
 "siteSysopEmailAddress": "email@example.com",
 "siteGridSquare": "AB12cd",
 "siteCity": "City, Country",
 "siteInformation": "https://github.com/svlbtl/PhantomSDR-Plus",
 "siteHardware": "Hardware specs",
 "siteSoftware": "Software version",
 "siteReceiver": "SDR model",
 "siteAntenna": "Antenna description",
 "siteNote": "Additional notes",
 "siteIP": "http://domain.com:9002",
 "siteSDRBaseFrequency": 0,
 "siteSDRBandwidth": 2048000,
 "siteRegion": 1,
 "siteChatEnabled": true
}
```

## Sustav izgradnje

### Konfiguracija izgradnje u Mesonu

#### `meson.build` (korijen)

Definira:

- Metapodatke projekta
- Ovisnosti
- Opcije prevoditelja
- Popise izvornih datoteka
- Ciljeve izgradnje

#### `meson_options.txt`

Dostupne opcije:

```
option('opencl', type: 'boolean', value: false, description: 'Enable OpenCL support')
option('cuda', type: 'boolean', value: false, description: 'Enable CUDA support')
option('optimization', type: 'string', value: '3', description: 'Optimization level')
```

## Ovisnosti datoteka

### Ovisnosti izgradnje pozadinskog dijela

```
spectrumserver binary depends on:
├─ C++ source files (src/**/*.*.cpp)
├─ External libraries:
│ ├── FFTW3
│ ├── WebSocket++
│ ├── FLAC
│ ├── Opus
│ ├── Liquid-DSP
│ ├── Boost
│ ├── zlib
│ ├── zstd
│ └─ OpenCL/CUDA (optional)
└─ Subproject headers:
 ├── nlohmann/json
 └─ toml11
```

### Ovisnosti izgradnje sučelja

```
frontend/dist/ depends on:
├─ Source files (frontend/src/**)
├─ npm packages (node_modules/):
│ ├── Svelte
│ ├── Vite
│ ├── @wasm-audio-decoders/opus-ml
│ └─ ...
└─ Static assets (frontend/public/)
```

## Tok podataka

### Tok rada poslužitelja

1. SDR Hardware → rtl\_sdr/hackrf\_transfer/etc.  
↓
2. Sample Stream → stdin → spectrumserver  
↓
3. spectrumserver:
  - FFT calculation (waterfall)
  - Demodulation (audio)
  - Compression (FLAC/Opus)
 ↓
4. WebSocket → Browser Client  
↓
5. Browser:
  - Render waterfall
  - Decode and play audio
  - Display controls

## Tok korisničke interakcije

1. User clicks on waterfall  
↓
2. JavaScript sends frequency change request  
↓
3. WebSocket → spectrumserver  
↓
4. spectrumserver:
  - Updates demodulator frequency
  - Sends new audio stream
 ↓
5. Browser receives and plays new audio

## Vodič za izmjenu datoteka

### Kada mijenjate kod pozadinskog dijela (`src/**`):

```
cd PhantomSDR-Plus
meson compile -C build
Server restart required
```

### Kada mijenjate kod sučelja (`frontend/src/**`):

```
cd PhantomSDR-Plus/frontend
npm run build
cd ..
Server restart required (for static files)
```

### Kada mijenjate konfiguraciju (`.toml`, `.json`):

```
Restart server
./stop-websdr.sh
./start-rtl.sh # (or appropriate start script)
```

### Kada mijenjate oznake (`markers.json`):

```
Reload page in browser
No server restart needed
```

## Važne putanje

### Putanje u izvođenju

- **Konfiguracija:** `./config-*.toml`
- **HTML korijen:** `./frontend/dist/`
- **Oznake:** `./markers.json`
- **Povijest razgovora:** `./chat_history.txt`
- **FFTW wisdom:** `./fftw_wisdom`, `./phantom_fftw_wisdom`

## Putanje izgradnje

- Izlazna binarna datoteka: `./build/spectrumserver`
- Izlaz sučelja: `./frontend/dist/`
- Node moduli: `./frontend/node_modules/`

## Putanje izvornog koda

- Izvorni kod pozadinskog dijela: `./src/`
- Izvorni kod sučelja: `./frontend/src/`
- DSP biblioteke: `./jsdsp/`

## Uobičajene radnje s datotekama

### Dodavanje nove SDR konfiguracije

1. Kopirajte postojeću konfiguraciju: `cp config-rtl.toml config-mydevice.toml`
2. Uredite parametre: `nano config-mydevice.toml`
3. Stvorite početnu skriptu: `cp start-rtl.sh start-mydevice.sh`
4. Uredite početnu skriptu: `nano start-mydevice.sh` — mijenjajte samo blok **RECEIVER CONFIGURATION** na vrhu (`RX_LABEL`, `RX_COMM` = naziv procesa prijamnika, `RX_ARGS`, `CONFIG`, `FIFO` te kuku `prestart` ako je uređaju potrebna). Logika pokretača/watchdoga/zapisivanja ispod toga je općenita i ne treba je mijenjati.
5. Postavite je kao izvršnu: `chmod +x start-mydevice.sh`

`stop-websdr.sh` već zaustavlja bilo koji `start-*.sh --watchdog`; ako vaš prijamnik koristi naziv procesa različit od `rx888_stream/rx_sdr/rtl_sdr/hackrf_transfer/cf32_to_real`, dodajte i ondje redak `killall -9 <naziv>`.

### Prilagodba sučelja

1. Izmijenite izvorni kod: `nano frontend/src/App.svelte`
2. Ponovno izgradite: `cd frontend && npm run build && cd ..`
3. Ponovno pokrenite poslužitelj: `./stop-websdr.sh && ./start-rtl.sh`

### Dodavanje vlastitih oznaka

1. Uredite datoteku oznaka: `nano markers.json`
2. Format:
 

```
json { "markers": [{ "frequency": 145500000, "label": "2m Calling", "mode": "FM" }] }
```
3. Ponovno učitajte preglednik (ponovno pokretanje poslužitelja nije potrebno)

## Upravljanje verzijama

### Datoteke koje treba pratiti u Gitu

- Izvorni kod (`src/`, `frontend/src/`, `jsdsp/`)
- Primjeri konfiguracije (`config.example.*.toml`)
- Sustav izgradnje (`meson.build`, `meson_options.txt`)

- Dokumentacija (`*.md`, `docs/`)
- Skripte (`*.sh`)

### **Datoteke koje treba zanemariti (`.gitignore`)**

- Rezultati izgradnje (`build/`, `frontend/dist/`)
- Ovisnosti (`frontend/node_modules/`)
- Korisnički podaci (`chat_history.txt`)
- Osobne konfiguracije (`config-rtl.toml` ako je prilagođena)
- Binarni podaci (`*.o`, `*.so`)

---

**Ova dokumentacija strukture trebala bi vam pomoći da se snađete u kodu PhantomSDR-Plusa i razumijete ga.**

Za upute o postavljanju pogledajte [INSTALLATION.md](#). Za informacije o korištenju pogledajte [USER\\_GUIDE.md](#).

**73 de SV1BTL & SV2AMK**