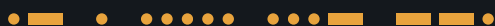


AE5VG

AMATEUR RADIO STATION



THE OPERATING SKILLS

A field guide to the operating skills bundled with **fldigi-mcp** and **n3fjp-mcp** — how an agent keys the transmitter and returns to receive at the exact instant an over ends, runs a contest QSO from CQ to logged contact, and survives everything a real band throws at it. Every rule in this guide was proven on the air.

FIELD GUIDE · VERSION 0.2 · JULY 2026

2

OPERATING
SKILLS

6

RULES IN THE
STANDARD

7

SPECIAL CASES
HANDLED

3

WORKED EXAMPLES
FROM THE AIR

CONTENTS

BEFORE YOU START

A	Skills at a glance	03
B	Installing the skills	04
C	Your first session — Claude for hams	05
D	How the system works	07
E	The operating standard	08

THE SKILLS

01	fldigi-operating	09
02	contest-operating	10

ON THE AIR

F	The special-case playbook	11
G	Worked examples from Field Day	12

REFERENCE

H	Logging to N3FJP	14
I	About & provenance	15

SKILLS AT A GLANCE

Two skills, one operating system. Each in one view — what it governs, and what you get. One full page per skill inside.

#	SKILL	SHIPS WITH	WHAT IT GOVERNS	WHAT YOU GET
01	fldigi-operating	fldigi-mcp	The radio: keying the transmitter, returning to receive the instant an over ends, and polling the RX stream so no caller is missed.	Instant TX→RX handoff + a caller never ignored
02	contest-operating	n3fjp-mcp	The contest: the QSO state machine from CQ to logged contact, the special-case playbook, and verified N3FJP logging.	Completed exchanges + a clean, duplicate-free log

WHO READS THESE SKILLS

You don't invoke these skills by name — the **agent** does, when you say things like “*call CQ on 20 meters*” or “*run the Field Day loop*.” The skills are the agent's operating procedures. This guide is for the human at the station: what the agent will do on your behalf, and how to verify it is doing it right.

INSTALLING THE SKILLS

Each skill ships inside its MCP package — install the server, and the skill travels with it. Nothing is configured separately.

In Claude Desktop `.mcpb` bundle — recommended

1 Install the bundles.

Double-click `fldigi-mcp.mcpb` and `n3fjp-mcp.mcpb` (or add them under Settings → Extensions). Each bundle carries its skill at `skills/<name>/SKILL.md` plus this guide.

2 Set the transmit gate.

In the `fldigi-mcp` extension settings, enter your callsign. With no callsign set the station is receive-only — nothing can key the radio.

In Claude Code / Cowork

1 Copy the skill directories

from each repo into `~/.claude/skills/` (or a project's `.claude/skills/`): `skills/fldigi-operating/` and `skills/contest-operating/`.

2 Connect the MCP servers

(`uvx fldigi-mcp` , `uvx n3fjp-mcp` or from the repos) so the tools the skills reference are available.

Verify `thirty seconds`

1 Check the radio link.

Ask “*what's the fldigi status?*” — mode, frequency, and T/R state should come back. If not, the `diagnostics` tool tells you whether it's a network problem before you blame `fldigi`.

2 Dry-run the handoff.

On a dummy load: “*send TEST DE <YOURCALL> and return to receive.*” `fldigi` should key, send, and drop PTT on its own — no lingering carrier.

GOOD TO KNOW

`fldigi`'s XML-RPC default port is **7362**. A different configured port (e.g. 7372) usually means a `localhost→remote` forwarder such as `mcp-host-bridge` is intended — check `FLDIGI_PORT` before concluding `fldigi` is down.

YOUR FIRST SESSION — CLAUDE FOR HAMS

You do not need to know anything about AI to run this station. If you can install fldigi and type a sentence, you can operate. This chapter is the whole learning curve.

What you're actually installing

Claude is an assistant made by Anthropic. It runs in an ordinary desktop program — the **Claude Desktop app** (Windows or Mac), which looks like a messaging window: you type at the bottom, answers appear above. The two **MCP servers** in this project are best thought of as rig-interface cables: one plugs Claude into fldigi, the other into N3FJP. You type plain English; Claude works the programs. Nothing about your fldigi or N3FJP setup changes — they run exactly as they always have.

Setup, once

1 Install the Claude Desktop app.

Download it from `claude.ai/download`, install like any program, and sign in (create the account on the same page). A paid plan is recommended — the free tier can run out of messages mid-QSO.

2 Run your shack software as usual.

Start fldigi (its XML-RPC interface is on by default). If you want contest logging, start N3FJP and enable its TCP API (Settings → Application Program Interface → checked).

3 Add the two extensions.

Double-click `fldigi-mcp.mcpb`, click Install when Claude Desktop opens, and enter your callsign when asked — that callsign is the transmit gate. Repeat with `n3fjp-mcp.mcpb`. Done — this is a one-time step.

4 Say hello.

Open a new chat and type: “What's the fldigi status?” If mode, frequency, and T/R state come back, your station is connected.

Operating is a conversation

There are no commands to memorize and no syntax to get wrong. You type what you would tell a competent operator sitting at your rig:

YOU TYPE	WHAT HAPPENS
What's the fldigi status?	Mode, frequency, T/R, and callsign gate — your pre-flight check.
Set the rig to 14.070 USB and BPSK31.	Claude QSYs the rig and sets the modem.
Call CQ Field Day until someone answers. Work them — our exchange is 2A STX — log the contact to N3FJP, then keep calling.	The full loop in this guide: CQ, listen, exchange, TU, log, repeat. Claude narrates each QSO as it happens.
stop	

Transmission stops immediately and the radio returns to receive. Works at any moment, mid-over included.

YOU ARE STILL THE CONTROL OPERATOR

Claude operates; **you** hold the license. Part 97 responsibility — supervision, station identification, staying in the band — remains yours, exactly as if a guest operator sat at your key. Stay at the station while it transmits. Two safeguards are built in: with no callsign configured the station physically cannot key up, and “stop” always works.

FIRST TIME? DUMMY LOAD.

Run your first full session into a dummy load: “*call CQ and work whoever comes back*”, watch one complete cycle — TX drop, listen window, log entry — then connect the antenna. If anything misbehaves, type “*run the fldigi diagnostics*” and read the answer; it distinguishes a network problem from an fldigi problem.

HOW THE SYSTEM WORKS

A QSO is a loop with four states. **fldigi-operating** governs the radio in every state; **contest-operating** decides what to say and when to log.

1 • CQ

Anyone out there?

CQ FD CQ FD de <MYCALL>
<MYCALL> pse k

2 • EXCHANGE

Caller decoded — send ours

<CALL> de <MYCALL>
Pse copy 2A 2A STX STX
de <MYCALL> BK

3 • TU

They QSL — confirm theirs back

QSL <CLASS> <SECT> TU
<CALL> de <MYCALL> GL
FD sk sk

4 • LOG

Write it, then QRZ

log → log_qso ... check
logged: true

- **Skills trigger on plain language.** No commands to memorize. Say “*call CQ until someone answers, then work them and log it*” and both skills engage. You can also ask for a skill by name when you want a specific lens.
- **The radio layer runs first because it changes everything above it.** An over that doesn't drop PTT on time transmits idle carrier on top of the answering station — no exchange strategy survives that. The `^r` handoff is not optional.
- **The two skills chain deliberately.** fldigi-operating delivers clean overs and an honestly-read RX stream; contest-operating consumes that stream, runs the exchange, and hands a confirmed QSO to the logger. Findings that die on a garbled callsign become an AGN request instead of a log entry.
- **The operator stays in command.** Say “stop” at any time — `transmit → abort` drops PTT immediately and discards queued text. The callsign gate means an unconfigured station cannot transmit at all.

THE HABIT THAT MATTERS

Run it on a live band, not a tour. Put fldigi on a dummy load, say “*call CQ and work whoever comes back,*” and watch one full cycle — TX drop, listen window, log entry — before you connect the antenna.

THE OPERATING STANDARD

Six rules, inherited by both skills. Every one of them exists because we broke it once — on the air, at Field Day 2026.

1 Never poll the TX buffer to decide when to stop.

fldigi holds PTT on an empty buffer by design. End every over with `^r` (via `transmit → send`) and fldigi drops to receive at the exact sample the last character finishes. Polling always lags; the lag is transmitted on top of your caller.

2 The RX buffer never contains your own echo.

Every character came from an external station. Your callsign in RX means someone is calling you — pursue it. “Dismiss as own echo” logic makes you ignore callers; we wrote it, it did, and we deleted it.

3 Never log without a confirmed callsign.

Class and section can be inferred from a repeat; a callsign cannot. An incomplete QSO is not a contact — abandon it and CQ again.

4 Trust `logged: true`, nothing else.

N3FJP's ENTERRESPONSE can report 0 for a QSO that was written (networked mode commits asynchronously). Blind retries on `records_added: 0` produced duplicate log records. Retry only when `logged` is false.

5 Budget two overs for an unreadable station.

One AGN, one listen window. Still garble? Back to CQ. A marginal signal that costs five minutes is four QSOs you didn't make.

6 Double what must survive QRM.

`2A 2A STX STX`, callsign twice on CQ, and echo the other station's exchange back in your TU so they can verify what you logged before you QRZ.

WHY THIS EXISTS

These skills were built by running one real autonomous Field Day session and keeping the after-action report — the full account is in `n3fjp-mcp/docs/LESSONS-FIELD-DAY-2026.md`, and the worked examples in chapter F retell it verbatim.

FLDIGI-OPERATING

Governs the radio: transmit, hand off to receive, listen properly.

Transmitting an over

```
transmit → send "<over text>\n" (return_to_rx=true, the default)
```

One call: clears the TX buffer, appends `^r`, keys the radio. End messages with a real newline (`\n`) — the two-character string `^n` transmits literally. If building an over incrementally with `text → add_tx`, the last fragment must end in `^r`, with nothing added after it.

OPERATION	BEHAVIOR	USE FOR
<code>^r</code> via <code>transmit → send</code>	Graceful auto-return the moment the over ends	Every normal over
<code>transmit → abort</code>	Immediate stop, discards queued text	Panic button — caller back early, wrong message, operator says stop
<code>transmit → rx</code>	Returns to RX only <i>after</i> the buffer drains	Almost never — slow, and <i>not</i> the stop button

Listening

- **Delta reads only.** `text → rx_length` for the end position, then `text → read` from your last position. Save the new position.
- **Restart detection.** `rx_length` smaller than your last position means fldigi restarted — re-baseline and continue.
- **Listen window.** After `get_trx_status` returns `"rx"`, listen ~10 seconds (1–2 polls) before the next CQ. PSK31 operators type slowly.
- **Validate callsigns** against `[A-Z0-9]{1,3}[0-9][A-Z]{1,3}` before any directed reply — QRM garbage looks call-ish.

```
loop:
  transmit → send "CQ CQ de <MYCALL> <MYCALL> pse k\n"
  poll controls → get_trx_status until "rx"
  listen ~10 s (1–2 polls of text → rx_length)
  if plausible callsign in new RX text → work the station
  else → loop
```

CONTEST-OPERATING

Governs the contest: what to say in each state, when a QSO is real, and when it goes in the log.

The exchange, tuned on the air

- **Double everything important.** `2A 2A STX STX` copies far better than `2A STX` through QSB and QRM.
- **Echo their exchange back in the TU** (`QSL 1D AZ TU W7XYZ ...`) — the other station verifies you logged them correctly before you QRZ.
- **Callsign-only callers are normal.** Some operators send just their call and wait. Send your exchange; theirs arrives in the next over. Others send everything up front — then you go straight to TU after your exchange.
- **Log while the TU is still transmitting.** The radio is busy ~15 seconds — exactly the time logging takes. When TX drops you are ready to CQ.

Logging in one call

```
log → log_qso {"call": "W7XYZ", "contest": "field_day",
               "exchange": {"class": "1D", "section": "AZ"}}
```

Exchange fields go in the `exchange` dict — they are **not** top-level parameters. Success is `logged: true` in the response; chapter G has the full protocol and its quirks.

THE RULE THAT KEEPS THE LOG CLEAN

Never retry on `records_added: 0` alone — that value can be 0 for a QSO that was written. Retry only when `logged` is false. Blind retries are where duplicate log records come from.

THE SPECIAL-CASE PLAYBOOK

All seven encountered live at Field Day 2026 — none of them hypothetical.

SITUATION	WHAT THE AGENT DOES
Callsign only, no exchange	Normal. Send your exchange; theirs arrives in the next over.
Exchange arrives with the first call	Also normal (<code>K6ABC de W7XYZ 1D AZ 1D AZ pse k</code>). Send yours; they QSL; straight to TU.
“No copy” / “repeat”	Resend once, abbreviated: <code><CALL> de <MYCALL> · 2A STX 2A STX · BK</code> . Still nothing → QRZ once, then CQ.
Garbled / partial callsign	<code>de <MYCALL> AGN AGN pse k</code> , one listen window. Still unreadable → CQ. Two overs maximum.
Two stations doubled	Interleaved callsigns and control garbage. One AGN, then CQ — usually one calls again alone.
Another station CQing on frequency	PSK's narrow passband decodes neighbors. <code>CQ ... de <OTHERCALL></code> is not a caller — ignore and continue.
Signal lost mid-QSO	Exchange never completed → do not log. An incomplete QSO is not a contact.

FIELD NOTE · N3FJP MODAL DIALOGS

Modal dialogs on the logging PC block the entire N3FJP API. If logging calls stop responding mid-contest, a dialog is open and needs human attention — nothing the agent can clear remotely.

WORKED EXAMPLES FROM FIELD DAY

Three real QSOs from the session log, 14.070 MHz BPSK31 — transcribed as decoded, garble included. All callsigns are anonymized (ARRL-style sample calls) out of respect for the stations worked; the exchanges, flow, and garble are verbatim.

EXAMPLE 1 · W7XYZ — THE TEXTBOOK QSO

LOGGED · 1D AZ

TX CQ FD CQ FD de K6ABC K6ABC pse k

RX K6ABC de W7XYZ 1D AZ; 1D AZ PSE K

Caller sent callsign and full exchange up front — no need to ask.

TX W7XYZ de K6ABC K6ABC · Pse copy 2A 2A STX STX · de K6ABC BK

RX K6ABC QSL TU 73 de W7XYZ · GL es 73 · sk

TX QSL 1D AZ TU W7XYZ de K6ABC GL FD sk sk

Logged while the TU was still transmitting: log_qso → logged: true.

✓ **LOGGED** — clean copy both ways, exchange confirmed in both directions, one pass through the state machine.

EXAMPLE 2 · N4DEF — CALLSIGN FIRST, EXCHANGE AFTER

LOGGED · 1D NFL

TX CQ FD CQ FD de K6ABC K6ABC pse k

RX **4* K6ABr DE N4DEF N4DEet N4DEF

Callsign only — repeated three times through light garble, readable. No exchange yet: this is normal, not an error.

TX N4DEF de K6ABC K6ABC · Pse copy 2A 2A STX STX · de K6ABC BK

RX **** K6ABC · QSL PSE COPY 1D 1D NFL DE N4DEe K

Their QSL and exchange arrive together in the second over.

TX QSL 1D NFL TU N4DEF de K6ABC GL FD sk sk

✓ **LOGGED** — the state machine tolerates the exchange arriving one over late; the callsign was confirmed before anything was logged.

EXAMPLE 3 · KØGHI — NO COPY, ABANDONED CORRECTLY

NOT LOGGED

TX CQ FD CQ FD de K6ABC K6ABC pse k
RX 6ABC de KØGHI KØGH · pse cpy 1E UT 1E UT
TX KØGHI de K6ABC K6ABC · Pse copy 2A 2A STX STX · de K6ABC BK
RX nt z repeat no cpy
They didn't copy us — resend once, abbreviated (playbook, row 3).
TX KØGHI de K6ABC · 2A STX 2A STX · BK
RX cptt t te e t eg
TX KØGHI de K6ABC AGN pse k
RX (silence — two listen windows)
TX CQ FD CQ FD de K6ABC K6ABC pse k

✗ **NOT LOGGED** — we never confirmed they copied our exchange. Rule 3 of the standard: an incomplete QSO is not a contact. Two overs spent, then back to work.

LOGGING TO N3FJP

The verified protocol, and every quirk we hit on the air.

Preferred: one call

```
log → log_qso {"call": "W7XYZ", "contest": "field_day",
               "exchange": {"class": "1D", "section": "AZ"}}
```

Manual alternative

```
log → set_many {"CALL": "W7XYZ", "CLASS": "1D", "SECTION": "AZ"}
log → enter
```

SUCCESS IS LOGGED: TRUE — TRUST NOTHING ELSE

Both paths return a `logged` boolean derived from the QSO-count delta and/or an ENTEREVENT push. `records_added` can report `0` for a QSO that was written — N3FJP networked (master-table) mode commits asynchronously. Retry only when `logged` is false.

Quirks, observed live

- `set_many → calltab` can trigger an ENTEREVENT directly (the band/mode/dupe check fires on callsign entry). If it does, the QSO is logged — do not send another `enter`.
- `calltab` also populates COUNTRY/STATE/section lookups — useful for checking that the section you copied is plausible for the callsign.
- A `CALLTABDUPEEVENT` means the call is already in the log for this band/mode — usually a sign you (or a retry) already logged it.
- Exchange fields belong in the `exchange` dict of `log_qso` — passing them as top-level parameters silently logs nothing.

ABOUT & PROVENANCE

These skills were not designed on paper. They are the distillate of one real autonomous contest session.

At ARRL Field Day 2026, a class-2A club station ran 20-meter BPSK31 with an agent operating fldigi and N3FJP through `fldigi-mcp` and `n3fjp-mcp` — calling CQ, working callers, and logging contacts, with the human operator supervising and able to stop transmission at any moment. Every rule, timing, and special case in this guide comes from that session's after-action report, `n3fjp-mcp/docs/LESSONS-FIELD-DAY-2026.md`.

RESOURCE	WHERE
fldigi-mcp	github.com/sbrunner-atx/fldigi-mcp · PyPI: <code>fldigi-mcp</code>
n3fjp-mcp	github.com/sbrunner-atx/n3fjp-mcp · PyPI: <code>n3fjp-mcp</code>
Skill sources	<code><repo>/skills/<name>/SKILL.md</code>
This guide's sources	<code><repo>/docs/brand/</code> (HTML + CSS, WeasyPrint)
After-action report	<code>n3fjp-mcp/docs/LESSONS-FIELD-DAY-2026.md</code>

Corrections and additions are welcome — open an issue on either repository.

73

This is private, personal work by Stefan Brunner, AE5VG — built for amateur radio operators. Not affiliated with fldigi, N3FJP, or any employer. MIT licensed, like the servers it documents. GL es 73 de AE5VG sk.