egzumer /
uv-k5-firmware-custom

Code Issues 41 Pull requests 5 Discussions Actions Wiki

uv-k5-firmware-custom / **README.md**

MatiasFernandez and **egzumer** Fix typo in README.md ✓ 3 weeks ago

```
247 lines (201 loc) · 11.8 KB
```

uv-k5-firmware-custom / **README.md** ↑ Top

Preview Code Blame Raw

Open re-implementation of the Quansheng UV-K5/K6/5R v2.1.27 firmware

This repository is a merge of OneOfEleven custom firmware with fagci spectrum analizer plus my few changes.
All is a cloned and customized version of DualTachyon's open firmware found here ... a cool achievement !

> Tip
>
> There is a work done by others on forks of this repository. I encourage you to take a look at those too. SEE HERE

> Warning
>
> Use this firmware at your own risk (entirely). There is absolutely no guarantee that it will work in any way shape or form on your radio(s), it may even brick your radio(s), in which case, you'd need to buy another radio. Anyway, have fun.

Table of Contents

Main features:

- many of OneOfEleven mods:
  - AM fix, huge improvement in reception quality
  - long press buttons functions replicating F+ action
  - fast scanning
  - channel name editing in the menu
  - channel name + frequency display option
  - shortcut for scan-list assignment (long press `5 NOAA`)
  - scan-list toggle (long press `* Scan` while scanning)
  - configurable button function selectable from menu
  - battery percentage/voltage on status bar, selectable from menu
  - longer backlight times
  - mic bar
  - RSSI s-meter
  - more frequency steps
  - squelch more sensitive
- fagci spectrum analyzer (**F+5** to turn on)
- some other mods introduced by me:
  - SSB demodulation (adopted from fagci)
  - backlight dimming
  - battery voltage calibration from menu
  - better battery percentage calculation, selectable for 1600mAh or 2200mAh
  - more configurable button functions
  - long press MENU as another configurable button
  - better DCS/CTCSS scanning in the menu (`* SCAN` while in RX DCS/

CTCSS menu item)

  - Piotr022 style s-meter
  - restore initial freq/channel when scanning stopped with EXIT, remember last found transmission with MENU button
  - reordered and renamed menu entries
  - LCD interference crash fix
  - many others...

Manual

Up to date manual is available in the Wiki section

Radio performance

Please note that the Quansheng UV-Kx radios are not professional quality transceivers, their performance is strictly limited. The RX front end has no track-tuned band pass filtering at all, and so are wide band/wide open to any and all signals over a large frequency range.

Using the radio in high intensity RF environments will most likely make reception anything but easy (AM mode will suffer far more than FM ever will), the receiver simply doesn't have a great dynamic range, which results in distorted AM audio with stronger RX'ed signals. There is nothing more anyone can do in firmware/software to improve that, once the RX gain adjustment I do (AM fix) reaches the hardwares limit, your AM RX audio will be all but non-existent (just like Quansheng's firmware). On the other hand, FM RX audio will/should be fine.

But, they are nice toys for the price, fun to play with.

User customization

You can customize the firmware by enabling/disabling various compile options, this allows us to remove certain firmware features in order to make room in the flash for others. You'll find the options at the top of "Makefile" ('0' = disable, '1' = enable) ..

Build option	Description
🧰 **STOCK QUANSHENG FEATURES**	
ENABLE_UART	without this you can't configure radio via PC !

Build option	Description
ENABLE_AIRCOPY	easier to just enter frequency with butts
ENABLE_FMRADIO	WBFM VHF broadcast band receiver
ENABLE_NOAA	everything NOAA (only of any use in the USA)
ENABLE_VOICE	want to hear voices ?
ENABLE_VOX	
ENABLE_ALARM	TX alarms
ENABLE_TX1750	side key 1750Hz TX tone (older style repeater access)
ENABLE_PWRON_PASSWORD	power-on password stuff
ENABLE_DTMF_CALLING	DTMF calling fuctionality, sending calls, receiving calls, group calls, contacts list etc.
ENABLE_FLASHLIGHT	enable top flashlight LED (on, blink, SOS)
🧰 **CUSTOM MODS**	
ENABLE_BIG_FREQ	big font frequencies (like original QS firmware)
ENABLE_SMALL_BOLD	bold channel name/no. (when name + freq channel display mode)
ENABLE_CUSTOM_MENU_LAYOUT	changes how the menu looks like
ENABLE_KEEP_MEM_NAME	maintain channel name when (re)saving memory channel
ENABLE_WIDE_RX	full 18MHz to 1300MHz RX (though front-end/PA not designed for full range)
ENABLE_TX_WHEN_AM	allow TX (always FM) when RX is set to AM

Build option	Description
ENABLE_F_CAL_MENU	enable the radios hidden frequency calibration menu
ENABLE_CTCSS_TAIL_PHASE_SHIFT	standard CTCSS tail phase shift rather than QS's own 55Hz tone method
ENABLE_BOOT_BEEPS	gives user audio feedback on volume knob position at boot-up
ENABLE_SHOW_CHARGE_LEVEL	show the charge level when the radio is on charge
ENABLE_REVERSE_BAT_SYMBOL	mirror the battery symbol on the status bar (+ pole on the right)
ENABLE_NO_CODE_SCAN_TIMEOUT	disable 32-sec CTCSS/DCS scan timeout (press exit butt instead of time-out to end scan)
ENABLE_AM_FIX	dynamically adjust the front end gains when in AM mode to help prevent AM demodulator saturation, ignore the on-screen RSSI level (for now)
ENABLE_AM_FIX_SHOW_DATA	show debug data for the AM fix
ENABLE_SQUELCH_MORE_SENSITIVE	make squelch levels a little bit more sensitive - I plan to let user adjust the values themselves
ENABLE_FASTER_CHANNEL_SCAN	increases the channel scan speed, but the squelch is also made more twitchy
ENABLE_RSSI_BAR	enable a dBm/Sn RSSI bar graph level in place of the little antenna symbols
ENABLE_AUDIO_BAR	experimental, display an audio bar level when TX'ing
ENABLE_COPY_CHAN_TO_VFO	copy current channel settings into frequency mode. Long

Build option	Description
	press `1 BAND` when in channel mode
ENABLE_SPECTRUM	fagci spectrum analyzer, activated with `F + 5 NOAA`
ENABLE_REDUCE_LOW_MID_TX_POWER	makes medium and low power settings even lower
ENABLE_BYP_RAW_DEMODULATORS	additional BYP (bypass?) and RAW demodulation options, proved not to be very useful, but it is there if you want to experiment
ENABLE_BLMIN_TMP_OFF	additional function for configurable buttons that toggles `BLMin` on and off wihout saving it to the EEPROM
ENABLE_SCAN_RANGES	scan range mode for frequency scanning, see wiki for instructions (radio operation -> frequency scanning)
🧰 **DEBUGGING**	
ENABLE_AM_FIX_SHOW_DATA	displays settings used by AM-fix when AM transmission is received
ENABLE_AGC_SHOW_DATA	displays AGC settings
ENABLE_UART_RW_BK_REGS	adds 2 extra commands that allow to read and write BK4819 registers
🧰 **COMPILER/LINKER OPTIONS**	
ENABLE_CLANG	**experimental, builds with clang instead of gcc (LTO will be disabled if you enable this)
ENABLE_SWD	only needed if using CPU's SWD port (debugging/ programming)

Build option	Description
ENABLE_OVERLAY	cpu FLASH stuff, not needed
ENABLE_LTO	reduces size of compiled firmware but might break EEPROM reads (OVERLAY will be disabled if you enable this)

Compiler

arm-none-eabi GCC version 10.3.1 is recommended, which is the current version on Ubuntu 22.04.03 LTS. Other versions may generate a flash file that is too big. You can get an appropriate version from: https://developer.arm.com/downloads/-/gnu-rm

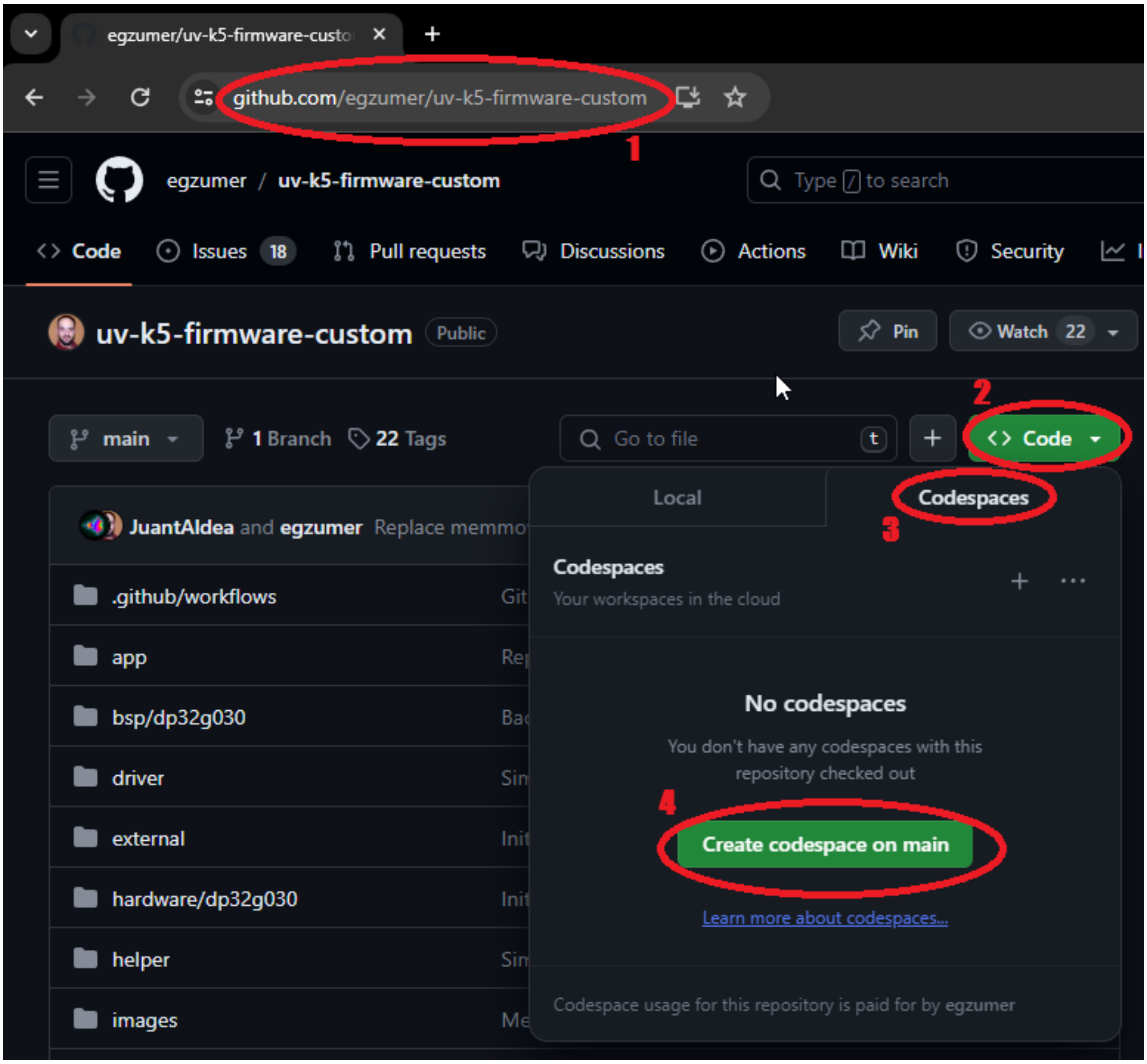
clang may be used but isn't fully supported. Resulting binaries may also be bigger. You can get it from: https://releases.llvm.org/download.html

Building

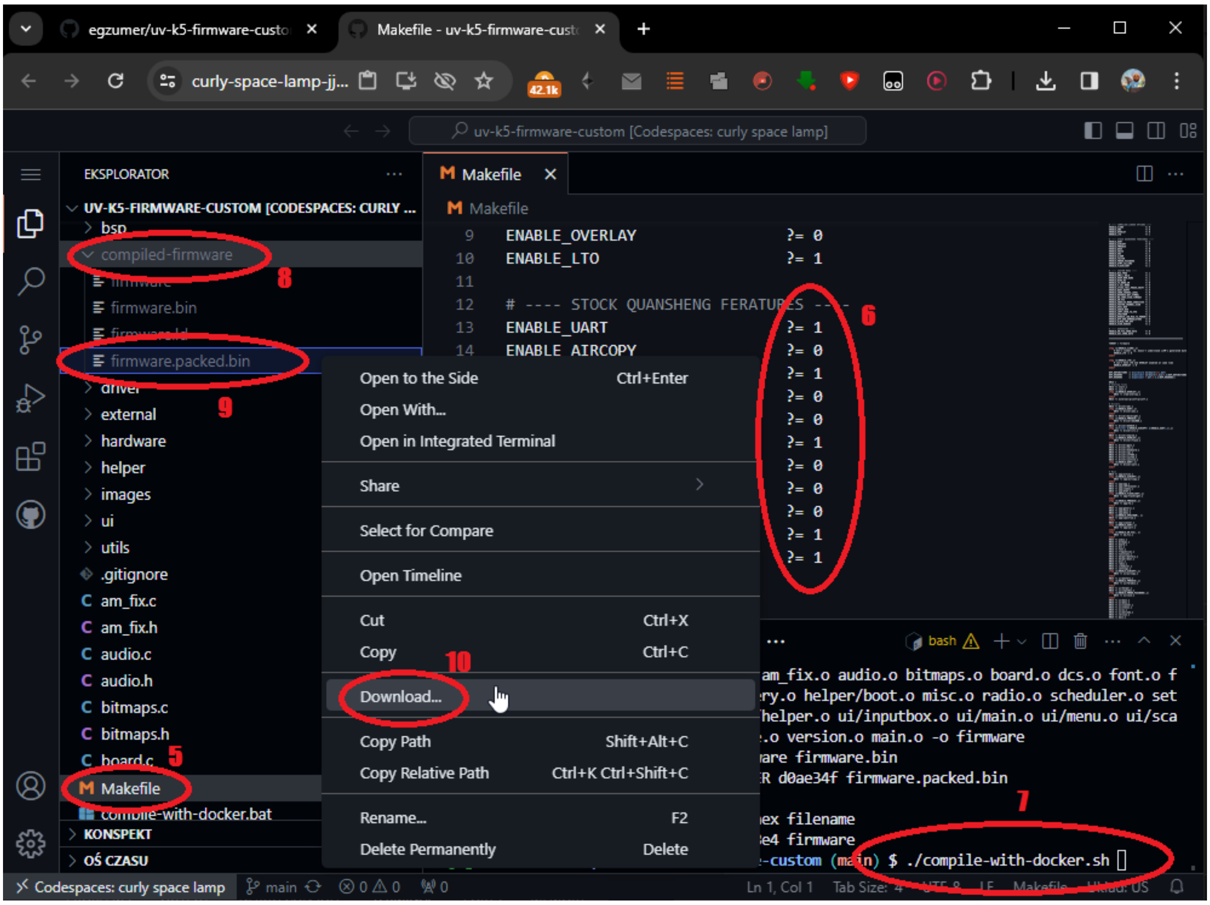
Github Codespace build method

This is the least demanding option as you don't have to install enything on your computer. All you need is Github account.

1. Go to https://github.com/egzumer/uv-k5-firmware-custom
2. Click green `Code` button
3. Change tab from `Local` to `Codespace`
4. Click green `Create codespace on main` button


egzumer/uv-k5-firmware-custo
github.com/egzumer/uv-k5-firmware-custom
1
egzumer / uv-k5-firmware-custom
Type / to search
Code
Issues 18
Pull requests
Discussions
Actions
Wiki
Security
uv-k5-firmware-custom
Public
Pin
Watch 22
2
main
1 Branch
22 Tags
Go to file
Code
Local
Codespaces
3
JuantAldea and egzumer Replace memmo
.github/workflows
app
bsp/dp32g030
driver
external
hardware/dp32g030
helper
images
Codespaces
Your workspaces in the cloud
No codespaces
You don't have any codespaces with this repository checked out
4
Create codespace on main
Learn more about codespaces...
Codespace usage for this repository is paid for by egzumer

5. Open `Makefile`
6. Edit build options, save `Makefile` changes
7. Run `./compile-with-docker.sh` in terminal window
8. Open folder `compiled-firmware`
9. Right click `firmware.packed.bin`
10. Click `Download`, now you should have a firmware on your computer that you can proceed to flash on your radio. You can use online flasher



```
ENABLE_OVERLAY                  ?= 0
ENABLE_LTO                      ?= 1

# ---- STOCK QUANSHENG FERATURES ----
ENABLE_UART                     ?= 1
ENABLE_AIRCOPY                  ?= 0
                                ?= 1
                                ?= 0
                                ?= 0
                                ?= 1
                                ?= 0
                                ?= 0
                                ?= 0
                                ?= 1
                                ?= 1
```

```
am_fix.o audio.o bitmaps.o board.o dcs.o font.o f
ry.o helper/boot.o misc.o radio.o scheduler.o set
helper.o ui/inputbox.o ui/main.o ui/menu.o ui/sca
.o version.o main.o -o firmware
are firmware.bin
R d0ae34f firmware.packed.bin

ex filename
e4 firmware
-custom (main) $ ./compile-with-docker.sh
```

Docker build method

If you have docker installed you can use compile-with-docker.bat (Windows) or compile-with-docker.sh (Linux/Mac), the output files are created in `compiled-firmware` folder. This method gives significantly smaller binaries, I've seen differences up to 1kb, so it can fit more functionalities this way. The challenge can be (or not) installing docker itself.

Windows environment build method

1. Open windows command line and run:

```
winget install -e -h git.git Python.Python.3.8 GnuWin32.Make
winget install -e -h Arm.GnuArmEmbeddedToolchain -v "10
2021.10"
```

2. Close command line, open a new one and run:

```
pip install --user --upgrade pip
pip install crcmod
mkdir c:\projects & cd /D c:/projects
git clone https://github.com/egzumer/uv-k5-firmware-
custom.git
```

3. From now on you can build the firmware by going to `c:\projects\uv-k5-firmware-custom` and running `win_make.bat` or by running a command line:

   ```
   cd /D c:\projects\uv-k5-firmware-custom
   win_make.bat
   ```

4. To reset the repository and pull new changes run (!!! it will delete all your changes !!!):

   ```
   cd /D c:\projects\uv-k5-firmware-custom
   git reset --hard & git clean -fd & git pull
   ```

I've left some notes in the win_make.bat file to maybe help with stuff.

Credits

Many thanks to various people on Telegram for putting up with me during this effort and helping:

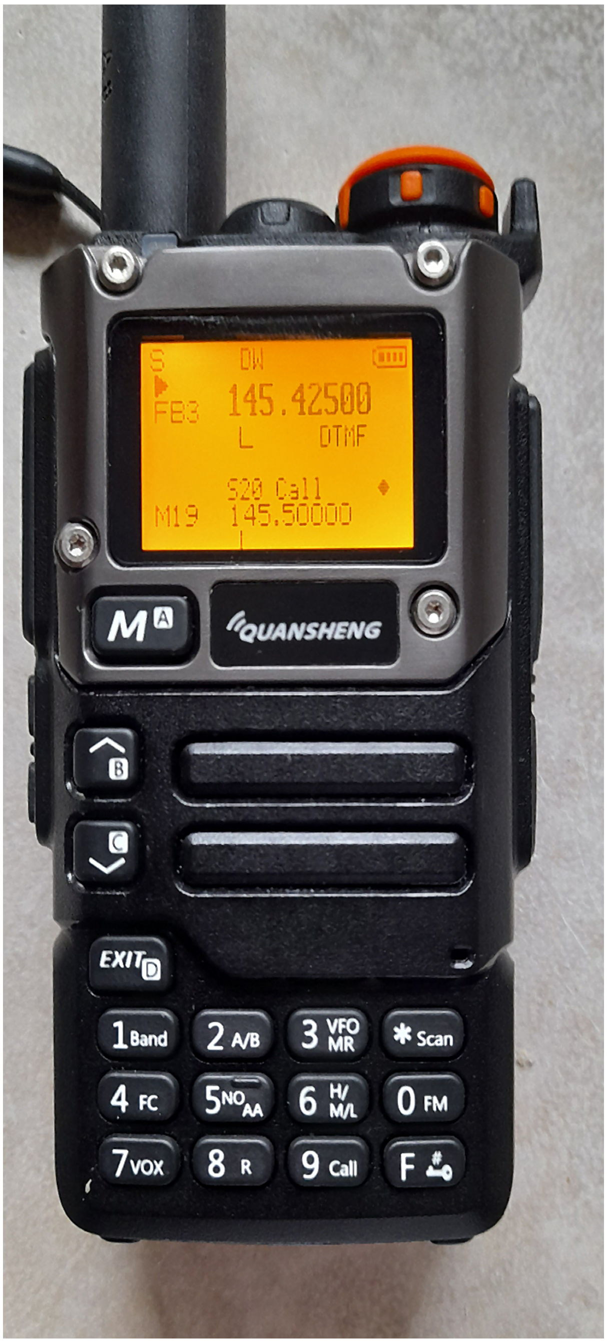
- OneOfEleven
- DualTachyon
- Mikhail
- Andrej
- Manuel
- @wagner
- @Lohtse Shar
- @Matoz
- @Davide
- @Ismo OH2FTG
- OneOfEleven
- @d1ced95
- and others I forget

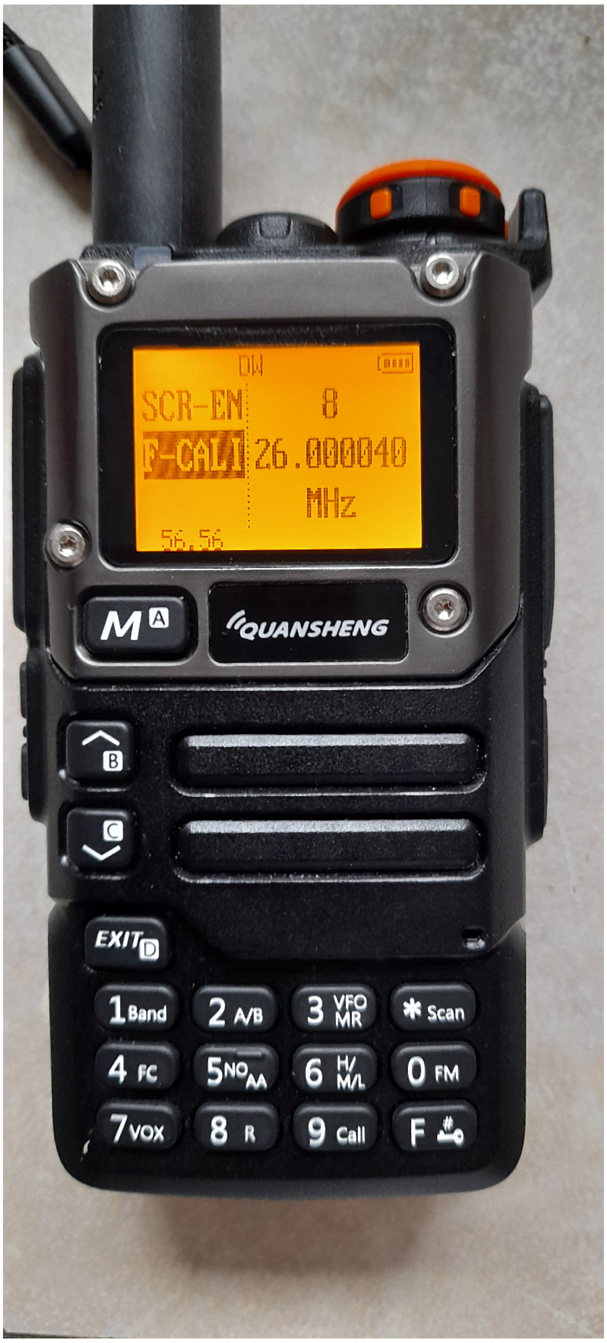
Other sources of information

ludwich66 - Quansheng UV-K5 Wiki
amnemonic - tools and sources of information

License

Example changes/updates


S DW
FB3 145.42500
L DTMF
S20 Call
M19 145.50000
L
M A
QUANSHENG
B
C
EXIT D
1 Band
2 A/B
3 VFO MR
✱ Scan
4 FC
5 NO AA
6 H/ M/L
0 FM
7 VOX
8 R
9 Call
F #


DW
SCR-EN 8
F-CAL/1 26.000040
MHz
56.56
QUANSHENG

Video showing the AM fix working ..

AM_fix.mp4

VFO2 133.60000
QUANSHENG
EXIT
1 Band
2 A/B
3 VFO MR
Scan
4 FC
5 NOAA
6 H M L
0 FM
7 VOX
8 R
9 Call
F

○ 0:00 / 1:45 ○