Wireless Cat connection for TS-50s

After played a while with ESP12e WiFi micro-controller I got idea to use it for connecting TS-50s cat to computer. TS-50s does not have proper connector, just TTL- level at digital unit pcb inside the rig.
There are several way to connect TS-50s to PC. At the time RS-232 com ports were used connecting was made with Maxim 232 chip, or similar.
Now, at USB bus time, cheap Chinese usb2ttl converter can be used.
Both of them have one disadvantage. They do not have galvanic isolation between PC and rig.

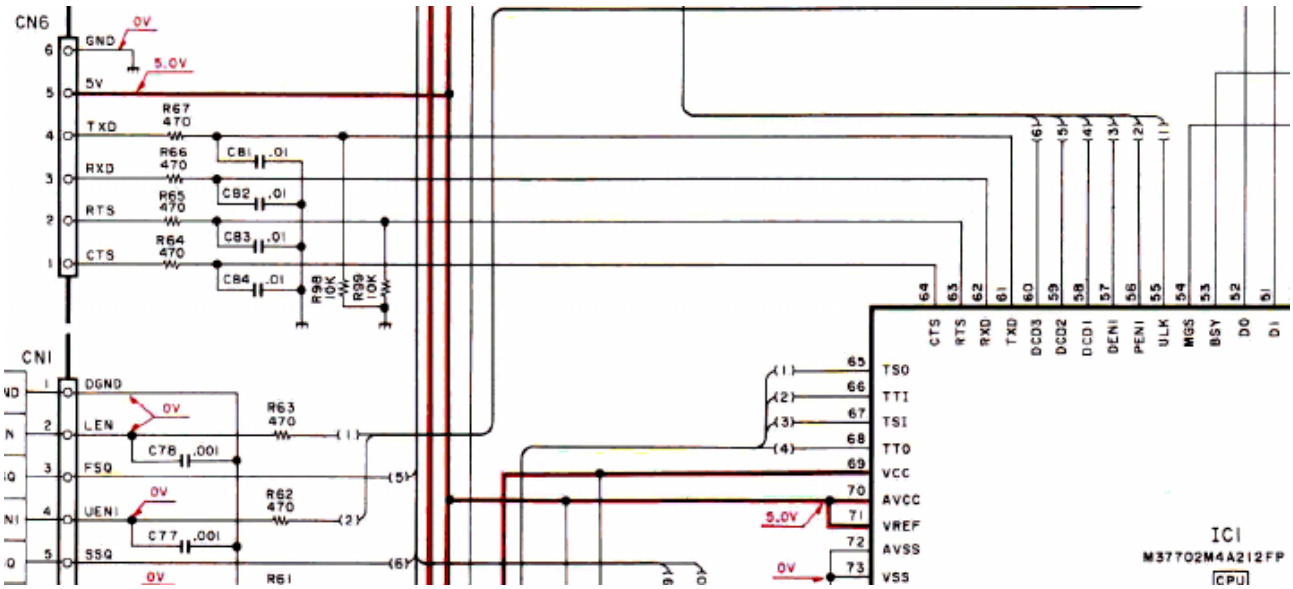
By using WiFi network isolation is guaranteed.

TS-50s socket CN6 does carry also 5V supply that can feed ESP12e. How ever there is one problem: Rig uses 5V and ESP12e is 3.3V device.

Small regulator LM1117, or similar, can do the job to drop Vcc to right level.

How ever data from rig comes still at 5V level. Simple way to fix this is to use 2 diodes in TxD that drops voltage around 1.2V with grounding resistor.
ESP12e is not very sensitive for low voltage 1-levels, but 0-levels should be very near of GND.

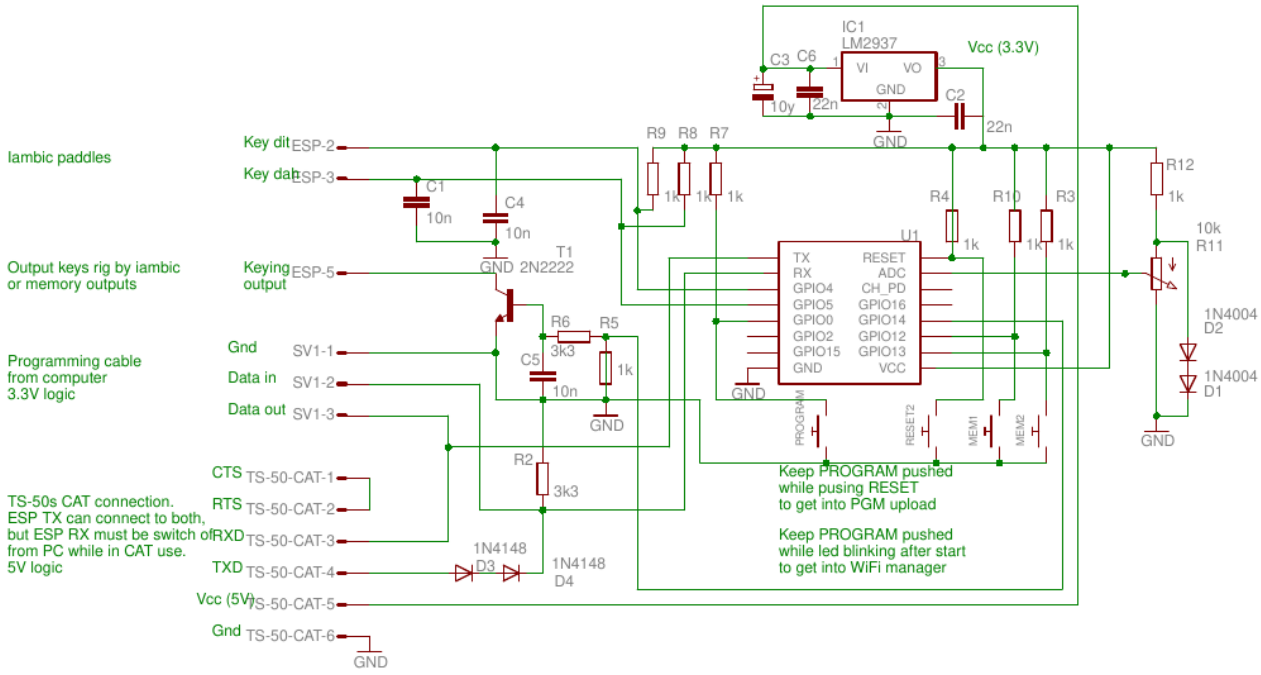
On the other hand 3.3V levels that ESP12e TXD pushes to rig is directly acceptable level for TS-50s.



Data direction can be found from service manual. Lines that have pull down resistors of 10k are transmitted from rig to pc.
Rig is receiving commands with lines without pull down resistors. Sounds funny, but so it is.

RTS and CTS should do some handshaking, but in tests I found out that best response can be found if they are just connected together and nowhere else.
ESP12e transmit data goes to rig’s receive data and rig's transmit data to esp12e receive data. 5V feed regulator and ground is common for supply voltage and data.

Schematic diagram



From diagram all connections can be seen.
It also shows that iambic paddles are connected. YES: ESP12e has also cw-keyer !

Program

Program is a modification from previous version.
Differences at keyer side are 2 memories (stored in Eeprom) and CW speed adjust with potentiometer.

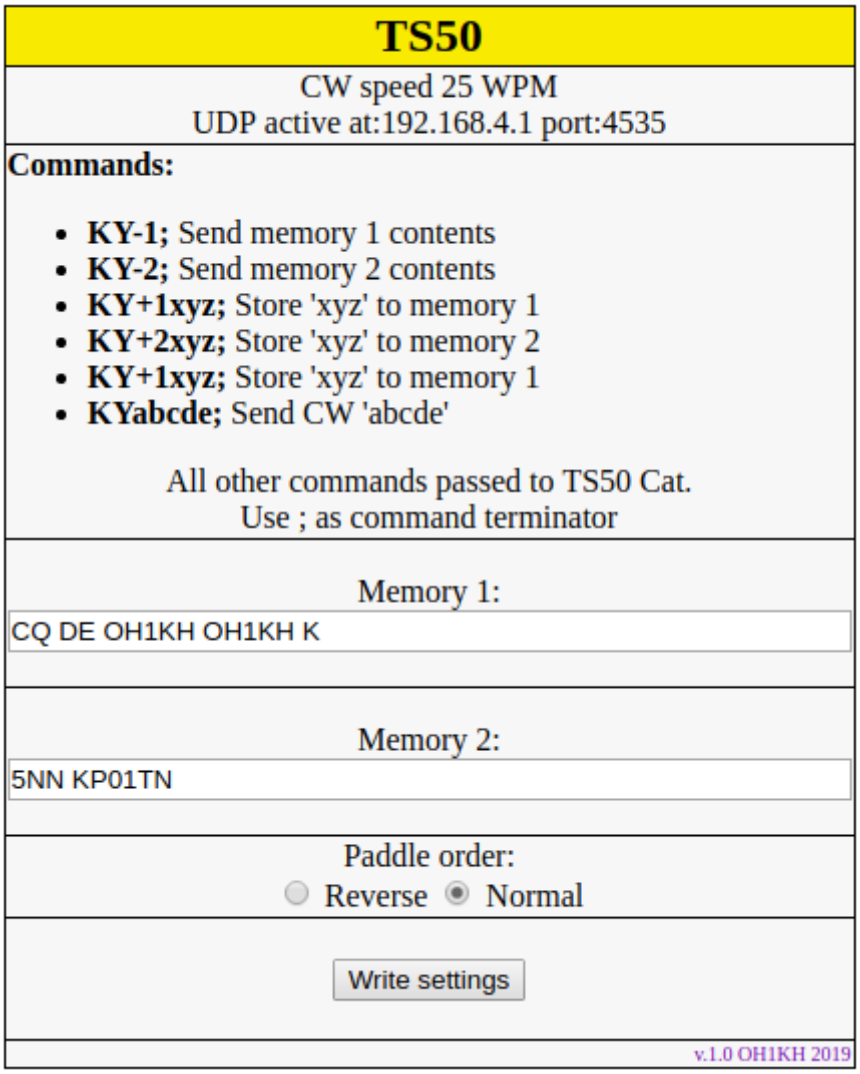
Connection to TS50 is made by connecting to WiFi network TS50 (no password). This version cannot connect to other WiFi access points.

If ”USE_UDP” is selected at compile phase then:

At http://192.168.4.1 (or any address, TS50 access point has captive DNS that directs all requests to ESP) there is a web page to modify memories and paddle order settings. It has also brief help for some CW output commands.
”KY” commands are modification from TS460’s send CW command. They are not passed to TS50 rig.
All other commands are passed to rig. Remember that Kenwood's ”end of command” is character ; No line feeds or carriage returns are needed after that.

Connection to rig cat is made using UDP at 192.168.4.1 port 4535

Web page:



If ”USE_UDP” is commented out at compile phase then:

Rig CAT is connected via TCP at 192.168.4.1 port 4535

There is no web page (as ESP can not handle Web and Tcp clients at same time). Memories can still be saved and executed via ”KY”-commands.

TCP connect may be easier to set up as UDP connect.

UDP connect that also allows WEB may be better on expedition usage when CW memories can be set using a smartphone.

Setting up communication

Windows users have to find a program that creates virtual serial port that can be connected with UDP or TCP over network.
As having no Windoze in house I can not give further help.

Linux uses can use 'socat' to create virtual serial port for network UDP or TCP.
While true; do ; done loop can be used to create virtual serial port that keeps itself up when host program closes. Without loop socat has to be started again every time before host rigctld starts.

TCP:

while true; do socat -v -x pty,link=$HOME/dev/ttyV0,wait-slave,raw tcp:192.168.4.1:4535,reuseaddress ; done

UDP:

while true; do socat -v -x pty,link=$HOME/dev/ttyV0,wait-slave,raw udp-sendto:192.168.4.1:4535; done

After setting up virtual serial port Hamlib rigctld can be started:

/usr/bin/rigctld -m 201 -r $HOME/dev/ttyV0 -t 4532 -s 4800 -vvvvv –set-conf=write_delay=5, post_write_delay=100, timeout=1500, retry=3

After rigctld is running a client program like cqrlog, wsjt-x, fldigi, qsstv, grig etc… can be started. Within those program's setup interface "Hamlib rigctld" at localhost:4532 must be used with rig model #2 (Net hamlib rigctld). Then those programs understand to access the previously started rigctld that then communicates to rig via virtual serial port.

With that kind of setup 2 or even more programs can run at same time without any conflict in rig serial traffic.
Also if rig model changes only parameters at separately started rigctld (-m 201 -r $HOME/dev/ttyV0) has to be changed to new rig model and serial port. All client programs still having same configs.

(see http://www.saunalahti.fi/~sakny/bin/cqrlog2/setting_rigctld_for_all_programs.pdf)

When everything works ok **socat** parameters **-v -x** and **rigctld** parameter **-vvvvv** can be removed as they only produce debug information to terminal.

–
Saku
OH1KH