ESP32-radio

This document describes the realization of an Internet radio based on an ESP32 WiFi module.

The ESP32 is the successor of the ESP8266. For the ESP8266 an Internet radio was build and now the software is poterd to the ESP32. The Internet radio described here uses the ESP32 as well as a VS1053 module to decrypt the MP3 stream and a 1.8 TFT color display to give some information about the radio station that's playing. This display has also a SD card slot. Mp3 tracks on an SD card can be played by the ESP32-radio.

Features:

- Can connect to thousands of Internet radio stations that broadcast MP3 audio streams.
- Can connect to a standalone mp3-file on a server.
- mp3 playlists supported.
- Can play MP3 files from SD card.
- Uses a minimal number of components.
- Has a preset list of a maximum of 100 favorite radio stations in a configuration file.
- Can be controlled by a tablet or other device through the built-in webserver.
- Can be controlled by MQTT commands. Status info is published by using MQTT.
- Can be controlled by commands on the serial input.
- Can be controlled by IR.
- Up to 14 input ports can be configured as control buttons like volume, skip to the next/previous/first/favorite preset station.
- The strongest available WiFi network is automatically selected. Passwords are kept as preferences in flash. Heavily commented source code, easy to add extra functionality.
- Debug information through serial output.
- Big ring buffer for smooth playback.
- Update of software over WiFi (OTA).
- Parameters like WiFi information and presets can be edited in the web interface.
- Plays iHeartRadio streams.
- Preset, volume, bass and treble settings are saved in preferences.

Software:

The software for the radio is supplied as an Arduino sketch that can be compiled for the ESP32 using the Arduino IDE version 1.8.2. No Arduino is required in this project.

The following libraries are used:

- WiFi – for establishing the communication with WiFI
- SPI – for communication with VS1053 and TFT display
- WiFiMulti - to select the strongest WiFi network
- Adafruit_GFX – for writing info on the TFT screen (if configurated)
- TFT_ILI9163C – driver for the TFT screen (if configurated)
- ArduinoOTA for software update over WiFi.
- PubSubClient to handle MQTT messages.
- TinyXML for decoding the iHeartRadio xml.
- SD and FS for reading from SD card.

The map with the ESP32-radio sketch must also contain the supplied headerfiles "index_html.h", "favicon_ico.h", "radio_css.h", "config_html.h", "defaultprefs.h", "mp3play.h" and "about_html.h". These files are included for the webinterface in PROGMEM.

Preferences

Preferences for ESP32-Radio are kept in Flash memory (NVS).

You may change the contents of NVS with the "Config"-button in the web interface. An example of the contents is:

```
mqttport = 1883
mqttuser = none
mqttpasswd = none
mqtttopic = P_espradio
mqttpubtopic = P_espradioIP
#
wifi_00 = NETGEAR-11/XXXXXX
wifi_01 = ADSL-11/YYYYYY
#
volume = 72
toneha = 0
tonehf = 0
tonela = 0
tonelf = 0
#
preset = 6
#
preset_00 = 109.206.96.34:8100                     #  0 - NAXI LOVE RADIO, Belgrade, Serbia
preset_01 = airspectrum.cdnstream1.com:8114/1648_128 #  1 - Easy Hits Florida 128k
preset_02 = us2.internet-radio.com:8050            #  2 - CLASSIC ROCK MIA WWW.SHERADIO.COM
preset_03 = airspectrum.cdnstream1.com:8000/1261_192 #  3 - Magic Oldies Florida
preset_04 = airspectrum.cdnstream1.com:8008/1604_128 #  4 - Magic 60s Florida 60s Classic Rock
preset_05 = us1.internet-radio.com:8105            #  5 - Classic Rock Florida - SHE Radio
preset_06 = icecast.omroep.nl:80/radio1-bb-mp3     #  6 - Radio 1, NL
preset_07 = 205.164.62.15:10032                    #  7 - 1.FM - GAIA, 64k
preset_08 = skonto.ls.lv:8002/mp3                  #  8 - Skonto 128k
preset_09 = 94.23.66.155:8106                      #  9 - *ILR CHILL and GROOVE
preset_10 = ihr/IHR_IEDM                           # 10 - iHeartRadio IHR_IEDM
preset_11 = ihr/IHR_TRAN                           # 11 - iHeartRadio IHR_TRAN
#
gpio_00 = uppreset = 1
gpio_12 = upvolume = 2
gpio_13 = downvolume = 2
gpio_14 = stop
gpio_17 = resume
gpio_21 = station = icecast.omroep.nl:80/radio1-bb-mp3
#
ir_40BF = upvolume = 2
ir_C03F = downvolume = 2
```

Lines starting with "#" are comment lines and are ignored. Comments per line (after "#") will also be ignored, except for the "preset_" lines. The comments on the "preset_" lines are used to identify the presets in the web interface. Note that the numbers ranges from 00 to 99. If the highest numbered station is reached, the next station will be 00 again. URLs with mp3 files or mp3 playlists (.m3u) are allowed.

Presets starting with "ihr/" are iHeartRadio stations.

The "gpio_" lines specify input pins and the command that is executed if the input pin goes from HIGH to LOW.

The "ir_" lines specify IR-codes and the command that is executed if the IR-code is received.

The preferences can be edited in the web interface. Changes will in some cases be effective after restart of the Esp-radio.

Using Winamp to find out the correct preset line for a station.

Press Alt-3 in the main window (left picture). You will see info for the playing station (right picture).
The top line (with "http") will contain the information for the preset, in this example:

"us1.internet-radio.com:8105". The complete line for this station in the preferences would be:

```
preset_05 = us1.internet-radio.com:8105          #  5 - Classic Rock Florida - SHE Radio
```

Optional:

Digital control through input pins:

Normally the radio is controlled by the web interface. However, free digital inputs (GPIO) may be connected to buttons to control the radio. Their function can be programmed using the webinterface.
Here is an overview of all free GPIOs.

GPIO	Remarks
0	Flash, free for input
12	
13	
14	
17	
22	
25	
26	
27	
32	
33	
34	No pull-up

You can assign commands to the digital inputs by adding lines in the configuration (Webinterface, "Config" page). Examples:

```
gpio_00 = uppreset = 1
gpio_12 = upvolume = 2
gpio_13 = downvolume = 2
gpio_14 = stop
gpio_17 = resume
gpio_21 = station = icecast.omroep.nl:80/radio1-bb-mp3
```

In this example the ESP32-Radio will execute the command "uppreset=1" if GPIO0 will go from HIGH to LOW. The commands are equal to the commands that are handled by the serial input or by the MQTT interface.

IR Interface.

The radio can be controlled by an IR remote control like this:

To use this interface, the output pin of a VS1838B receiver must be connected to pin GPIO35 of the ESP32:

The software will read the raw code of the IR transmitter, making it possible to use virtually any remote control to be used. I tested it with the 21 button remote as well as with an LG TV remote.
To assign functions to the buttons, watch the debug log output while pressing a button. For example, press the +volume button. You will see something like:

```
D: IR code 807F received, but not found in preferences!
```

Now add the command:

```
ir_807F = upvolume = 2
```

to the preferences in the config page of the web interface. Likewise you can assign functions to all buttons, for example:

```
ir_8A31 = uppreset = 1
ir_719A = station = us1.internet-radio.com:8105
ir_1F6B = mute
```

Hardware:

The radio is built with the following widely available hardware:

- An ESP-32 module. This is basically an ESP32 on a small print. I used a DOIT ESP32 Development Board. See figure 1 below. The ESP32 is running on 160 MHz.
- A VS1053 module. See figure 2.
- A 1.8 inch color TFT display. See figure 3..
- Two small speakers.
- A Class D stereo amplifier to drive the speakers. Best quality if powered by a separate power source.
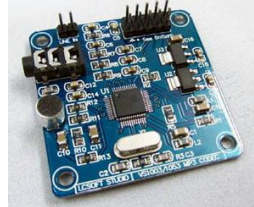
Here is a picture of the radio in a test configuration. The VS1053 is connected, but not the TFT:
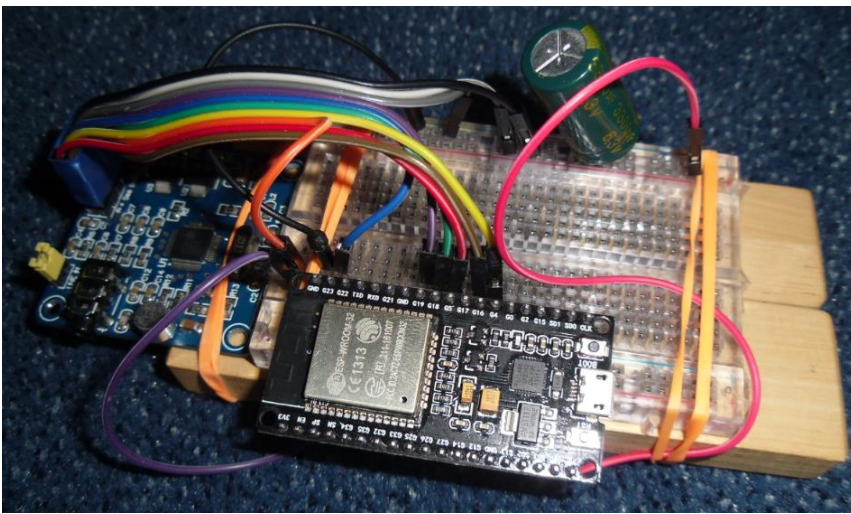
The module is not very breadboard friendly. The (hanging over) +5 Volt of the ESP32 is wired to the 5Volt rail of the breadboard. The big capacitor (3300 µF) is added to allow powering the module from USB. Without it, the USB has insufficient power to drive the ESP32.

The radio may be powered by a 5 V adapter. The radio will function on single LiPo cell as well, so I added a small charge circuit powered by the 5 V input. The amplifier uses a separate LiPo cell to minimize noise caused by the ESP32. The TFT and VS1053 work on 3.8 to 5 Volt.

I used a small perforated board to connect the ESP32 module and the TFT and to mount it in a small speaker box. The TFT is visible through a hole in the front of the box:

Wiring:

The logic wiring in the table below. The analog amplifier and the speakers are not included.

```
ESP32dev Signal  Wired to LCD     Wired to VS1053      SDCARD  Wired to the rest
-------- ------  --------------   -------------------  ------  ----------------
GPIO16   -       -                pin 1 XDCS           -       -
GPIO5    -       -                pin 2 XCS            -       -
GPIO4    -       -                pin 4 DREQ           -       -
GPIO2    -       pin 3 D/C        -                    -       -
GPIO18   SCLK    pin 5 CLK        pin 5 SCK            CLK     -
GPIO19   MISO    -                pin 7 MISO           MISO    -
GPIO23   MOSI    pin 4 DIN        pin 6 MOSI           MOSI    -
GPIO21   -       -                -                    CS      -
GPIO15   -       pin 2 CS         -                    -       -
GPI03    RXD0    -                -                    -       Reserved serial input
GPIO1    TXD0    -                -                    -       Reserved serial output
GPIO34   -       -                -                    -       Optional pull-up resistor
GPIO35   -       -                -                    -       Infrared receiver VS1838B
-------  ------  --------------   -------------------  ------  ----------------
GND      -       pin 8 GND        pin 8 GND                    Power supply GND
VCC 5 V  -       pin 7 BL         -                            Power supply
VCC 5 V  -       pin 6 VCC        pin 9 5V                     Power supply
EN       -       pin 1 RST        pin 3 XRST
```

Amplifier and power circuit.

The amplifier is a class D stereo amplifier. If the power is shared with the power supply of the radio, you will hear much noise. So I used a separate LiPo battery (Batt 2) for the amplifier.

During operation only Batt 1 will be charged. If the radio is switched off, both batteries will be recharged by the LiPo charger. S1a, S1b and S1c is a triple On-On switch.

Note that there may be high currents in the "off"-position if Batt 1 is fully discharged. Use protected batteries only!
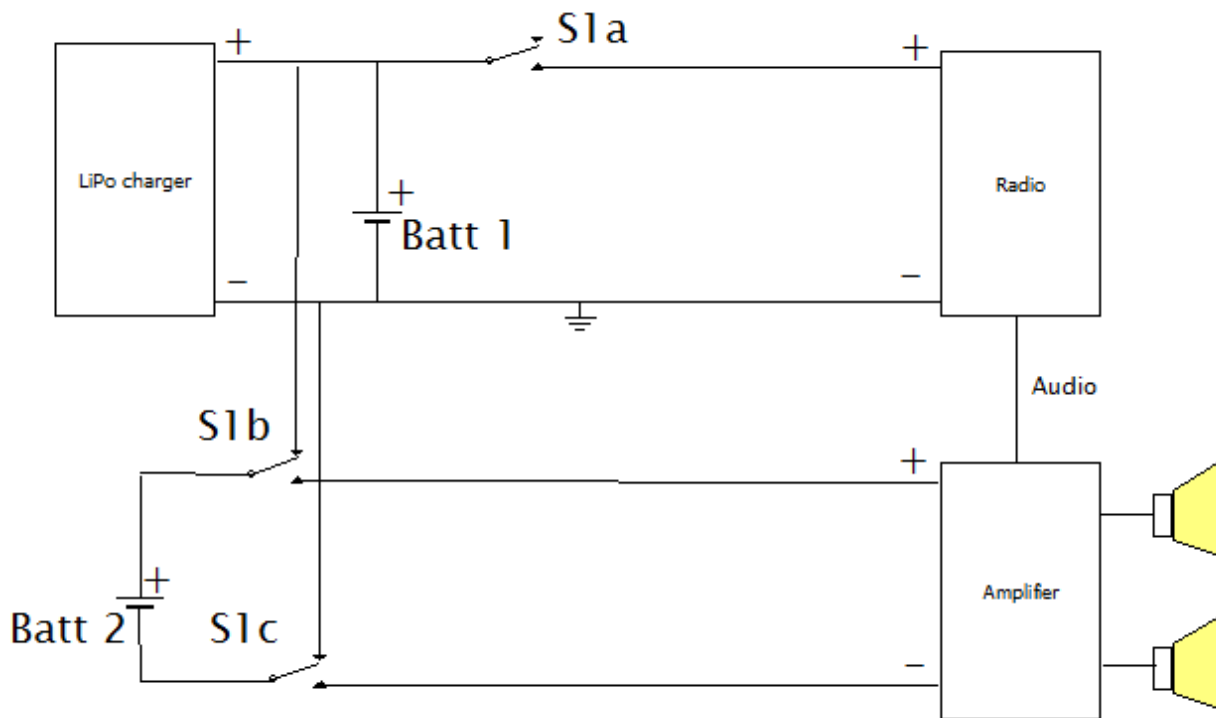
Web interface.

The web interface is simple and can be adapted to your needs. The basic idea is to have a html page with embedded javascript that displays an interface to the radio. Command to the radio can be sent to the http server on the ESP32. The IP address of the webserver will be displayed on the TFT during startup. The webpages are defined in PROGMEM.

Capabilities of the webserver:

Let's assume that the IP of the ESP32-radio is 192.168.2.12. From your browser you can show a simple root page by entering the following URL: http://192.168.2.12. This will display the index_html.h page from the PROGMEM as well as favicon_ico.h.

If your computer is configured for mDNS, you can also use http://ESP32Radio.local in your browser.

The following simple web interface will be displayed:

Clicking on one of the available buttons will control the ESP32-radio. The reply of the webserver will be visible in the status box below the buttons. A click will be translated into a command to the ESP32-radio in the form:

    http://192.168.2.13/?<parameter>=<value>

For example: http://192.168.2.13/?upvolume=2

Not all functions are available as buttons in the web interface shown above. Commands may also come from MQTT or serial input. Not all commands are meaningful on MQTT or serial input. Working commands are:

```
preset      = 12                        Select start preset to connect to
uppreset    = 1                         Select next preset or playlist entry
downpreset  = 1                         Select previous preset or playlist entry
preset_00   = <mp3 stream>              Specify station for a preset 00-99
volume      = 95                         Percentage between 0 and 100
upvolume    = 2                         Add percentage to current volume
downvolume  = 2                         Subtract percentage from current volume
toneha      = <0..15>                    Setting treble gain
tonehf      = <0..15>                    Setting treble frequency
tonela      = <0..15>                    Setting bass gain
tonelf      = <0..15>                    Setting treble frequency
station     = <mp3 stream>              Select new station (will not be saved)
station     = <URL>.mp3                 Play standalone .mp3 file (not saved)
station     = <URL>.m3u                 Select playlist (will not be saved)
xml         = <Mountpoint>              Select iHeartRadio station (not saved)
mute                                    Mute (or unmute) the music
stop                                    Stop player
resume                                  Resume player
wifi_00     = mySSID/mypassword         Set WiFi SSID and password *)
mqttbroker  = mybroker.com              Set MQTT broker to use *)
mqttport    = 1883                      Set MQTT port (default 1883) to use *)
mqttuser    = myuser                    Set MQTT user for authentication *)
mqttpasswd  = mypassword                Set MQTT password for authentication*)
mqttprefix  = none                      Set MQTT prefix for pub/sub.
mp3track    = <nodeID>                  Play MP3 track from SD card
status                                  Show current URL to play
test                                    For test purposes
debug       = 0 or 1                    Switch debugging on or off
reset                                   Restart the ESP32
```

Commands marked with "*)" are sensible during power-up only.

Station may also be of the form "skonto.ls.lv:8002/mp3". The default port is 80.
Station may also point to an mp3 playlist. Example: "www.rockantenne.de/webradio/rockantenne.m3u".
Station may be an .mp3-file on a remote server. Example: "www.stephaniequinn.com/Music/Rondeau.mp3".
Station may also point to a local .mp3-file on SPIFFS. Example: "localhost/friendly.mp3". Note that there is only limited space to store .mp3-files on SPIFFS.

It is allowed to have multiple (max 100) "preset_" lines. The number after the "_" will be used as the preset number.
The comment part (after the "#") will be shown in the webinterface.

It is also allowed to have multiple "wifi_" lines. The strongest Wifi access point will be used.

Configuration

The "Config" button will bring up a second screen. Here you can edit the preferencese. The available Wifi networks are listed as well. The config screen will be shown automatically if the ESP32 cannot connect to one of the WiFi stations specified in the preferences. In that case the ESP32 will act as an accesspoint with the name "ESP32Radio". You have to connect to this AP with password "ESP32Radio". Than the ESP32-radio can be reached at http:// 192.168.4.1.

Then you have the opportunity to edit the preferences. A "default" button is available to fill the editbox with some example data. To a quick start, just fill in you WiFi network name and password and click "Save". Restart the radio afterwards.

After changing the contents of this page, it must be saved to the preferences by clicking the "Save" button. Changes will have effect on the next restart of the ESP-radio, so click the "Restart" button.

MQTT interface.

The MQTT interface can handle the same commands as the web interface.

As publish command on a Linux system may look like:

```
$ mosquitto_pub -h broker.hivemq.com -t espradio -m volume=80
```

Note that the broker in this example is heavily used, this may cause some delay. If you use your own broker the reaction on commands will be much better.

Remove the lines starting with "mqtt" if no MQTT is required or set mqttbroker to "none".

You can use an MQTT client like https://kamilfb.github.io/mqtt-spy/ to view the MQTT interface.
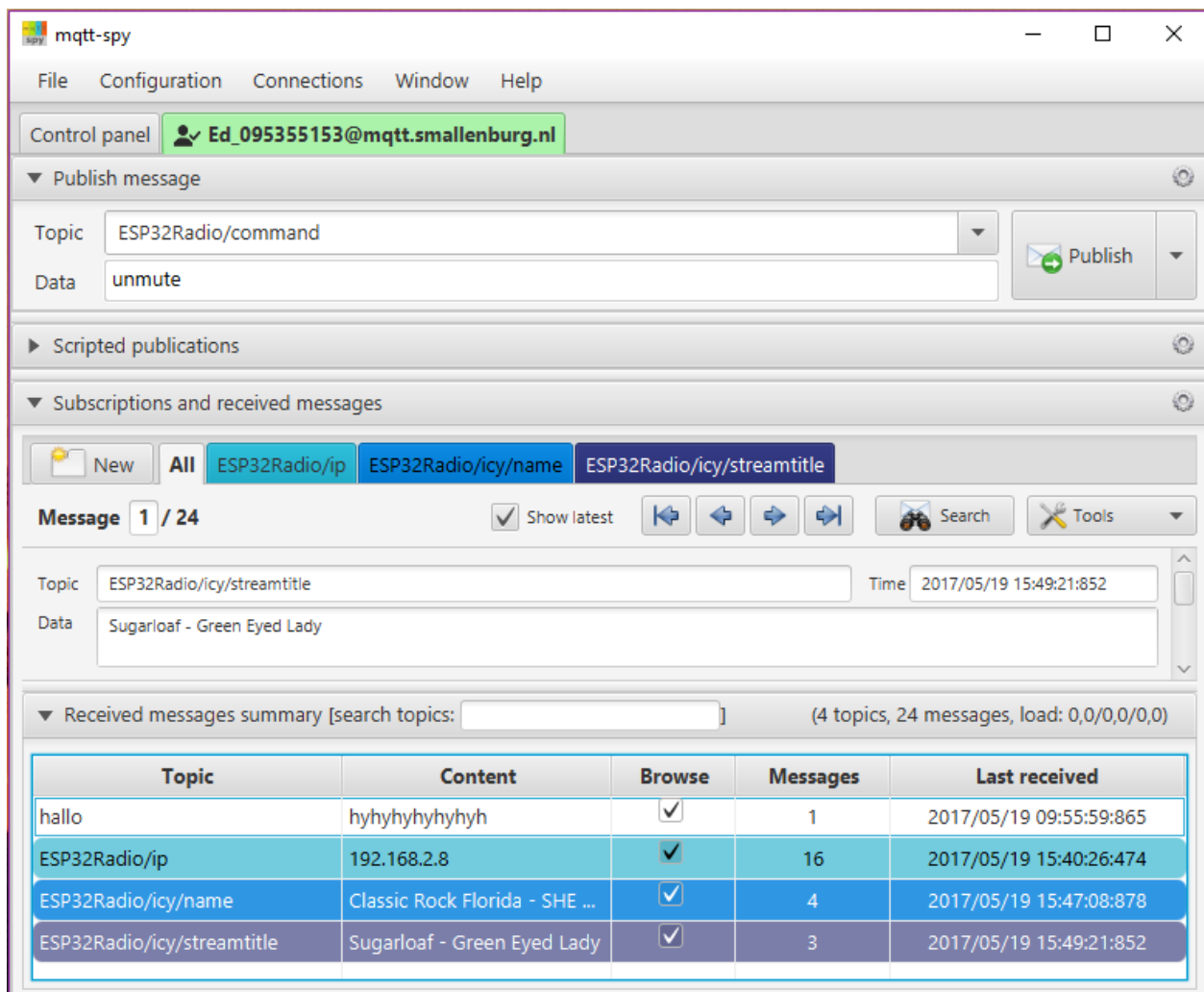
Subscribe to "ESP32Radio/ip" and you will see the IP-address of your radio. You will see the IP address of your radio or the IP address of a different user. So be sure to use unique names for your topics. This can be accomplished by specifying a unique prefix in the preferences, see below.

The parameters in preferences for this example are:

```
# MQTT broker, credentials and topic  to subscribe
mqttbroker = broker.hivemq.com        # Broker to connect with
mqttprefix = none                     # Prefix for pub/sub.  Default is part of MAC-address.
mqttport = 1883                       # Portnumber (1883 is default)
mqttuser = none                       # (No) username for broker
mqttpasswd = none                     # (No) password for broker
mqtttopic = espradio                  # Topic for receiving commands
#
```

In this example I published the command "unmute" to the radio. The radio published the IP-address 192.168.2.8 to the broker (once every 10 minutes).

MQTT PUB topics.
In this version 3 topics will be published to MQTTand can be subscribed to by an MQTT client:

prefix/ip	The IP-address of the ESP32-Radio
prefix/icy/name	The name of the station
prefix/icy/streamtitle	The name of the stream
prefix/nowplaying	Track information

Command to control the ESP32-Radio are to be published to "prefix/command".

TFT library patches.
The display is used in mode "3". The TFT-library has a bug for this mode. Height and width are reversed here.

To correct it, find the source code TFT_ILI9163C.cpp on your computer (mine is at "documents/Arduino/libraries/ TFT_ILI9163C-master") and change the 2 lines 1096 and 1097 to:

```
_width  = _TFTHEIGHT;//-__OFFSET;
_height = _TFTWIDTH;
```

Find TFT_ILI9163C_settings.h and edit it so that the right board will be selected. As an example you will find the configuration for the "blue 1.8 SPI 128x160 board" in the TFT_ILI9163C-master.zip file.

Alternative way of configuration.
An extra sketch "Esp32_radio_init" is supplied as an alternative to initialize the preferences (in Non-Voltile Storage of the ESP32). Just change lines 39 and 40 (the specs for WiFi networks) to match your network(s).

Upload and run the sketch once and then load the EPS32-Radio.

Debug (serial 115200 Baud) output.

This is an example of the debug output.

```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
ets Jun  8 2016 00:22:57

rst:0x10 (RTCWDT_RTC_RESET),boot:0x17 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0x00
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0008,len:8
load:0x3fff0010,len:160
load:0x40078000,len:10632
load:0x40080000,len:252
entry 0x40080034

D: Starting ESP32-radio Version Wed, 19 May 2017 10:30:00 GMT...  Free memory 195084
D: GPIO0 is HIGH
D: GPIO12 is HIGH
D: GPIO13 is HIGH
D: GPIO14 is HIGH
D: GPIO17 is HIGH
D: GPIO21 is HIGH
D: GPIO22 is HIGH
D: GPIO25 is HIGH
D: GPIO26 is HIGH
D: GPIO27 is HIGH
D: GPIO32 is HIGH
D: GPIO33 is HIGH
D: GPIO34 is LOW
D: GPIO35 is LOW
D: GPIO0 will execute uppreset = 1
D: GPIO12 will execute upvolume = 2
D: GPIO13 will execute downvolume = 2
D: GPIO14 will execute stop
D: GPIO17 will execute resume
D: GPIO21 will execute station = icecast.omroep.nl:80/radio1-bb-mp3
D: Added SSID NETGEAR-11 to list of networks
D: Added SSID ADSL-11 to list of networks
D: Scan Networks
D: Scan completed
D: Number of available networks: 5
D:  1 - HZN240825082           Signal: -67 dBm, Encryption WPA2_PSK,
D:  2 - NETGEAR-11             Signal: -67 dBm, Encryption WPA2_PSK,  Acceptable
D:  3 - ADSL-11                Signal: -85 dBm, Encryption WPA_WPA2_PSK,  Acceptable
D:  4 - Roulet 9 Gast          Signal: -85 dBm, Encryption WPA_WPA2_PSK,
D:  5 - Roulet 9               Signal: -86 dBm, Encryption WPA_WPA2_PSK,
D: End of list
D: Command: mqttbroker with parameter mqtt.smallenburg.nl
D: Command: mqttprefix with parameter ESP32Radio
D: Command: mqttuser with parameter edzelf
D: Command: mqttpasswd with parameter ******
D: Command: volume with parameter 74
D: Command: toneha with parameter 0
D: Command: tonehf with parameter 0
D: Command: tonela with parameter 0
D: Command: tonelf with parameter 0
D: Command: preset with parameter 6
D: Reset VS1053...
D: End reset VS1053...
D: Slow SPI,Testing VS1053 read/write registers...
D: Fast SPI, Testing VS1053 read/write registers again...
D: endFillByte is 0
D: Connect to WiFi
D: Connected to NETGEAR-11
D: IP = 192.168.2.8
D: Start server for commands
D: MQTT uses prefix ESP32Radio
D: Init MQTT
D: (Re)connecting number 1 to MQTT mqtt.smallenburg.nl
D: STOP requested
D: Song stopped correctly after 0 msec
D: New preset/file requested (6/0) from icecast.omroep.nl:80/radio1-bb-mp3
D: Connect to new host icecast.omroep.nl:80/radio1-bb-mp3
D: Connect to icecast.omroep.nl on port 80, extension /radio1-bb-mp3
D: Connected to server
D: Publish to topic ESP32Radio/ip : 192.168.2.8
D: Server: Icecast 2.4.2
D: Content-Type: audio/mpeg
D: Cache-Control: no-cache
D: Expires: Mon, 26 Jul 1997 05:00:00 GMT
D: Pragma: no-cache
D: icy-br:192
D: ice-audio-info: bitrate=192
D: icy-br:192
D: icy-genre:Talk
D: icy-name:Radio 1
D: icy-pub:0
D: icy-url:http://www.radio1.nl
D: icy-metaint:16000
D: Switch to DATA, bitrate is 192
D: Publish to topic ESP32Radio/icy/name : Radio 1
D: First chunk:
D: 52 11 01 10 85 8C D1 F4
D: 2E 63 94 EB 2F CD AE 0A
D: 02 70 51 10 51 F8 2D 44
D: 65 90 8E 57 23 D1 2F 90
D: Metadata block 48 bytes
D: Streamtitle found, 47 bytes
D: StreamTitle='NPO Radio 1 - Nieuws en Co - NOS';
D: Publish to topic ESP32Radio/icy/streamtitle : NPO Radio 1 - Nieuws en Co - NOS
```