wiFred

WiFi throttle for model railroads using the wiThrottle protocol

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
0.2-WIP	2-7-2021	Converting to asciidoc	HR

Contents

1 Specification wishlist 1

2 Development history 2

3 Wireless clock 2

Abstract

This document describes the usage and configuration of the wiFred - a very simple wireless throttle for model railroads to connect to wiThrottle servers like JMRI. It also contains schematics and BOMs for the device - for both LiPo battery versions in active development - as well as programming instructions and assembly tips, and also an overview of options for the server side of things.

The most recent version of this document can be found at https://newheiko.github.io/wiFred, https://github.com/newHeiko/wiFred/raw/master/documentation/docu.pdf and https://github.com/newHeiko/wiFred/blob/master/documentation/docu.adoc.

If you want to know more about the development history of the wiFred, skip ahead to section [?] - otherwise read on with section [?].

\thispagestyle \maketitle \clearpage \tableofcontents \clearpage wiFred Wireless throttle hardware {#throttle}

\vspace{0.5em} \centering Red LED Green LED (Left) Green LED (Right) Status ---------------------- - ----------------- ------------------ -- -- Slow Blinking (0.5 Hz) Off Off Trying to connect to WiFi network Fast Blinking (2 Hz) Off Off Successful WiFi connection, trying to connect to wiThrottle server and acquire locos Off Off On Regular operation, forward direction Off On Off Regular operation, reverse direction Off Flashing On Emergency stop, forward direction. Also happens when switching direction with speed potentiometer not at zero Off On Flashing Emergency stop, reverse direction. Also happens when switching direction with speed potentiometer not at zero Off Off Blinking Battery low, regular operation, forward direction Off Blinking Off Battery low, regular operation, reverse direction Off Flashing Blinking Battery low, Emergency stop, forward direction Off Blinking Flashing Battery low, Emergency stop, reverse direction Short flashes Off Off Throttle in low-power mode Off Off Off Battery empty or no battery inserted On Off Off No connection to existing WiFi network. Created internal configuration WiFi network On On On Configuration mode enabled while connected to existing WiFi network. All locos emergency stop to avoid runaways. Push SHIFT + ESTOP again to exit configuration mode

```
: LED patterns and their meaning on the wiFred
throttle[]{label="ledTable"}
```

\vspace{0.5em} To recover from an emergency stop, turn speed potentiometer to zero to re-gain control.

Quickstart Guide

```
Follow these steps for a new throttle (see later chapters for ↩
    more
explanation or if you run into trouble)

-5. Use PCB to determine positions of holes and cutouts in ↩
    housing

-4. Make said cutouts
```

```
-3. Solder components to PCB

-2. Connect lithium battery to PCB, charge with Micro USB charger ↩
     if
    required

-1. Flash firmware to ATMega~328P

0.  Move any of the loco selection switches to "enabled" to power
    ESP8266, then flash firmware to ESP8266, move loco selection  ↩
        switch
    back to "disabled" to turn off power to ESP8266 again

1.  Test fit PCB into housing, removing plastic parts of housing  ↩
    as
    required

2.  Fit PCB into housing, insert four screws to fix PCB to  ↩
    housing

3.  Fit lithium battery into other half of housing, fix with  ↩
    double
    sided tape or similar, taking care that the battery will not  ↩
        be
    squeezed or pinched by any parts on the PCB when the housing  ↩
        is
    closed

4.  Make sure communication jumpers are set correctly, close  ↩
    housing and
    fix back cover with two screws

5.  Add throttle knob

6.  Move any loco selection switch to "enabled" to power ESP8266

7.  Using any WiFi client (laptop, smartphone, tablet\...), find  ↩
    and
    connect to network *wiFred-configXXXX*

8.  Using any web browser, navigate to *http://192.168.4.1* or
    *http://config.local*

9.  Enter your WiFi configuration or scan for available networks  ↩
    (and a
    throttle ID if you like -- highly recommended to easier tell  ↩
        them
    apart) **and hit the *Submit*-Button**

10. For every loco you want to control with this throttle, enter ↩
    the
    appropriate details below

11. **Hit the *Submit*-Button** for every loco

12. Configure function settings for each loco on the respective ↩
    sub
    pages if required

13. Restart the throttle by clicking on
     ‘Restart system to enable new WiFi settings‘

Your throttle should now be ready to use and connect to your ↩
    wiThrottle
server on startup. Refer to the chapters below if it does not or ↩
    contact
the author of this document.

Before operating the throttle, fully charge the battery which ↩
    will also
calibrate the internal battery voltage measurements. Before the ↩
    first
full charge, the throttle may not shut down when the battery is ↩
    empty
which can lead to damage to the battery. This can be checked by
comparing the device’s battery voltage measurement on the status ↩
    subpage
of the configuration website to voltage readings from a ↩
    multimeter on
the battery terminals -- an accuracy of 50 mV to 100 mV is OK.

Usage
-----

\centering
![Controls and features of the
wiFred-throttle[]{label="throttleControls"}](images/_DSC0136){# ↩
    throttleControls
height="100mm"}

Figure~[1](#throttleControls){reference-type="ref"
reference="throttleControls"} shows the controls of the wireless
throttle. They consist of the following:
```

```
-   Four loco selection switches (loco 1 on the left, loco 4 on  ↩
    the
    right, move towards speed potentiometer to enable)

-   Speed potentiometer (Counter-clockwise endstop: Stop,  ↩
    clockwise
    endstop: Full speed)

-   Direction switch -- move right for forward movement, left for
    reverse movement

-   Black function keys F0 to F8

-   Yellow shift key to trigger F9-F16 and turn on flashlight  ↩
    function

-   Red emergency stop key

-   Two green direction indicator LEDs next to speed  ↩
    potentiometer

-   Red status LED next to speed potentiometer

-   Red charging indicator LED at lower end of device -- lit  ↩
    while
    charging

-   Green fully charged indicator LED at lower end of device --  ↩
    lit when
    fully charged as long as charger still connected

As soon as any of the loco selection switches is moved into the
"enabled" position, the throttle will boot up and try to connect  ↩
    to a
wireless network. When all four loco selection switches are " ↩
    disabled",
the throttle will disconnect from the wireless network after a  ↩
    grace
period of five seconds. The device will then go into low power  ↩
    mode, in
which the battery will last for more than a year.

If no connection to the network configured into the device can be
established within 60 seconds, the throttle will create it's own
wireless network named *wiFred-config* plus four hex digits taken ↩
     from
the MAC address of the throttle WiFi interface, for example
```

```
*wiFred-config0CAC*, to enable configuration as described in
section~[2](#config){reference-type="ref" reference="config"}.

Four different locos with long DCC addresses can be assigned to  ↩
    the four
loco selection switches. Commands derived from the speed  ↩
    potentiometer,
the direction switch and the function keys will be transmitted to ↩
     all
selected locos (near) simultaneously, with a certain translation  ↩
    table
enabling some locos to go backwards when others go forwards and  ↩
    also
limiting function keys to some of the four locos only -- this is
described in more detail in
sections~[2.2.4](#throttle_LocoConf){reference-type="ref"
reference="throttle_LocoConf"}
and~[2.2.5](#throttle_FunctionConf){reference-type="ref"
reference="throttle_FunctionConf"}.

Pushing the red emergency stop key will cause the throttle to  ↩
    send an
emergency stop signal to all four locos attached. After an  ↩
    emergency
stop, turn the speed potentiometer to zero to re-enable control  ↩
    of the
locos.

Pushing the red emergency stop key while holding down the shift  ↩
    key will
place the device into configuration mode (as well as issueing an
emergency stop to all attached locos). See
section~[2](#config){reference-type="ref" reference="config"} for ↩
     more
details on how to access the throttle to do the configuration.

Any change in the loco selection switches will cause the throttle ↩
     to
send an emergency stop command to all attached locos. This makes  ↩
    sure
that any loco that is deselected will stop on the layout and  ↩
    avoids
newly selected locos suddenly taking off at speed. The same is  ↩
    true for
a change in the direction switch, to avoid high-speed reverse  ↩
    maneuvers.
Turn the speed potentiometer to zero to re-enable control of the  ↩
```

```
    locos.

When the battery is low, the device will not re-activate before  ↩
    charging
the batteries, but continue operating for approximately an hour  ↩
    if
active. When the battery is empty, it will disconnect and enter  ↩
    low
power mode. Expected runtime is around 20 hours of full time  ↩
    operations,
more if the throttle is placed in low power mode when the locos  ↩
    are not
running.

During startup and operation, the LEDs will show the patterns  ↩
    explained
in table~[\[ledTable\]](#ledTable){reference-type="ref"
reference="ledTable"}.

Charging the wiFred
-------------------

The wiFred can be charged through the Micro-USB connector at the  ↩
    lower
end of the device. Maximum charging current is approximately 400
    mA and
the device does not communicate with the USB host, so technically ↩
     there
is no guarantee that charging from a USB cable will work, but  ↩
    most
chargers, computer ports or power banks do not check the current  ↩
    before
powering up.

As long as the battery is being charged, the red charging  ↩
    indicator LED
will be lit. When the battery is fully charged, the green charged
indicator LED will be lit as long as the charger is still  ↩
    connected.
Expected charging time is around five to six hours for a full  ↩
    charge.

Even while charging, the device can still be operated ( ↩
    particularly
helpful with a power bank) but since the operating current will  ↩
    come out
of the battery, the battery will never be fully charged.
```

```
If both charging status LEDs light up when a charging cable is
connected, probably the internal connection to the battery is  ↩
    faulty.

Hardware description
--------------------

The wiFred hardware is centered around an ESP8266 for the WiFi
connection. The ESP8266 communicates through it’s serial port  ↩
    with an
ATMega~328P microcontroller which manages the power, controls the ↩
     LEDs,
reads the loco selection switches, speed potentiometer, direction ↩
     switch
and pushbutton switches for functions and emergency stop. The
communication goes through a 2x3 pin header which enables the  ↩
    user to
connect a programming cable to the same serial port if removing  ↩
    the
jumpers.

Optionally, two white 5
    mm-LEDs protruding from the top of the PCB can
be installed to serve as a flashlight. They are driven by a
constant-current source directly from the battery and enabled  ↩
    when
pushing the yellow SHIFT key.

The wiFred is powered by a single cell LiPo battery. The ATMega ↩
    ~328P is
connected directly to the LiPo cell, going into sleep mode when  ↩
    no loco
selection switch is active, thereby reducing the power  ↩
    consumption to
less than 1
    mA. The ESP8266 is powered by a low-drop linear voltage
regulator with an output voltage of 3 V which is disabled by the
ATMega~328P when the device goes into standby.

The schematic is split into several pages and can be found in
figures~[2](#schematicPage1){reference-type="ref"
reference="schematicPage1"} to~[5](#schematicPage4){reference- ↩
    type="ref"
reference="schematicPage4"}. It has been created with kicad and  ↩
    is
available on the github repository at
*http://github.com/newHeiko/wiFred* along with the PCB design.
```

```
\centering
![Master schematic sheet with battery connector, charging circuit ↩
     and
power
supply[]{label="schematicPage1"}](images/wfred_rev2){# ↩
    schematicPage1
width="\textwidth"}

\centering
![Schematic sheet including ESP8266 for WiFi connection with  ↩
    bootloader
enabling jumper and connection to programming
cable[]{label="schematicPage2"}](images/wfred-wifi- ↩
    Wifi_connection){#schematicPage2
width="\textwidth"}

\centering
![Schematic sheet including ATMega 328P along with crystal and in ↩
     system
programming
header[]{label="schematicPage3"}](images/wfred-controller_rev2- ↩
    Controller){#schematicPage3
width="\textwidth"}

\centering
![Schematic sheet including pushbutton switches, loco selection
switches, direction switch, speed potentiometer and flashlight  ↩
    LEDs with
controller[]{label="schematicPage4"}](images/User_interface_rev2- ↩
    User_Interface){#schematicPage4
width="\textwidth"}

Hints for building the wiFred
-----------------------------

The PCB has holes in the center of the LED footprints to enable
transferring their positions to a StrapuBox housing with a sharp  ↩
    needle
or to drill pilot holes with a 1
    mm drill. For all other holes, there is
a drill jig available which also allows the drilling of pilot  ↩
    holes for
the pushbutton switches, the direction control switch and the  ↩
    speed
potentiometer. Figure~[7](#transferHoles){reference-type="ref"
reference="transferHoles"} shows the process and it's results.  ↩
    Holes for
the pushbutton switches should be drilled at 3.5
```

```
    mm diameter. Holes for
the LEDs should be drilled at 3
    mm diameter and holes for the speed
potentiometer at 8 mm, for the direction switch at 6.5
    mm diameter. The
cutouts for the loco selection switches are best drilled at 5
    mm or
5.5 mm and extended to fit when the PCB is assembled with a sharp ↩
    hobby
knife and a file.

\centering
![Using the original PCB and the drilling jig to transfer the ↩
    positions
of the holes to the housing -- better results will be achieved ↩
    when the
PCBs are screwed in
position[]{label="transferHoles"}](images/_DSC0124 "fig:"){# ↩
    transferHoles
width="0.49 \textwidth"} ![Using the original PCB and the ↩
    drilling jig
to transfer the positions of the holes to the housing -- better ↩
    results
will be achieved when the PCBs are screwed in
position[]{label="transferHoles"}](images/_DSC0128 "fig:"){# ↩
    transferHoles
width="0.49 \textwidth"}

The remaining assembly is a basic exercise in installing all the
components to the PCB, listed in
table~[\[wiFredBOM\]](#wiFredBOM){reference-type="ref"
reference="wiFredBOM"}. From assembling the prototypes, the ↩
    suggested
order of installing the components is as follows:

1.  IC101, IC102, IC201 (note: Rotate PCB so Designator is right ↩
    side
    up, then Pin 1 is on top left) and IC301

2.  X201 and D201

3.  USB connector CON101

4.  Capacitors and Resistors in 0805 size (first those on the ↩
    same side
    as the items before) [\[0805devices\]]{#0805devices
    label="0805devices"}

5.  U401
```

```
6.  Capacitors and Resistors not installed in
    step~[\[0805devices\]](#0805devices){reference-type="ref"
    reference="0805devices"} -- that is R403, R404, R405, C401,  ↩
        C402 and
    C403

7.  Pushbutton switches SW305 to SW312 and SW314 to SW316 --  ↩
    taking care
    to put the red one at SW312 and the yellow one at SW311

8.  Pin headers K401, K402 and P401 (correct alignment of K401  ↩
    and K402
    can be assured by adding a jumper before soldering)

9.  Pin headers P101 and P201

10. Loco selection switches SW301 to SW304

11. LEDs D101, D102 and D301 to D303 with 3mm spacers to the PCB  ↩
    --
    making sure the Anode (long pin) is aligned with the square  ↩
        pad on
    all of them

12. LEDs D304 and D305 -- making sure the Anode (long pin) is  ↩
    aligned
    with the square pad on both, they can be installed on top or  ↩
        bottom
    of the PCB as desired

13. Direction switch SW313 (screwed into the PCB with an 8
    mm hex nut
    first, then attached to it's pads using the cutoffs from D301 ↩
        , D302
    and D303) and Speed potentiometer RV301 (screwed into the PCB ↩
         with a
    10 mm hex nut first)

\vspace{0.5em}
\centering
  Designator                   Package  ↩
                                                      Designation
  ----------------------------  ↩
      ------------------------------------------------------  ↩
      ----------------------------
  C102,C101                    C\_0805\_HandSoldering  ↩
                                      4u7
```

```
C105,C103, C302              C\_0805\_HandSoldering  ↩
                                    1u
C206,C205                    C\_0805\_HandSoldering  ↩
                                    22p
C401,C203, C202,C201, C207   C\_0805\_HandSoldering  ↩
                                    100n
C402,C301                    C\_0805\_HandSoldering  ↩
                                    22u
C403                         C\_0805\_HandSoldering  ↩
                                    100u
CON101                       USB\_Micro-B\_Molex-105017-0001  ↩
                           USB-MICRO-B
D101                         LED\_D3.0mm  ↩
                                               LED - red
D102                         LED\_D3.0mm  ↩
                                               LED - green
D201                         SOT-23\_Handsoldering  ↩
                                     BAR43
D301                         LED\_D3.0mm  ↩
                                               STOP - red
D302                         LED\_D3.0mm  ↩
                                               FORWARD - green
D303                         LED\_D3.0mm  ↩
                                               REVERSE - green
D303,D302, D301,D101, D102   LED Spacer  ↩
                                                3mm
D304                         LED\_D5.0mm\_Horicontal\_FLIPPED\ ↩
    _O1.27mm                  LED white
D305                         LED\_D5.0mm\_Horicontal\_O1.27mm  ↩
                              LED white
IC101                        SOT95P270X145-5N  ↩
                                          MCP73831T-2ACI\_OT
IC102                        SOT95P275X110-5N  ↩
                                          NCV8161BSN300T1G
IC201                        TQFP-32\_7x7mm\_Pitch0.8mm  ↩
                                ATMEGA328P-A
IC301                        SOT-23-6\_Handsoldering  ↩
                                   MIC2860-2PYD6
K401                         Pin\_Header\_Straight\_1x03\ ↩
    _Pitch2.54mm               UART\_ESP
K402                         Pin\_Header\_Straight\_1x03\ ↩
    _Pitch2.54mm               UART\_AVR
P1                           PCB  ↩
                                                       124mm x  ↩
    35mm x 1.6mm
P101                         Pin\_Header\_Angled\_1x02\_Pitch2 ↩
    .54mm                     BATT
```

```
P201                          Pin\_Header\_Straight\_2x03\ ↩
   _Pitch2.54mm\_SMD            ISP
P401                          Pin\_Header\_Straight\_1x02\ ↩
   _Pitch2.54mm                 ESP\_BOOTLOAD
R101,R102                     C\_0805\_HandSoldering  ↩
                                     680R
R103                          C\_0805\_HandSoldering  ↩
                                     2k2
R301                          C\_0805\_HandSoldering  ↩
                                     4k7
R304,R303, R302,R204          C\_0805\_HandSoldering  ↩
                                     220R
R305                          C\_0805\_HandSoldering  ↩
                                     15k
R405,R404, R403,R201, R104    C\_0805\_HandSoldering  ↩
                                     10k
RV301                         P160KNPD  ↩
                                                    10k lin  ↩
   P160KNPD-4FC20B10K
SW301                         OS102011MS2Q  ↩
                                                LOCO1
SW302                         OS102011MS2Q  ↩
                                                LOCO2
SW303                         OS102011MS2Q  ↩
                                                LOCO3
SW304                         OS102011MS2Q  ↩
                                                LOCO4
SW305                         SW\_SPST\_PTS645  ↩
                                            F0
SW306                         SW\_SPST\_PTS645  ↩
                                            F1
SW307                         SW\_SPST\_PTS645  ↩
                                            F2
SW308                         SW\_SPST\_PTS645  ↩
                                            F3
SW309                         SW\_SPST\_PTS645  ↩
                                            F4
SW310                         SW\_SPST\_PTS645  ↩
                                            F5
SW311                         SW\_SPST\_PTS645  ↩
                                            SHIFT
SW312                         SW\_SPST\_PTS645  ↩
                                            ESTOP
SW313                         100SP1T1B1M1QEH  ↩
                                             DIRECTION
SW314                         SW\_SPST\_PTS645  ↩
                                            F6
```

```
  SW315                        SW\_SPST\_PTS645  ↩
                                            F7
  SW316                        SW\_SPST\_PTS645  ↩
                                            F8
  U401                         ESP-12E\_SMD  ↩
                                                ESP-12E
  X201                         Crystal\_SMD\_TXC\_7M-4pin\_3.2x2 ↩
      .5mm\_HandSoldering   14.7456MHz

  : List of components for the wiFred PCB[]{label="wiFredBOM"}

To form a complete BOM, also include the parts listed in
table~[\[wiFredBOMextra\]](#wiFredBOMextra){reference-type="ref"
reference="wiFredBOMextra"} which are not soldered to the PCB but ↩
     used
in assembly later on.

\vspace{0.5em}
\centering
  Designator     Package                      Designation
  -------------- ---------------------------  ↩
      -------------------------
  B1             Battery                      Lithium battery 1700 ↩
      mAh
  H1a            Housing black                Strapubox 2090
  or H1b         Housing white                Strapubox 2090
  J1,J2          Jumper
  K1a            Potentiometer Knob silver    24mm
  or K1b         Potentiometer Knob black     24mm
  P1             PCB                          124mm x 35mm x 1.6mm
  S1,S2, S3,S4   Mounting Screws              2,9mm x 6,5mm

  : List of components for the wiFred excluding electronic parts  ↩
      to
  solder to PCB[]{label="wiFredBOMextra"}

After assembling the PCB with all the components, the holes and  ↩
    cutouts
in the enclosure most likely will have to be reworked / extended  ↩
    to
actually fit the PCB, then the PCB can be screwed into the  ↩
    enclosure
with four screws. Afterwards the battery should be connected to  ↩
    P101
making sure the orientation is correct as shown in
figure~[8](#battConnection){reference-type="ref"
reference="battConnection"} and printed on the PCB, then the  ↩
```

```
    battery
should be glued to the bottom of the enclosure with double-sided  ↩
    tape so
it does not collide with any parts on the PCB, particularly P101  ↩
    and
SW313. Finally, both the ATMega~328P and the ESP8266 will need to ↩
     be
programmed as described in the next section.

\centering
![Connection of battery to P101 -- black wire is GND, red wire is
positive[]{label="battConnection"}](images/_DSC0148){# ↩
    battConnection
width="0.8 \textwidth"}

Programming instructions
------------------------

The ATMega~328P is programmed using the regular AVR ISP  ↩
    connection on
P201. Pin 1 -- GND -- is towards the PCB edge, as shown in
figure~[9](#progAVR){reference-type="ref" reference="progAVR"}.  ↩
    An ISP
dongle with either automatic voltage selection or 3.3
    V supply voltage
should be used to avoid placing too high voltage on the ESP8266,  ↩
    which
can only support 3.3
    V power. The firmware for the ATMega~328P can be
found in the *software/avr-firmware*-subdirectory of the github
repository with both a precompiled hexfile and all source code  ↩
    including
a Makefile to recompile as needed. After writing the firmware  ↩
    file and
the eeprom file, also the fuse bits need to be set properly as  ↩
    detailed
in the *main.c*-file.

\centering
![Programming connection for ATMega~328P -- Pin 1 on purple
cable[]{label="progAVR"}](images/_DSC0146){#progAVR
width="0.8 \textwidth"}

The ESP8266 is programmed using the Arduino IDE connected via a  ↩
    serial
or USB-to-serial port to the K401 header as shown in
figure~[10](#progESP){reference-type="ref" reference="progESP"}.  ↩
    The
```

```
serial port needs to be at 3.3
    V-levels like from an FTDI232-device run
at 3.3 V. To program the ESP8266, first the ATMega~328P has to be
programmed, a battery has to be connected and reasonably charged  ↩
    and one
of the loco selection switches needs to be moved to the "enabled"
position

\centering
![Programming connection for ESP8266 -- GND on orange wire, then  ↩
    TXD of
programming cable (RXD of ESP8266), then RXD of programming cable ↩
     (TXD
of ESP8266) -- also note the jumper on
P401[]{label="progESP"}](images/_DSC0138){#progESP
width="0.8 \textwidth"}

All files in the *software/esp-firmware*-subdirectory of the  ↩
    github
repository need to be placed in a folder, then the main sketch
*arduino\_main\_sketch.ino.ino* needs to be opened with the  ↩
    Arduino IDE.
Settings for the Arduino IDE can be found inside the main file,
programming the device should work using the *Upload*-button in  ↩
    the
*Sketch*-menu.

To put the ESP8266 into programming mode, a jumper needs to be  ↩
    placed
across the P401 header before powering up the ESP8266 by enabling ↩
     one of
the loco selection switches to start the device in programming  ↩
    mode. The
red STOP LED should start flashing and the bootloader should show ↩
     some
results on the serial port and during download the LED on the  ↩
    ESP8266
module should flash as well.

After programming, two jumpers need to be placed between the K401 ↩
     and
K402 pin headers to re-enable communication between the ESP8266  ↩
    and the
ATMega~328P as shown in figure~[11](#serialJumpers){reference- ↩
    type="ref"
reference="serialJumpers"}.

\centering
```

```
![Communication jumpers for connecting the ESP8266 and the
ATMega~328P[]{label="serialJumpers"}](images/_DSC0149){# ↩
    serialJumpers
width="0.8 \textwidth"}

\clearpage
wiFred Wireless throttle configuration {#config}
======================================

Before using the device, it must be configured. At the very least ↩
    , the
General Configuration
page~[13](#throttleConfMainPage){reference-type="ref"
reference="throttleConfMainPage"} has to be submitted once to be  ↩
    saved
to non-volatile memory. If no valid configuration is detected at
startup, the device will start with a default configuration with  ↩
    no
locos enabled and no WiFi settings, so it won't be able to  ↩
    connect to
any WiFi network.

After entering any kind of text (names, numbers\...) into text  ↩
    fields,
the corresponding "Save" button has to be pressed to submit the  ↩
    changes
to the wiFred.

Entering configuration mode
---------------------------

There are two ways to enter configuration mode:

1.  Power up the throttle/select a loco when the configured WiFi  ↩
    network
    is not in range (or when there is no valid configuration --  ↩
        the
    first startup of a new throttle will fall into this category)

2.  Press SHIFT and ESTOP together when the throttle is connected

In the first case, the throttle will create a wireless network  ↩
    named
*wiFred-config* plus four hex digits taken from the MAC address  ↩
    of the
throttle WiFi interface, for example *wiFred-config0CAC* and  ↩
    announce
```

```
its presence under the name *config.local* as well as creating a ↩
    captive
portal. Any WiFi device with a web browser can connect to that ↩
    network
and open a web browser to point to *http://192.168.4.1* or
*http://config.local*. This has been tested with Mozilla Firefox ↩
    and
Opera on Linux with Avahi (a Zeroconf implementation) and Safari ↩
    on iOS
13.

In the second case, the throttle will only announce its presence ↩
    under
the name *config.local* using the Bonjour/Zeroconf-protocol. Any ↩
    device
on the same WiFi network with Bonjour/Zeroconf can use a web ↩
    browser to
access the configuration at *http://config.local*. See
section~[3.6](#configurationComputer){reference-type="ref"
reference="configurationComputer"} for an explanation what is ↩
    required
to have your device read Bonjour/Zeroconf announcements. This has ↩
     been
tested with Mozilla Firefox and Opera on Linux with Avahi (a ↩
    Zeroconf
implementation).

If the IP address or the name of the throttle during normal ↩
    operation is
known, the configuration pages can also be accessed by pointing a ↩
     web
browser to it at any time while it is connected. Note that this ↩
    is
mostly untested and therefore not recommended while the throttle ↩
    is
running locos.

\centering
![Screenshot of wiThrottle screen showing one throttle
connected[]{label="withrottleScreenshot"}](images/ ↩
    withrottle_Screenshot){#withrottleScreenshot
width="0.8 \textwidth"}

Throttle configuration
----------------------

Figure~[13](#throttleConfMainPage){reference-type="ref"
reference="throttleConfMainPage"} shows the first page you will  ↩
    see when
you point a web browser at your wiFred throttle. It is divided  ↩
    into
multiple sections explained in the following chapters.

\centering
![Screenshot of wiFred main configuration
page[]{label="throttleConfMainPage"}](images/ ↩
    wiFred_configuration_page){#throttleConfMainPage
width="0.8 \textwidth"}

### General configuration {#throttle_GeneralConf}

In the "General configuration" section there is only one  ↩
    configuration
option: The throttle name. This is a free-form identification  ↩
    string of
the throttle. It shows up in the wiThrottle window of JMRI as  ↩
    shown in
figure~[12](#withrottleScreenshot){reference-type="ref"
reference="withrottleScreenshot"} and can be used to identify the
throttle during configuration. The wiFred also announces its  ↩
    presence on
the WiFi network through Bonjour/Zeroconf using a sanitized  ↩
    version of
the name, i.e. a throttle called "Heiko Prototype 2-2" will  ↩
    announce its
presence as *heikoprototype22.local* when not in configuration  ↩
    mode.

### WiFi configuration

The "WiFi configuration" section shows a list of configured WiFi
networks. The wiFred will connect to any network in this list,  ↩
    more or
less randomly choosing one if multiple configured networks are in ↩
     range.

Existing entries can be removed by clicking on the "Remove SSID"  ↩
    button
in the line of the network that shall be removed.

New entries can be added either by manually entering the SSID and
PSK[^1] if required and clicking the "Manually add network"  ↩
    button or by
clicking on the "Scan for networks" link which takes the user to  ↩
```

```
    the
page shown in figure~[14](#throttleConfWiFiPage){reference-type=" ↩
    ref"
reference="throttleConfWiFiPage"}.

\centering
![Screenshot of wiFred "Scan for
WiFi"-page[]{label="throttleConfWiFiPage"}](images/wiFred_wifi- ↩
    scan_page){#throttleConfWiFiPage
width="0.8 \textwidth"}

This page will take a few seconds to load, since the scan for  ↩
    networks
has to be completed first. It shows all the networks found during ↩
     the
scan. Networks can be added to the list by clicking the "Add  ↩
    network"
button, after entering the PSK[^2] in the field next to it.

Note that the wiFred does not support WPS and it won't accept  ↩
    multiple
networks with the same SSID but different PSKs. More details  ↩
    regarding
the network requirements can be found in
section~[3](#serverSetup){reference-type="ref" reference=" ↩
    serverSetup"}.

The new WiFi configuration will not be activated until the wiFred ↩
     is
restarted, either through a power-cycle or by clicking on the " ↩
    Restart
wiFred to enable new WiFi-settings" link on the configuration  ↩
    page.

### Loco server configuration

Following the WiFi configuration, the section "Loco server
configuration" allows configuring the wiThrottle server to which  ↩
    the
wiFred shall connect. The default setting -- automatically detect ↩
     server
-- works well if there is only one wiThrottle server on the  ↩
    network. It
will connect to any server announcing its presence on port 12090  ↩
    through
Zeroconf/Bonjour, the result of the Zeroconf/Bonjour-search will  ↩
    be
```

```
shown here when the wiFred has automatically discovered a server.

### Loco configuration {#throttle_LocoConf}

Following the "Loco server configuration", there are four  ↩
    identical
sections assigned to the four different locomotives which can be
controlled with this throttle. Each section consists of the  ↩
    following
settings:

DCC address: Can be a short address between 1 and 127 (also used  ↩
    for
consists) or a long address between 0 and 10239. Note: Short  ↩
    addresses
between 1 and 127 are not the same as long addresses between 1  ↩
    and 127.
If this is set to -1, the corresponding loco is disabled.

Long address?: Checkbox to change the behaviour of the DCC  ↩
    address input
field described above.

Reverse?: If checked, the corresponding loco will invert it’s  ↩
    travel
direction. Mainly intended for back-to-back consists without  ↩
    decoder
reconfiguration.

Function mapping: Link to the function mapping subpage for the
corresponding loco, as described in
section~[2.2.5](#throttle_FunctionConf){reference-type="ref"
reference="throttle_FunctionConf"}. Clicking this link will lose  ↩
    all
information entered on the current page and take the web browser  ↩
    to a
different subpage.

**Reminder: Changes are saved using the "Save loco config" button ↩
     which
may look different in different web browsers (firefox shown).**

### DCC function configuration {#throttle_FunctionConf}

By default, if a function key is pressed, the throttle will send  ↩
    the
appropriate commands to every loco under control. Under certain
```

```
circumstances, this may not be desired -- the obvious example  ↩
    being a
loco in the middle of a multi-unit consist, which should not have ↩
     lights
or ditchlights. So this page -- shown in
figure~[15](#throttleConfigFunctionPage){reference-type="ref"
reference="throttleConfigFunctionPage"} -- offers the option to  ↩
    chose
between three different settings for every function on each of  ↩
    the four
locomotives (one page per locomotive):

\centering
![Screenshot of wiFred function handling config
page[]{label="throttleConfigFunctionPage"}](images/ ↩
    wiFred_function_page){#throttleConfigFunctionPage
width="0.8 \textwidth"}

Always Off: When the loco is enabled by moving the selection  ↩
    switch to
the "selected" position, the current status of the function is  ↩
    queried.
If the function is on, a function key press will be simulated to  ↩
    turn it
off. No other function key events will be sent to this loco for  ↩
    this
function.

Throttle controlled: When the first loco is enabled by moving the
selection switch to the "selected" position, the current status  ↩
    of the
function is queried and saved. When selecting the next loco, the  ↩
    status
is queried. If it does not match the first loco, the function  ↩
    status is
changed by simulating a function key press. Afterwards, key  ↩
    presses are
handed through to the loco.

Always On: Similar to the "Always Off" setting, but the throttle  ↩
    will
attempt to enable the function when the locomotive is selected  ↩
    and
ignore any further function key presses. This will probably not  ↩
    work
with so-called momentary functions that are only active as long  ↩
    as the
```

```
function key is pressed.

**Reminder: Changes are saved using the "Save function  ↩
    configuration"
button which may look different in different web browsers ( ↩
    firefox
shown).**

### wiFred status

The "wiFred status" section shows the current battery voltage, as
measured by the wiFred. This is updated on reloading the page,  ↩
    not
continuosly.

### wiFred system

The "wiFred system" section consists of two links:

-   Reset wiFred to factory defaults -- which leads to a  ↩
    confirmation
    page shown in~[16](#throttleConfResetPage){reference-type=" ↩
        ref"
    reference="throttleConfResetPage"} to reset all configuration ↩
         data
    to factory defaults as on a new wiFred.

-   Update wiFred firmware -- which leads to a firmware update  ↩
    page
    shown in~[17](#throttleConfUpdatePage){reference-type="ref"
    reference="throttleConfUpdatePage"} to update the wiFred  ↩
        firmware of
    the ESP8266. Find the .bin-file from the arduino build folder ↩
        , click
    on "Choose file", navigate to the .bin-file and finally  ↩
        initiate the
    update with a click on "Update" -- which will take a while.

\centering
![Screenshot of wiFred configuration reset
page[]{label="throttleConfResetPage"}](images/wiFred_reset_page) ↩
    {#throttleConfResetPage
width="0.8 \textwidth"}

\centering
![Screenshot of wiFred firmware update
page[]{label="throttleConfUpdatePage"}](images/wiFred_update_page ↩
    ){#throttleConfUpdatePage
width="0.8 \textwidth"}

\clearpage
Options for server setup {#serverSetup}
========================

Figure~[18](#runningTrains){reference-type="ref"
reference="runningTrains"} shows the connections between the  ↩
    devices
required to run trains using the wiFred.

\centering
![Overview of devices required to run trains with the
wiFred[]{label="runningTrains"}](images/runningTrains){# ↩
    runningTrains
width="0.99 \textwidth"}

The symbols in figure~[18](#runningTrains){reference-type="ref"
reference="runningTrains"} symbolize the following parts:

1.  An IEEE 802.11b/g/n 2.4
    GHz WiFi access point described in detail in
    section~[3.3](#serverWiFi){reference-type="ref"
    reference="serverWiFi"} [\[indexWiFi\]]{#indexWiFi
    label="indexWiFi"}

2.  A PC or laptop computer with Windows, Linux or MacOS to run  ↩
    the JMRI
    server described in detail in
    section~[3.4](#serverJMRI){reference-type="ref"
    reference="serverJMRI"} [\[indexJMRIserver\]]{# ↩
        indexJMRIserver
    label="indexJMRIserver"}

3.  A way to connect the JMRI server to the model railroading  ↩
    layout
    described in detail in
    section~[3.5](#serverLayoutConn){reference-type="ref"
    reference="serverLayoutConn"} [\[indexLocoBuffer\]]{# ↩
        indexLocoBuffer
    label="indexLocoBuffer"}

4.  A device with a web browser connected to the same network as  ↩
    the
    wiFred to configure it -- can be the same physical device
    as~[\[indexJMRIserver\]](#indexJMRIserver){reference-type=" ↩
        ref"
    reference="indexJMRIserver"} if requirements in
    section~[3.6](#configurationComputer){reference-type="ref"
    reference="configurationComputer"} are met
    [\[indexConfigurationComputer\]]{#indexConfigurationComputer
    label="indexConfigurationComputer"}

Multiple options for every step or combining these steps are  ↩
    described
in the following sections.

Basically, if a layout is set up to run trains with a smartphone  ↩
    running
wiThrottle or EngineDriver, a wiFred should work with no changes  ↩
    to the
layout configuration.

If a layout is set up in a way that trains can be run from a JMRI ↩
     screen
throttle on a computer, only a WiFi connection to the JMRI  ↩
    computer
needs to be added.

Out-of-the-box server-side options
----------------------------------

A pretty much out-of-the-box solution is provided by Steve Todd
at~[@raspiImage] which auto-detects multiple options to interface ↩
     to a
DCC layout and has been tested in the JMRI 4.16 version to work  ↩
    with the
wiFred, connecting to a Z21 black through both an RRCircuits
LocoBuffer~USB and a Digitrax~PR3 via Loconet.

Although untested so far, adding a Digitrax~LNWI~[@digitrax] to a
Digitrax system or an MRC~Prodigy~WiFi~[@mrc] to an MRC system  ↩
    should
allow the wiFred to run locos out-of-the-box as well.

Step by step instructions for a Windows computer
------------------------------------------------

Tested on Windows 7 64Bit

Requirements: WiFi 2.4GHz

Installation:
```

```
1.  Install HostedNetworkStarter from
    https://www.nirsoft.net/utils/wifi\_hotspot\_starter.html

2.  Install DHCP server from https://www.dhcpserver.de/cms/ ↩
    download/ --
    download and extract all the files from the zip file to  ↩
        somewhere on
    your harddrive, for example C:\\DHCPServer

3.  Install a JDK, version 8 and 11 have been tested. For example ↩
    ,
    https://adoptopenjdk.net/releases.html Version OpenJDK 11 ( ↩
        LTS), JVM
    HotSpot. Choose the 64bit version for most modern hardware,  ↩
        32bit
    only if you are running a 32bit operating system. Easiest  ↩
        option:
    MSI file, download and install.

4.  Install JMRI from https://www.jmri.org -- versions tested to  ↩
    run
    with the wiFred include 4.14, 4.16, 4.18 and 4.20. Most  ↩
        recent
    production version recommended.

Configuration:

1.  Start HostedNetworkStarter from the start menu, enter a  ↩
    Network Name
    and Network Key, then hit the Start button. Note the "Hosted  ↩
        Network
    Connection Name" for the next step

2.  Start the DHCP server wizard from C:\\DHCPServer\\dhcpwiz.exe ↩
    ,
    select the network with the name that's the same as the " ↩
        Hosted
    Network Connection Name" from the step before, hit "Next" a  ↩
        few
    times (deselecting all additional supported protocols), Write ↩
         INI
    file, Start Service and Configure Firewall Exceptions

3.  Start JMRI using the DecoderPro icon on the desktop, setup  ↩
    your
    layout connection, test if you can run a loco with a JMRI  ↩
        throttle
```

```
4.  Within JMRI, start the WiThrottle Server from the Actions  ↩
    menu. If a
    firewall popup comes up, allow all.

5.  Within JMRI, edit the Preferences from the Edit menu, choose
    WiThrottle on the left pane, click the "Start automatically  ↩
        with
    application" checkbox. All the Allowed Controls can be  ↩
        disabled.

Running:

1.  Start HostedNetworkStarter from the start menu, enter a  ↩
    Network Name
    and Network Key, then hit the Start button.

2.  Start JMRI using the DecoderPro icon on the desktop

WiFi access point requirements {#serverWiFi}
------------------------------

IEEE802.11bg 2.4GHz DHCP server comm between clients

Linux: hostapd (tested: netbook, Raspberry Pi 3 in a PiTop)  ↩
    Windows:
link to \... Hardware.

JMRI server requirements {#serverJMRI}
------------------------

Any PC.

Layout connection options {#serverLayoutConn}
-------------------------

Loconet: LocoBufferUSB Digitrax PR3 / PR4

Tested: Intellibox, Z21 black, DCS 51 Zephyr xtra

Should work: Anything JMRI can control trains on, even SPROG as  ↩
    command
station plus boosters\...

Computer or smartphone to configure wiFred {# ↩
    configurationComputer}
------------------------------------------
```

```
Webbrowser, Zeroconf. Avahi. Bonjour (iTunes?). MacOS out of the box?
iOS? Android?

For initial configuration of the wiFred, most of the devices mentioned
above can be omitted. As shown in
figure~[19](#confWifred){reference-type="ref" reference="confWifred"},
only a WiFi capable device with a web browser is required.

\centering
![For initial configuration, the requirements are very
small[]{label="confWifred"}](images/configuringWifred){#confWifred
width="0.5 \textwidth"}

\clearpage
wiFred Wireless throttle prototype {#oldThrottle}
==================================

Quickstart Guide
```

Follow these steps for a new throttle (see later chapters for more explanation or if you run into trouble)

-3. Use PCB to determine positions of holes and cutouts in housing

-2. Make said cutouts and glue little pieces of 3mm thick plastic or wood underneath PCB screw holes

-1. Solder components to PCB

1. Flash firmware to ESP8266 and to ATMega 328P
2. Test fit PCB into housing, removing plastic parts of housing as required
3. Fit PCB into housing, insert three screws to fix PCB to housing
4. Make sure communication jumpers are set correctly, close housing and fix back cover with two screws
5. Add throttle knob
6. Insert batteries
7. Using any WiFi client (laptop, smartphone, tablet...), find and connect to network **wiFred-configXXXX**
8. Using any web browser, navigate to **http://192.168.4.1**
9. Enter your WiFi configuration (and a throttle ID if you like — highly recommended to easier tell them apart) **and hit the Submit-Button**
10. Click on `Loco configuration subpage`

11. Enter your wiThrottle server settings
12. For every loco you want to control with this throttle, enter the appropriate details below
13. Finish by **hitting the Submit-Button**
14. Configure function settings for each loco on the respective sub pages if required
15. Restart the throttle by navigating back to the main configuration page and clicking on `Restart system to enable new WiFi settings`

Your throttle should now be ready to use and connect to your wiThrottle server on startup. Refer to the chapters below if it does not or contact the author of this document.

Usage

```
\centering
![Controls and features of the wiFred-throttle -- prototype
version[]{label="oldThrottleControls"}](images/throttle_Front " ↩
    fig:"){#oldThrottleControls
height="100mm"} ![Controls and features of the wiFred-throttle --
prototype
version[]{label="oldThrottleControls"}](images/throttle_Back "fig ↩
    :"){#oldThrottleControls
height="100mm"} ![Controls and features of the wiFred-throttle --
prototype
version[]{label="oldThrottleControls"}](images/ ↩
    throttle_Back_openBattery "fig:"){#oldThrottleControls
height="100mm"}

Figure~[22](#oldThrottleControls){reference-type="ref"
reference="oldThrottleControls"} shows the controls of the  ↩
    wireless
throttle. They consist of the following:

-   Four loco selection switches (loco 1 on the left, loco 4 on  ↩
    the
    right, move towards speed potentiometer to enable)

-   Speed potentiometer (Counter-clockwise endstop: Stop,  ↩
    clockwise
    endstop: Full speed)

-   Direction switch -- move right for forward movement, left for
    reverse movement

-   Black function keys F0 to F4

-   Two yellow shift keys to trigger F5-F8 (SHIFT1, lower key),  ↩
    F9-F12
    (SHIFT2, upper key) and F13-F16 (both shift keys)
```

```
-   Red emergency stop key

-   Two green direction indicator LEDs

-   One red status LED

-   Battery compartment (on the rear) for two AA cells, 1.2
    V to 1.5 V
    nominal voltage

As soon as a pair of batteries is inserted into the battery  ↩
    compartment
as the symbols inside the battery compartment show, the throttle  ↩
    will
boot up and try to connect to a wireless network. The throttle  ↩
    will not
be damaged if batteries are inserted wrongly, but it will not  ↩
    work
either. Use NiMH- or primary AA cells with 1.2 V to 1.5 V nominal
voltage, low self discharge NiMH cells like Eneloop~or similar  ↩
    are
recommended. Do not insert 3 V or 3.6
    V AA size lithium batteries as
this may damage the throttle.

If no connection to the network configured into the device can be
established within 60 seconds, the throttle will create it’s own
wireless network named *wiFred-config* plus four hex digits taken ↩
     from
the MAC address of the throttle WiFi interface, for example
*wiFred-config0CAC*, to enable configuration as described in
section~[2](#config){reference-type="ref" reference="config"}.

Four different locos with long DCC addresses can be assigned to  ↩
    the four
loco selection switches. Commands derived from the speed  ↩
    potentiometer,
the direction switch and the function keys will be transmitted to ↩
     all
selected locos (near) simultaneously, with a certain translation  ↩
    table
enabling some locos to go backwards when others go forwards and  ↩
    also
limiting function keys to some of the four locos only -- this is
described in more detail in
sections~[2.2.4](#throttle_LocoConf){reference-type="ref"
reference="throttle_LocoConf"}
```

```
and~[2.2.5](#throttle_FunctionConf){reference-type="ref"
reference="throttle_FunctionConf"}.

Pushing the red emergency stop key will cause the throttle to  ↩
    send an
emergency stop signal to all four locos attached. After an  ↩
    emergency
stop, turn the speed potentiometer to zero to re-enable control  ↩
    of the
locos.

Pushing the red emergency stop key while holding down either of  ↩
    the
shift keys will place the device into configuration mode (as well ↩
     as
issueing an emergency stop to all attached locos). See
section~[2](#config){reference-type="ref" reference="config"} for ↩
     more
details on how to access the throttle to do the configuration.

Any change in the loco selection switches will cause the throttle ↩
     to
send an emergency stop command to all attached locos. This makes  ↩
    sure
that any loco that is deselected will stop on the layout and  ↩
    avoids
newly selected locos suddenly taking off at speed. The same is  ↩
    true for
a change in the direction switch, to avoid high-speed reverse  ↩
    maneuvers.
Turn the speed potentiometer to zero to re-enable control of the  ↩
    locos.

When all four loco selection switches are set to the disabled  ↩
    state, the
throttle will send an emergency stop command to all four locos  ↩
    attached
and -- after a wait time of 30 seconds -- it will disconnect from ↩
     the
network and go into low power mode. To reconnect, re-enable any  ↩
    loco
selection switch.

The same happens when the batteries are empty, but the throttle  ↩
    will not
reactivate before changing the batteries. Expected runtime with a ↩
     pair
```

```
of 2500 mAh-NiMH-batteries is around 8-10 hours of full time  ↩
    operations,
more if the throttle is placed in low power mode when the locos  ↩
    are not
running.

During startup and operation, the LEDs will show the patterns  ↩
    explained
in table~[\[ledTable\]](#ledTable){reference-type="ref"
reference="ledTable"}.

Hardware description
--------------------

The wiFred hardware is centered around an ESP8266 for the WiFi
connection. The ESP8266 also reads the loco selection switches  ↩
    and the
battery voltage and communicates through it's serial port with an
ATMega~328P microcontroller which controls the LEDs, reads the  ↩
    speed
potentiometer, direction switch and pushbutton switches for  ↩
    functions
and emergency stop. The communication goes through a 2x3 pin  ↩
    header
which enables the user to connect a programming cable to the same ↩
     serial
port if removing the jumpers.

The wiFred is powered by two AA size battery cells connected to a
step-up converter creating 3.3 V for the entire device.

The schematic is split into several pages and can be found in
figures~[23](#oldSchematicPage1){reference-type="ref"
reference="oldSchematicPage1"}
to~[26](#oldSchematicPage4){reference-type="ref"
reference="oldSchematicPage4"}. It has been created with kicad  ↩
    and is
available on the github repository at
*http://github.com/newHeiko/wiFred* along with the PCB design.

\centering
![Master schematic sheet with batteries and power
supply[]{label="oldSchematicPage1"}](images/old_wfred_rev2){# ↩
    oldSchematicPage1
width="\textwidth"}

\centering
```

```
![Schematic sheet including ESP8266 for WiFi connection with  ↩
    bootloader
enabling jumper and connection to programming
cable[]{label="oldSchematicPage2"}](images/old_wfred-wifi- ↩
    Wifi_connection){#oldSchematicPage2
width="\textwidth"}

\centering
![Schematic sheet including ATMega 328P along with crystal and in ↩
     system
programming
header[]{label="oldSchematicPage3"}](images/old_wfred- ↩
    controller_rev2-Controller){#oldSchematicPage3
width="\textwidth"}

\centering
![Schematic sheet including pushbutton switches, loco selection
switches, direction switch and speed
potentiometer[]{label="oldSchematicPage4"}](images/ ↩
    old_User_interface_rev2-User_Interface){#oldSchematicPage4
width="\textwidth"}

Hints for building the wiFred
-----------------------------

The PCB has holes in the center of the pushbutton switch  ↩
    footprints and
LED footprints to enable transferring their positions to a  ↩
    StrapuBox
housing with a sharp needle or similar, and the position of the  ↩
    loco
selection switches can also be transferred to the housing by  ↩
    marking it
through the non-copper holes at their ends.
Figure~[28](#oldTransferHoles){reference-type="ref"
reference="oldTransferHoles"} shows the process and it's results. ↩
     Holes
for the pushbutton switches should be drilled at 3.5
    mm diameter and
countersunk from the inside. Holes for the LEDs should be drilled ↩
     at
3 mm diameter and holes for the speed potentiometer and direction  ↩
    switch
at 6.5 mm or 7
    mm diameter and countersunk. The cutouts for the loco
selection switches are best created when the PCB is assembled and
carefully cut out with a sharp hobby knife and a file until they  ↩
    fit.
```

```
\centering
![Using the PCB to transfer the positions of the holes to the
housing[]{label="oldTransferHoles"}](images/_DSC8652 "fig:"){# ↩
    oldTransferHoles
width="0.49 \textwidth"} ![Using the PCB to transfer the  ↩
    positions of
the holes to the
housing[]{label="oldTransferHoles"}](images/_DSC8653 "fig:"){# ↩
    oldTransferHoles
width="0.49 \textwidth"}

The remaining assembly is a basic exercise in installing all the
components to the PCB, listed in
table~[\[oldWiFredBOM\]](#oldWiFredBOM){reference-type="ref"
reference="oldWiFredBOM"}. From assembling the prototypes, the  ↩
    suggested
order of installing the components is as follows:

\vspace{0.5em}
\centering
  Designator                        Package  ↩
                                                     Designation
  ---------------------------------  ↩
      ------------------------------------------------------  ↩
      ----------------------------
  B101                              KEYSTONE1013  ↩
                                                BATT\_HOLDER
  C206,C205                         C\_0805\_HandSoldering  ↩
                                      22p
  C301,C105, C104,C102, C101        C\_0805\_HandSoldering  ↩
                                      22u/25V
  C401,C204, C203,C202, C201,C103   C\_0805\_HandSoldering  ↩
                                      100n
  C402                              C\_0805\_HandSoldering  ↩
                                      22u
  D301                              LED\_D3.0mm  ↩
                                                 STOP - red
  D302                              LED\_D3.0mm  ↩
                                                 FORWARD - green
  D303                              LED\_D3.0mm  ↩
                                                 REVERSE - green
  IC201                             TQFP-32\_7x7mm\_Pitch0.8mm  ↩
                                  ATMEGA328P-A
  K401                              Pin\_Header\_Straight\_1x03\ ↩
      _Pitch2.54mm                 UART\_ESP
  K402                              Pin\_Header\_Straight\_1x03\ ↩
```

```
    _Pitch2.54mm                UART\_AVR
L101                               L\_2424\_HandSoldering  ↩
                                    22u
P201                               Pin\_Header\_Straight\_2x03\ ↩
    _Pitch2.54mm\_SMD           ISP
P401                               Pin\_Header\_Straight\_1x02\ ↩
    _Pitch2.54mm                ESP\_BOOTLOAD
R301                               C\_0805\_HandSoldering  ↩
                                    4k7
R304,R303, R302                    C\_0805\_HandSoldering  ↩
                                    470R
R401                               C\_0805\_HandSoldering  ↩
                                    100k
R402                               C\_0805\_HandSoldering  ↩
                                    47k
R405,R404, R403,R201               C\_0805\_HandSoldering  ↩
                                    10k
RV301                              P160KNPD  ↩
                                                   10k lin  ↩
    P160KNPD-4FC20B10K
SW301                              OS102011MS2Q  ↩
                                               LOCO1
SW302                              OS102011MS2Q  ↩
                                               LOCO2
SW303                              OS102011MS2Q  ↩
                                               LOCO3
SW304                              OS102011MS2Q  ↩
                                               LOCO4
SW305                              KSC621G  ↩
                                                    F0
SW306                              KSC621G  ↩
                                                    F1
SW307                              KSC621G  ↩
                                                    F2
SW308                              KSC621G  ↩
                                                    F3
SW309                              KSC621G  ↩
                                                    F4
SW310                              KSC621G  ↩
                                                    SHIFT2
SW311                              KSC621G  ↩
                                                    SHIFT
SW312                              KSC621G  ↩
                                                    ESTOP
SW313                              100SP1T1B1M1QEH  ↩
                                            DIRECTION
U101                               TSSOP-8\_4.4x3mm\_Pitch0.65mm ↩
```

```
                                L6920D
  U401                              ESP-12E\_SMD  ↩
                                                 ESP-12E
  X201                              Crystal\_SMD\_TXC\_7M-4pin\_3 ↩
      .2x2.5mm\_HandSoldering   14.7456MHz
                                    Housing StrapuBox 6090
                                    Two Jumpers, 2.54mm
                                    Potentiometer Knob, 21 mm
                                    Three fastening screws, 2.9
                                        mm dia x 6.5 mm
```

```
  : List of components for the wiFred[]{label="oldWiFredBOM"}

1.  IC201 and U101 (note: Rotate PCB so Designator is right side  ↩
    up,
    then Pin 1 is on top left)

2.  X201

3.  Capacitors and Resistors in 0805 size (only those on the same ↩
     side
    as the items before) [\[old0805devices\]]{#old0805devices
    label="old0805devices"}

4.  U401

5.  LEDs D301 to D303

6.  Pushbutton switches SW305 to SW312

7.  Loco selection switches SW301 to SW304

8.  L101

9.  Capacitors and Resistors not installed in
    step~[\[old0805devices\]](#old0805devices){reference-type=" ↩
        ref"
    reference="old0805devices"}

10. Pin header P201

11. Pin headers K401, K402 and P401 (correct alignment of K401  ↩
    and K402
    can be assured by adding a jumper before soldering)

12. Direction switch SW313 (screwed into the PCB with an 8
    mm hex nut
    first, then attached to it’s pads using the cutoffs from D301 ↩
```

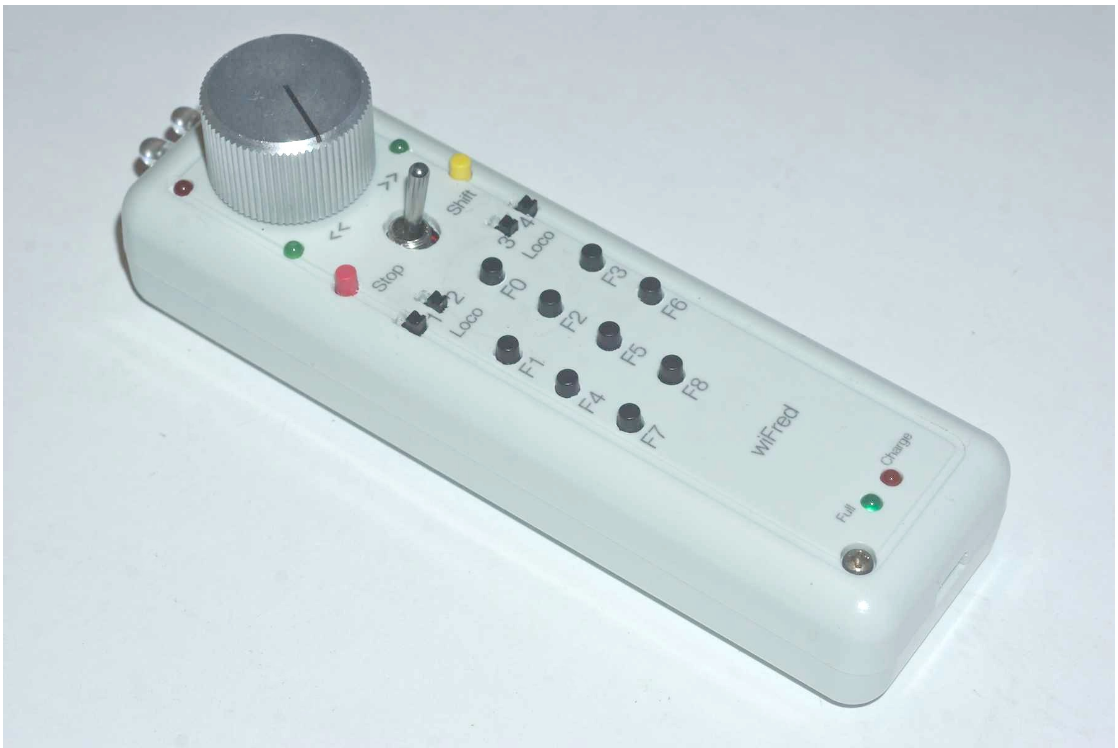
```
      , D302
   and D303) and Speed potentiometer RV301 (screwed into the PCB ↩
        with a
   10 mm hex nut first and slightly shortening the pins before
   soldering)

13. Battery holder B101

After assembling the PCB with all the components and drilling and
cutting the holes and cutouts into the housing, there are few  ↩
    steps
left. First, a few protrusions inside the housing need to be  ↩
    removed so
the PCB fits properly.
Figure~[30](#breakProtrusions){reference-type="ref"
reference="breakProtrusions"} shows how they can be removed  ↩
    easily,
remains may be cut off with a hobby knife. Second, new PCB  ↩
    mounting pads
need to be installed as shown in
figure~[31](#mountingPads){reference-type="ref"
reference="mountingPads"}. For the prototype, Forex PVC foam was  ↩
    used,
cut with a pair of scissors and glued to the housing with  ↩
    superglue,
making sure not to be in the way of any components on the PCB,  ↩
    but any
kind of easily worked upon material with a thickness of 3
    mm can be
used, as long as it will take self-driving screws (prototype uses ↩
     2.9
    mm
by 6.5 mm DIN~7981 screws). Third, the two shift keys need yellow  ↩
    paint
on the top and the emergency stop key needs red paint -- either  ↩
    any kind
of paint or a paint marker like Edding~751 will do. Finally, both ↩
     the
ESP8266 and the ATMega~328P will need to be programmed as  ↩
    described in
the next section.

\centering
![Removing protrusions inside the housing so the PCB
fits[]{label="breakProtrusions"}](images/_DSC8654 "fig:"){# ↩
    breakProtrusions
width="0.49 \textwidth"} ![Removing protrusions inside the  ↩
    housing so
```

```
the PCB
fits[]{label="breakProtrusions"}](images/_DSC8655 "fig:"){# ↩
    breakProtrusions
width="0.49 \textwidth"}

\centering
![New PCB mounting pads made from 3 mm thick Forex
PVC[]{label="mountingPads"}](images/_DSC8658){#mountingPads
width="0.8 \textwidth"}

Programming instructions
------------------------

The ESP8266 is programmed using the Arduino IDE connected via a  ↩
    serial
or USB-to-serial port to the K401 header as shown in
figure~[32](#oldProgESP){reference-type="ref" reference=" ↩
    oldProgESP"}.
The serial port needs to be at 3.3
    V-levels like from an FTDI232-device
run at 3.3 V.

\centering
![Programming connection for ESP8266 -- GND on orange wire, then  ↩
    TXD of
programming cable (RXD of ESP8266), then RXD of programming cable ↩
     (TXD
of ESP8266)[]{label="oldProgESP"}](images/_DSC8637){#oldProgESP
width="0.8 \textwidth"}

\centering
![Programming connection for ATMega~328P -- Pin 1 on purple
cable[]{label="oldProgAVR"}](images/_DSC8638){#oldProgAVR
width="0.8 \textwidth"}

All files in the *software/esp-firmware*-subdirectory of the  ↩
    github
repository need to be placed in a folder, then the main sketch
*arduino\_main\_sketch.ino.ino* needs to be opened with the  ↩
    Arduino IDE.
Settings for the Arduino IDE can be found inside the main file,
programming the device should work using the *Upload*-button in  ↩
    the
*Sketch*-menu.

To put the ESP8266 into programming mode, a jumper needs to be  ↩
    placed
across the P401 header before inserting batteries to start the  ↩
```

```
    device in
programming mode. The bootloader should show some results on the  ↩
    serial
port and during download the LED on the ESP8266 module should  ↩
    flash.

The ATMega~328P is programmed using the regular AVR ISP  ↩
    connection on
P201. Pin 1 -- GND -- is towards the PCB edge, as shown in
figure~[33](#oldProgAVR){reference-type="ref" reference=" ↩
    oldProgAVR"}.
An ISP dongle with either automatic voltage selection or 3.3
    V supply
voltage should be used to avoid placing too high voltage on the  ↩
    ESP8266,
which can only support 3.3
    V power. The firmware for the ATMega~328P can
be found in the *software/avr-firmware*-subdirectory of the  ↩
    github
repository with both a precompiled hexfile and all source code  ↩
    including
a Makefile to recompile as needed. After writing the firmware  ↩
    file, also
the fuse bits need to be set properly as detailed in the *main.c ↩
    *-file.

After programming, two jumpers need to be placed between the K401 ↩
     and
K402 pin headers to re-enable communication between the ESP8266  ↩
    and the
ATMega~328P.

\clearpage
Background for wiFred development {#background}
```

As of the writing of this document, JMRI [@jmri] has a long track record of offering a server for using smartphones as wireless model railroad throttles, along with apps like withrottle [@withrottleApp][^3] and EngineDriver [@EngineDriver]. This server will enable WiFi throttles to control locos any model railroading layout to which JMRI can build a connection [@jmrihardwaresupport]. In addition, Digitrax [@digitrax] and MRC [@mrc] offer specific hardware solutions to enable the connection of the abovementioned smartphone apps to their DCC systems through a WiFi network.

The Fremo [@fremo] is a European modular model railroading club whose unique requirements on it's DCC throttles led to the creation of the throttles FRED and FREDI [@fred] — a series of LocoNet-throttles which started their life as hobbyist projects with large numbers in circulation but were also commercially available from Uhlenbrock [@uhlenbrock].



1 Specification wishlist

In modular railroading events, particularly of the Fremo-americaN-group [@fremo], multiple people have evaluated the smartphone throttle solutions and found them lacking a nice, haptical feedback. But the idea of wireless control without locking into a specific vendor and their necessarily expensive equipment found great approval. So a wishlist was compiled to define the requirements for a wireless throttle:

- Same form factor as the FRED [@fred] with similar controls
- Option to control at least two, better four locomotives for double/triple traction (similar to the double FRED)
- Battery runtime of at least six hours
- Exchangeable batteries, so when the battery runs down, they can be quickly exchanged for a charged set or cheap primary cells
- Easy configuration, but not too easy to prevent operators from accidentally selecting other locomotives
- As little change to the existing Fremo Loconet network as possible

- Use of withrottle protocol, so the server side of the communication can be assumed to work and does not have to be developed as well

2 Development history

The first prototype versions of the wiFred were built to run from two AA cells, either dry batteries or rechargeable NiMH cells. As described in section [4](#oldThrottle){reference-type="ref" reference="oldThrottle"}, this led to some special adaptations of the housing to fit all components. Even then, experience with the prototypes showed the battery compartment cover did not really fit and easily broke when trying to open and close the battery compartment. So the next versions were built around an integrated lithium battery, losing the ability to exchange empty batteries, but with increased runtime and proper fit into the housing. Recharging of the second generation is done through a Micro USB connector, so a powerbank can extend the runtime of the device when the internal battery is exhausted. Also, the loco selection switches act as more of a power switch than they did with the first prototypes, reducing power consumption to a negligible amount when all locos are deselected.

3 Wireless clock

During the development of this wiFred another topic came up in the americaN group of the Fremo, namely wireless clocks with adjustable clock rate for Timetable & Trainorder operations. This led to the spinoff of the wiClock project[@wiClock].

\clearpage 99

[^1]: Pre-Shared Key, often just called password

[^2]: Pre-Shared Key, often just called password

[^3]: withrottle is also the name JMRI uses for the protocol and the server.