ExpressLRS
Manufacturer Design Guidelines

Version 1.6

Release Date 17/3/2023

Version History

1.1	
1.2	Add note about using ESP32-PICO-D4 with backpack
1.3	Add Rx reference pics, ESP32 option to RX, updated supported packet rates on 2.4G
1.4	Add Flashing in the factory
1.5	Add diversity/gemini information Add PWM14/16 information Update XO requirement and Add TCXO recommendation Expand the flashing section Clarify prototype/production sample policy Add TX/RX reference designs (at least pinout definitions) Add TX/RX approval checklist
1.6	Remove equal trace length requirement for True Diversity receiver

Important terms & conditions:

Thank you for considering the development of ExpressLRS hardware.
ExpressLRS (ELRS) is free to use (no license cost) for both private and commercial use, however there is **one condition that we ask from manufacturers** who are looking to add a new supported target to the repository:
For any new target (i.e. adding *"ManufacturerXYZ 2.4G receiver"* to the Configurator), samples must be provided to the ELRS development team. These samples can be provided during the hardware development stage, or at the time when the final product is ready, however it is **strongly advisable** to wait until the ELRS dev team have reviewed and approved the hardware (at which point we will add the new target name to the configurator) before going to market or accepting pre-orders.
This provides the ELRS dev team with a chance to ensure the hardware and firmware are 100% compatible, and it also ensures that if any after-sales support is required (i.e. helping users with firmware updates etc.), the devs have access to the same hardware, and can provide assistance. In addition, if the hardware is significantly updated from the initial sample, it's also advised to send a few more samples to the devs.

Engaging with the ELRS development team

What does a typical engagement look like between a manufacturer and the dev team that results in success?

1. The manufacturer contacts one of the devs, and expresses interest in developing hardware. If you are reading this, then this has likely already taken place.
2. The dev team will create a private discord server with key members of the ELRS dev team, and the manufacturer representative. Feel free to add your engineers to this chat.
3. Discuss the design - the ELRS devs would love to hear about what you have in mind for your new ELRS target. We ideally would like to see work-in-progress schematics / PCB renders, especially before prototypes are manufactured, and can help to provide feedback and review. Everything that is shared / discussed in the private chat is taken as "commercial in confidence" and will not be discussed externally to the dev team. Keep in mind that the devs work for free, and have day jobs / families, so sometimes it may take a little time for us to fully check over your work.
4. When prototype samples (~3 units are sufficient) are ready, these should be sent to the selected ELRS dev team for review and testing. Once we are happy with the functionality of the hardware, we will add the specific target for your hardware to the ELRS Configurator. This can be coordinated so that it is only publicly visible at the time of product release if desired.
5. If any changes to hardware are requested by the devs (i.e. as a result of problems being found), the manufacturer is expected to resolve the problems, and re-send the fixed samples to the dev team, before the target is added to the Configurator.
6. Once the hardware is approved, and the target has been merged, it will be available in the next release of ExpressLRS, unless otherwise negotiated.

7. Finally, the manufacturer sends the production unit (=the same unit which your customers will get) samples to the requested ELRS devs (~10 units). The samples will be reserved in the devs' hands for future technical support.

Transmitter Design Guidelines

Supported TX MCUs

- **ESP32 (preferred)**
- STM32F103C8
- STM32F303

Supported RF chips

- Semtech SX1276
- Semtech SX1280 / SX1281

Crystal oscillator

- For SX1280/SX1281, use a decent quality 52 MHz XO that has the 10 ppm tolerance and 10 pF load capacitance specification exactly.
  - **DO NOT ADD** external load capacitors. SX1280 has 10 pF internal load.
  - https://www.expresslrs.org/3.0/hardware/crystal-frequency-error/
- For SX1276, 32MHz XO with 10 ppm tolerance spec is acceptable.
  - Unlike SX1280, external load capacitors (matched with the XO) are required as usual.
- For both 2.4G and 900M, temperature-compensated crystal oscillators (TCXO) are ***strongly*** recommended whenever a product is expected to generate a fair amount of heat 🔥 (e.g.RF amps, VTX, etc).
- For both 2.4G and 900M, a ***shared*** TCXO is ***mandatory*** for Gemini transmitters and True Diversity receivers. See section *True Diversity hardware design issue* for more information.

Supported RF power levels

- 10mw
- 25mw
- 50mw
- 100mw
- 250mw
- 500mw*
- 1000mw*
- 2000mw*

*** Active (fan) cooling is required for power levels above 250mW**

We suggest if you develop hardware it supports at least 250mW output power.

Supported additional features

- OLED (SSD1306 128*64 or 128*32) and TFT (ST7735 80*160) screen support
- 5-way button support for menu navigation
- RGB LED support
- ESP8285 “backpack” support is **highly recommended** (for interfacing wirelessly with 3rd party devices) see backpack wiring section
- If ESP32 is used, WiFi firmware updates are supported
- If ESP32 is used, BLE joystick for use in simulators
- USB firmware updates are supported with a USB->serial chip (CP2102), ensure that reset signals from CP2102 or similar can be used to reset the ESP32 into upload mode without the need to button press
- Support for fan on/off control at higher RF power outputs. Fan is recommended for >250mW
- Support for LM75A temperature sensor
- Support for STK8BAxx accelerometer
- Buzzer support
- USB-C is recommended for modules that have USB

Transmitter Reference Designs

Use one of the reference pinout listed below:

- 2.4G External TX module:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/TX/EMAX%202400%20OLED.json
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/TX/Radiomaster%20Ranger.json
- 2.4G Internal TX module:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/TX/DIY%202400%20DupleTX.json
- 2.4G Gemini TX:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/TX/Generic%202400%20Gemini.json
  - https://oshwlab.com/jyesmith/expresslrs-gemini-tx
  - https://github.com/ExpressLRS/ExpressLRS/pull/1914
- 900M External TX module:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/TX/EMAX%20900%20OLED.json

Internal TX modules

- Internal (inside the handset) TX modules are fully supported when EdgeTX is running on the handset, which allows the TX to be flashed via “passthrough” from the USB port.

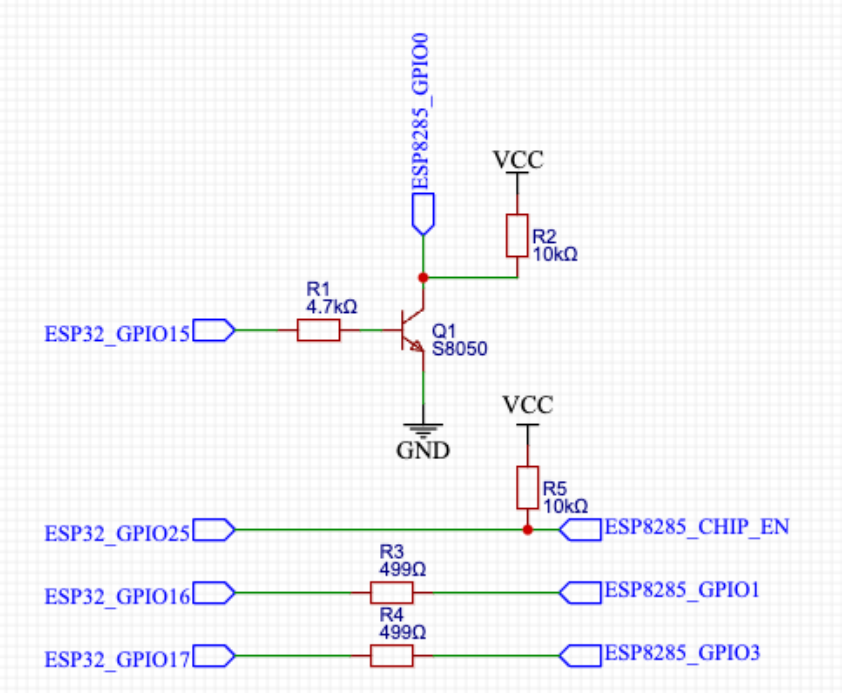
- Full duplex UART **must** be used between the handset MCU and the TX MCU, and the USB firmware update **must** be supported to allow recovery of "soft-bricked" modules
- A good wifi antenna is advisable (SMD or microstrip), especially on the internal modules, for wifi firmware updates, and also on the ESP8285 (backpack) for wireless control of other devices

Recommended Backpack Wiring

The wiring of the backpack ESP8285 to the ESP32 should be as follows.

- ESP32 GPIO16 to ESP8285 GPIO1 via series resistor
- ESP32 GPIO17 to ESP8285 GPIO3 via series resistor
- ESP32 GPIO25 to ESP8285 CHIP_EN
- ESP32 GPIO15 to ESP8285 GPIO0 via NPN transistor

If using a ESP32-PICO-D4, then GPIO16/17 should be GPIO9/10 respectively. This is because the ESP32-PICO-D4 uses GPIO16/17 for its internal flash memory.

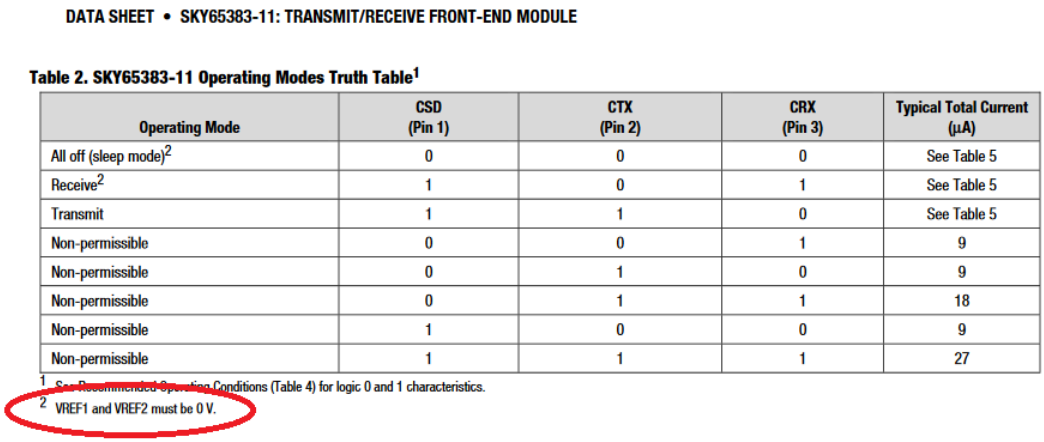


Wiring a backpack in this manner will allow the backpack firmware to be flashed via serial-passthrough from the ESP32.

Common issues encountered in TX designs

- **DO NOT ADD** load capacitors on SX1280 crystal oscillator without a proper test. SX1280 already has 10 pF internal load capacitance. Populating additional capacitors on the SX1280 crystal oscillator has been known to cause incompatibilities with hardware that does not use caps on the XTAL.

- The requirement specification of crystal oscillators on the SX1280 is **10 ppm tolerance and 10 pF load capacitance**. Low quality crystal oscillators or different load capacitance spec have been known to cause incompatibilities with other hardware.
    - To test the frequency error, follow the procedure described in here https://www.expresslrs.org/3.0/hardware/crystal-frequency-error/#measuring-absolute-xo-error-not-for-everyone
- The **Skyworks SKY65383-11** is a common 2.4G PA used in many designs. This chip **REQUIRES** VREF1 and VREF2 to be at 0V when in receive mode. Ensure the VREF pins are not simply tied to a constant voltage, and can be switched by the MCU.



DATA SHEET • SKY65383-11: TRANSMIT/RECEIVE FRONT-END MODULE

Table 2. SKY65383-11 Operating Modes Truth Table[1]

Operating Mode	CSD (Pin 1)	CTX (Pin 2)	CRX (Pin 3)	Typical Total Current (µA)
All off (sleep mode)[2]	0	0	0	See Table 5
Receive[2]	1	0	1	See Table 5
Transmit	1	1	0	See Table 5
Non-permissible	0	0	1	9
Non-permissible	0	1	0	9
Non-permissible	0	1	1	18
Non-permissible	1	0	0	9
Non-permissible	1	1	1	27

1 See Recommended Operating Conditions (Table 4) for logic 0 and 1 characteristics.
2 VREF1 and VREF2 must be 0 V.

- RP-SMA **must** be used for 2.4Ghz modules, and SMA **must** be used for 868/900Mhz. Internal / fixed antennas are excluded from this requirement.
- If the TX module has a USB port, it should be accessible externally, without needing to open the module enclosure.
- The value for the capacitor on the CHIP PU / ENABLE pin on the ESP32 should be >=1uF (10uF is known to work well). If this cap is under-valued, the auto-reset does not work in some cases during firmware updates.
- If the module includes any buttons (including a 5-way button), it would be advisable to use the GPIO / BOOT0 pin for at least one of the inputs, which will allow users to easily enter the bootloader by holding this during power-on.
- If 5-way joystick voltage divider values are too close to each other it may cause bad reads and ghost presses in the joystick so the values must be evenly spaced.
- When using high power amplifiers the active cooling heatsink and fan must be in direct contact with the chip itself, lack of proper cooling can cause signal degradation and failsafes.
- Modules with more than 250mW output power should have an option for external power to prevent brownouts from the radio power supply.

Receiver Design Guidelines

Supported RX MCUs

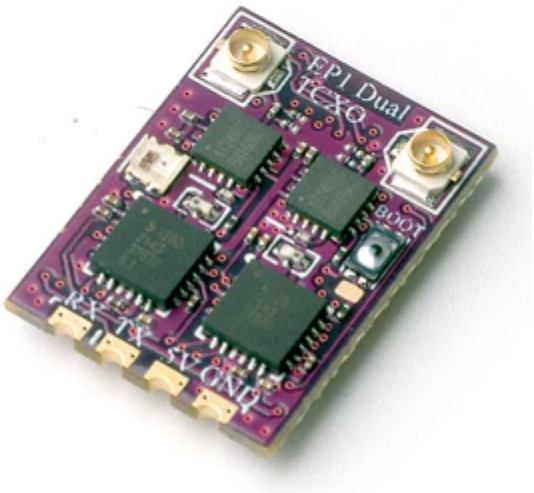
- **ESP32-pico-d4 (preferred for 6+ channel PWM and true diversity receivers)**
- **ESP8285 (preferred)**
- STM32F103
- STM32L432:
- STM32F303

Crystal oscillator

- https://www.expresslrs.org/3.0/hardware/crystal-frequency-error/
- Temperature-compensated crystal oscillators (TCXO) are ***strongly*** recommended!

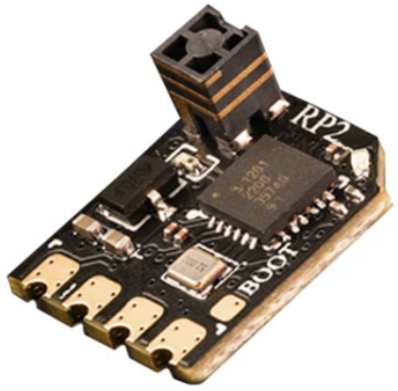
Receiver design recommendations

- This is a good example of the recommended RX layout – LED, Antenna Connectors, button are all on the top side, and the pad layout is a standard RX-TX-5V-GND with castellated mounting pads



- Receivers **must** use the standard crossfire nano pad layout (RX, TX, 5V, GND) seen from the top side (=where the antenna connected is placed) and **must** have a standard 2.54mm pin pitch. Suggest no larger than 12x20mm.
- Castellated mounting pads are strongly recommended
- A button is not strictly necessary on the RX, as binding is via the bind passphrase. A button can be helpful if the receiver has been “soft-bricked”, and

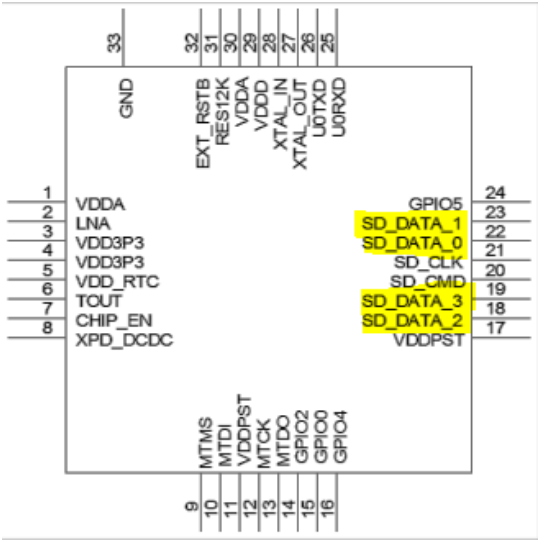
needs to be re-flashed from the bootloader. If a button is added to the RX, it **must** be connected to BOOT0.
    - Place LED and button on the same side as the antenna / IPEX connector
    - IPEX1 connector for external antennas (IPEX4 is **strongly** discouraged)
    - Molex SMD antennas are also an option on 2.4Ghz



- If ESP8285 is used
    - WiFi firmware updates are supported (use at least a simple trace antenna)
    - Voltage regulator **must** be spec to 500mA! Lower may cause boot failures.
- Diversity supported (antenna switching diversity, true diversity)

Common issues encountered in RX designs

- **DO NOT ADD** load capacitors on SX1280 crystal oscillator without a proper test. SX1280 already has 10 pF internal load capacitance. Populating additional capacitors on the SX1280 crystal oscillator has been known to cause incompatibilities with hardware that does not use caps on the XTAL.
- The requirement specification of crystal oscillators on the SX1280 is **10 ppm tolerance and 10 pF load capacitance**. Low quality crystal oscillators or different load capacitance spec have been known to cause incompatibilities with other hardware.
    - To test the frequency error, follow the procedure described in here https://www.expresslrs.org/3.0/hardware/crystal-frequency-error/#measuring-absolute-xo-error-not-for-everyone
- Voltage regulators that cannot support at least 500mA have been known to cause issues on boot
- The NN02-201 SMD antenna has been shown to perform significantly worse than other SMD based antenna offerings. The performance impact when using this antenna is unacceptably bad, therefore **this antenna should not be used** on ELRS receivers.
- The SD_DATA_0 and SD_DATA_1 pins on the ESP8285 **cannot be used** as GPIO pins, as they are used internally by the chip flash. These should not be connected to anything. The SD_DATA_2 and SD_DATA_3 pins can be used as GPIO, as ELRS uses DIO, not QIO for flash.



True Diversity hardware design issue

- Problem: In LoRa mode the SNR is below what you expect on a non-diversity receiver e.g. when testing on the bench a normal rx will have a SNR of >10. A problematic TD rx will be less than 10 and sometimes as low as 2. The problem can also be seen with reduced LQ in FLRC modes (SNR is not available for FLRC).
- However, the SNR/LQ is fine when used in Gemini mode.
- Solution: There is a hardware design issue seen on some prototype receivers (no released products). At this stage, a fix or reason is unknown, even after talks with Semtech. 🙁 The list below are possible workarounds (but not guaranteed to solve the problem)
  - Use a shared TCXO (single TCXO to feed both SX12xx chips)
  - When DCDC is used, it is possible to share the dcdc pins and eliminate an extra inductor, reducing the BOM
  - Separate the two SX1280s as far as the layout allowed

ESP32 PWM 14 & 16 channel receivers

- ELRS supports PWM receivers up to 14 channel with RF amps, or 16 channel without RF amps. These high channel count receivers use the ESP32 Pico D4 MCU.
- Strapping pins should be protected from servos that may pull them up/down during boot. This can lead to the rx booting in an unknown state. It is recommended to protect them in some manner e.g. non inverting buffer between the pin and servo.

Receiver Reference Designs

For pin allocations please use the following reference designs:

- 2.4G Simple RX
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400.json
  - https://github.com/ExpressLRS/ExpressLRS-Hardware/tree/master/PCB/2400MHz/RX_PP
  - https://github.com/ExpressLRS/ExpressLRS-Hardware/tree/master/PCB/2400MHz/RX_Nano
- 2.4G RX with PA/LNA
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20PA.json
  or
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20PA%20RGB.json
- 2.4G Antenna Diversity RX
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20Diversity%20PA.json
- 2.4G True Diversity RX (which is capable of doing Gemini RX). This target can be used with and without rf amps.
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20True%20Diversity%20PA.json
  - https://github.com/ExpressLRS/ExpressLRS/pull/1555#issuecomment-1146757077
  - https://github.com/ExpressLRS/ExpressLRS/pull/1555
- 2.4G PWM RX
  - 5CH: https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20PWMP5.json
  - 6CH: https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20PWMP6.json
  - 7CH: https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%202400%20PWMP7.json
  - 14 and 16 CH: https://github.com/ExpressLRS/ExpressLRS/pull/1970
- 900M Simple RX:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%20900.json
- 900M PWM RX:
  https://github.com/ExpressLRS/ExpressLRS/blob/master/src/hardware/RX/Generic%20900%20PWMP.json

- 900M True Diversity RX (which is capable of doing Gemini RX). This target can be used with and without rf amps.
    - https://github.com/ExpressLRS/ExpressLRS/pull/1993

SPI Receiver Design Guidelines

- SPI integration in Betaflight (expected in the stable release of betaflight 4.3)
- Reduces the cost with 1 less MCU
- Reduces end to end latency
- Disadvantages:
    - Firmware release frequency is slower due to integration with Betaflight
    - SPI is planned to be replaced by serial RX's for onboard designs, with a shared VTX/RX MCU (in development)
- Additional pins required
    - !!! It is recommended contacting the Betaflight Devs to confirm the final pin choice is suitable for DMA and Interrupt purposes !!!
    - RX_SPI_CS
    - RX_SPI_EXTI = DIO1
    - RX_SPI_EXPRESSLRS_RESET = Reset
    - RX_SPI_EXPRESSLRS_BUSY = Busy

TX/RX Final Approval Checklist

Transmitter

- Manufacturer provided Lua device display name (15 characters max) e.g. “Axis Thor 2400TX”
- RP-SMA connector for 2.4G antenna (suggested)
- Wifi range for firmware updates, and for backpack comms is at least 2m
- Bluetooth starts, connects to PC, and has a range of at least 2m
- Firmware can be updated via:
    - Passthrough / UART (double-check to confirm no issues with CP21xx timing)
    - Wifi access point / home network
- Measured power output matches expected output
    - Power measured on cold-boot
    - Power measured after >5mins operating at selected level
    - Maximum power heat-soak test for at least 1 hour
- RSSI/LQ for both uplink and downlink checked and compared against known good data
- If TX has backpack onboard:
    - Backpack wifi range is at least 2m
    - Backpack to/from main ESP32 comms verified as operational
    - Backpack can be flashed via passthrough and wifi
- MEGABauds are verified as working up to at least 3.75M
- If the TX supports >250mW RF power, a fan or other active cooling is onboard
- If the TX has a fan, it can be controlled correctly via the Lua script
- If the TX includes an OLED/TFT screen, the menu navigation is fully operational
- Existence of XTAL load caps are noted, and frequency offset checked and documented
- PA VREF1 and VREF2 verified to be correctly connected, and TLM RSSI checked

Receiver

- Manufacturer provided Lua device display name (15 characters max) e.g. “Namimno 2G4RX”
- Pad layout uses standard crossfire receiver ordering (GND, 5V, TX, RX) with 2.54mm pitch
- If a button is onboard, it is connected to BOOT0. If no button, BOOT0 pad is provided
- LED is operating with expected polarity
- LED on same side as antenna (suggested)
- Antenna connector is u.FL (IPEX1), not smaller MHF4/IPEX4 (suggested)
- VREG supports required current for wifi (>=500mA)
- Wifi range for firmware updates is at least 2m (10m suggested)
- Firmware can be updated via:

    - Passthrough
    - Wifi access point / home network
- If LNA/PA is onboard, measured power output matches expected output. If no PA, measured power output is ~17mW
- RSSI/LQ for both uplink and downlink checked and compared against known good data
- Frequency offset of SX1280 XTAL checked for compliance abs(fc) < 1000 (200kHz)
- Frequency offset of SX127x XTAL checked for compliance abs(fc) < 820 (100kHz)
- Diversity RX: Antenna switching works i.e. covering an antenna switches to the other and back again, RSSI visibly changes
- PWM RX: Jitter-free PWM output on all channels
- PWM RX: Receiver has proper strength pullups to boot with servo <10kohm impedance to ground on all channels.
- VBAT scale/offset valid for specified input voltage range (<0.5% error)

Flashing in the factory

Hardware flashing in the factory must be done using our approved flashing methods described below.

Generic ESP flashers are not suitable because they keep the default ESP data partitions. Due to the size of ExpressLRS firmware, binaries do not fit into the default ESP data partitions layout when doing OTA WiFi updates. The only way to recover from it is to flash using UART or BetaflightPassthrough methods. If that is not an option, users can recover the hardware wirelessly by using ExpressLRS/repartitioner binary.

Building the firmware

1. If there is a specific target built for your device in the ExpressLRS repository, then select the target you are building in PlatformIO and build that firmware. If there is no specific target you may choose one of the DIY targets that will work for your device.
2. Build the firmware in PlatformIO

After the build has completed you will have the firmware file(s) in a sub-folder of the .pio directory. E.g. `.pio/build/Unified_ESP32_2400_RX_via_UART/`
For all targets you will find

- `firmware.bin`

And for ESP32 targets you will also find

- `bootloader.bin`
- `partitions.bin`

For ESP32 targets you will also need to get the following files from your PlatformIO installation.
`~/.platformio/packages/framework-arduinoespressif32/tools/partitions/boot_app0.bin`

Windows environment

Command line interface

From the ExpressLRS `src` folder, note the following commands are entered on a single line.

For ESP32 based devices

```
python python\external\esptool\esptool.py --baud 460800 --before default_reset --after hard_reset write_flash -z --flash_mode dio --flash_freq 40m --flash_size detect 0x1000 bootloader.bin 0x8000 partitions.bin 0xe000 boot_app0.bin 0x10000 firmware.bin
```

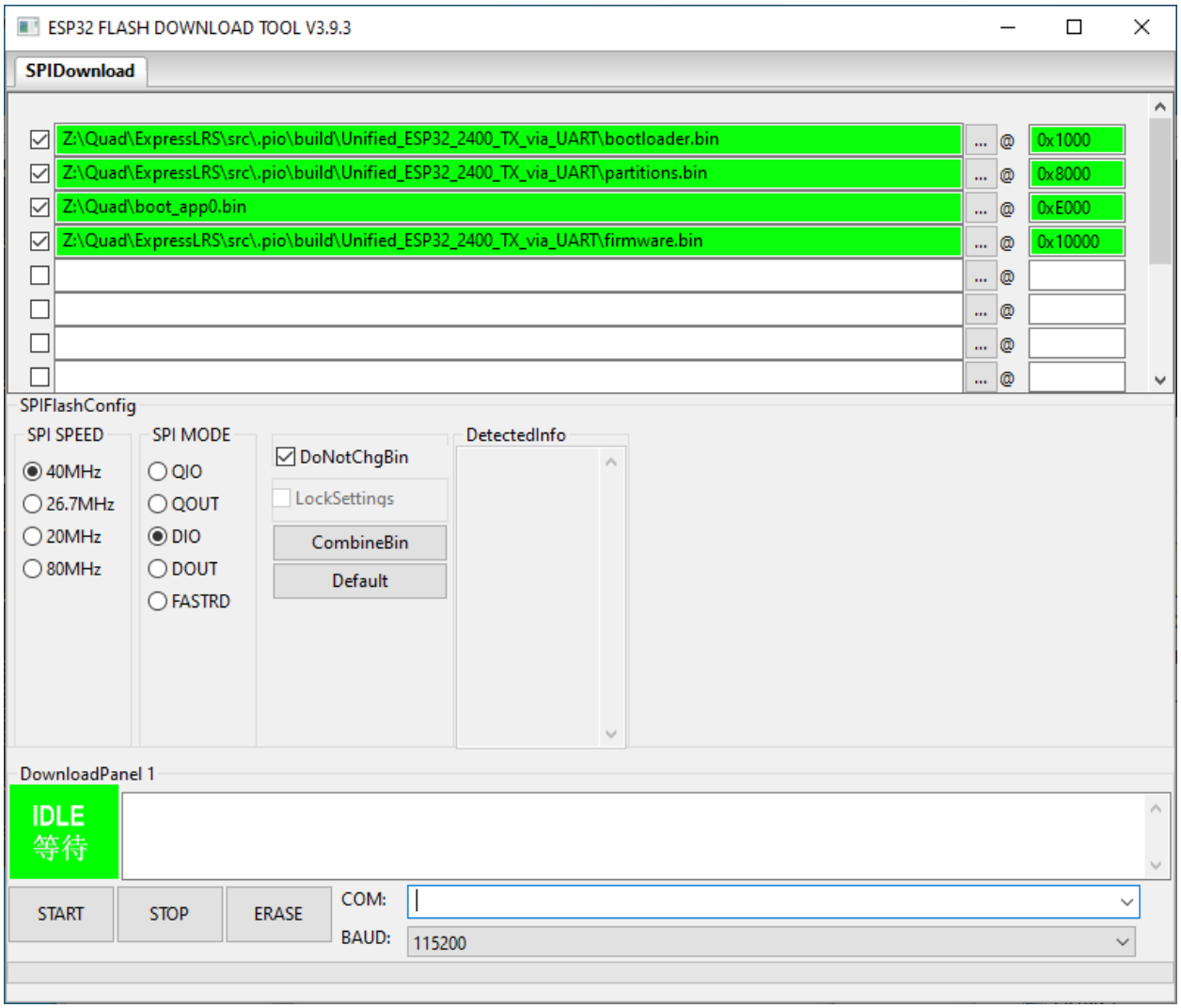
For ESP8285 based devices

```
python python\external\esptool\esptool.py --baud 460800 --after soft_reset
write_flash 0x0000 firmware.bin
```

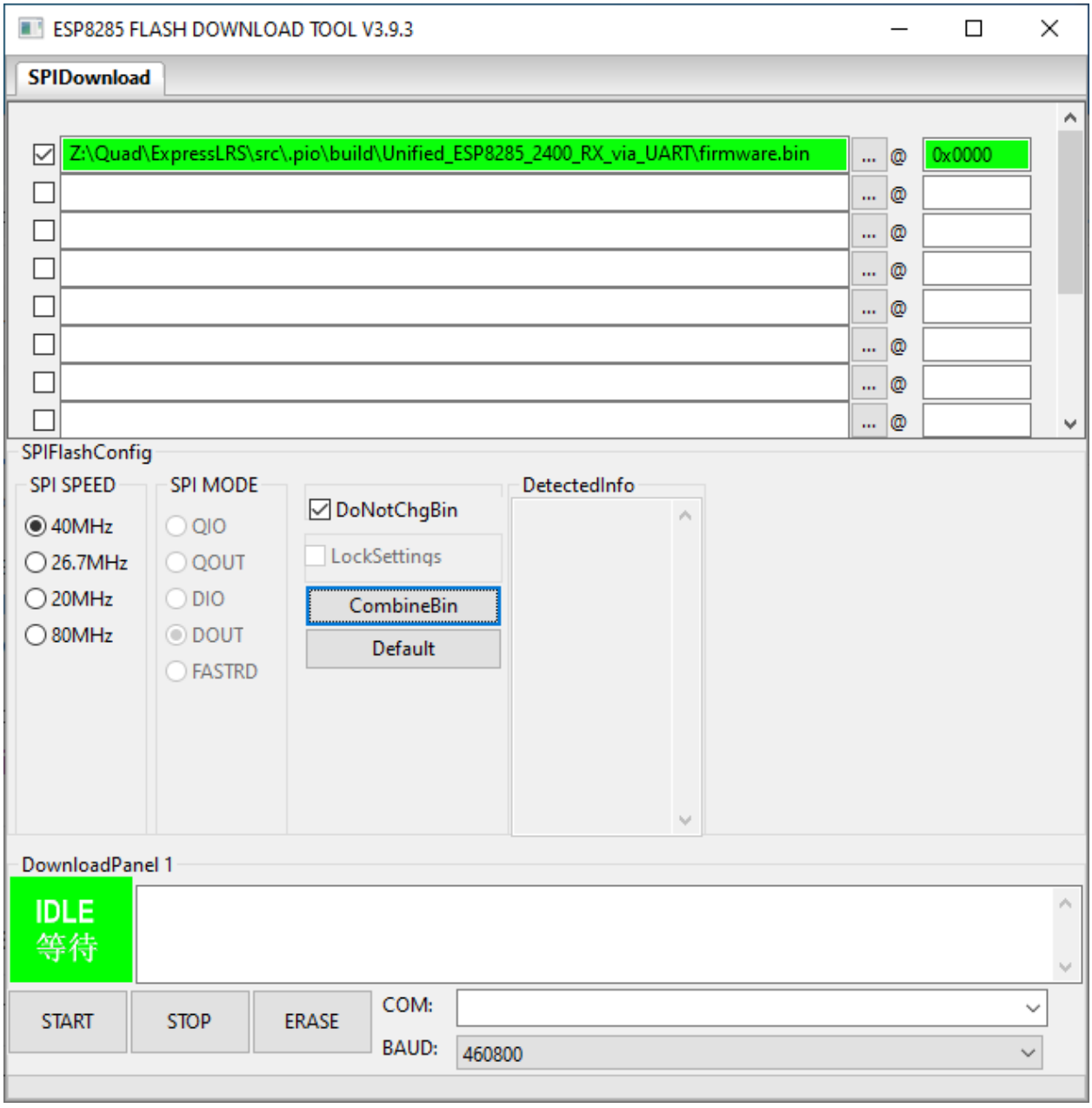
Graphical user interface

For flashing ESP based devices from Windows using the graphical interface, the ExpressLRS dev team recommends the use of the “Espressif Flash Download Tool”, this can be downloaded from the Espressif web site.

For ESP32 based devices



For ESP8285 based devices



Linux environment

Command line interface

From the ExpressLRS `src` folder, note the following commands are entered on a single line.

For ESP32 based devices

```
python python/external/esptool/esptool.py --baud 460800 --before
default_reset --after hard_reset write_flash -z --flash_mode dio --flash_freq
```

```
40m --flash_size detect 0x1000 bootloader.bin 0x8000 partitions.bin 0xe000
boot_app0.bin 0x10000 firmware.bin
```

For ESP8285 based devices

```
python python/external/esptool/esptool.py --baud 460800 --after soft_reset
write_flash 0x0000 firmware.bin
```

Information for marketing purposes

- Supported packet rates on 900mhz are:
    - 25Hz
    - 50Hz
    - 100Hz
    - 200Hz
- Supported packet rates on 2.4G are:
    - 50Hz
    - 100Hz (Full res mode)
    - 150Hz
    - 333Hz (Full res mode)
    - 250Hz
    - 500Hz
    - 1000Hz (FLRC only)
- Supported RF power levels are
    - 10mw
    - 25mw
    - 50mw
    - 100mw
    - 250mw
    - 500mw
    - 1000mw
    - 2000mw

- Supported protocols:
    - CRSF