ESPino - Specifications

Summary

Microcontroller	ESP8266 (32-bit RISC)
Communication	WiFi 802.11 (station, access point, P2P)
Operating Voltage	3.3V
Input Voltage	4.4-15V
Digital I/O Pins	9
Analog Input Pins	1 (10-bit ADC)
DC current per I/O Pin	12 mA
Max. DC current for the 3.3V Pin	800 mA
Flash Memory (Program)	4 MB
Instruction RAM	64 KB
Data RAM	96 KB
Boot ROM	64 KB
Clock Speed	80 Mhz

ESPino comes pre-programmed with the node-mcu firmware. It allows you to program it easily with the LUA programming language. You can read the node-mcu documentation here:

https://github.com/nodemcu/nodemcu-firmware/wiki/nodemcu_api_en

It can also be programmed directly in C or C++ via other tools like the Arduino IDE with support for the ESP8266 module.

Power

ESPino can be powered directly from the USB input (5V), or from the **Vin** pin with a source between 4.4V and 15V. The integrated LDO regulator is responsible of adjusting this voltage to the 3.3V needed for the correct operation of the board.

The power pins are:

- **Vin**: Unregulated external power input. The integrated regulator supports inputs between 4.4V and 15V at maximum 800mA
- **3.3V**: Internal 3.3V regulator output
- **GND**: Ground pins

Communication

The integrated ESP8266 chip brings wireless communications with the WiFi (802.11 b/g/n/d/e/i/k/r) standard. That way it can be easily connected to the Internet.

It can work either as an Access Point (Creates a network to which other devices, as a smartphone, can be connected), as a client of a router, or in WiFi Direct mode (P2P).

Also, it supports the following wired communication methods:

- UART (Serial)
- SPI
- I2C

For further details, see the ESP8266 Datasheet.

Integrated Devices

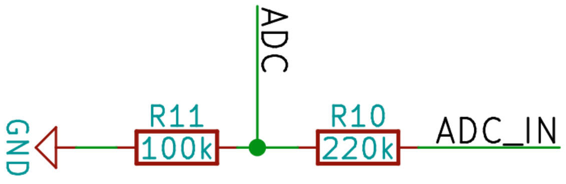
The board includes an RGB LED (red, green and blue), and two buttons. One is RESET and the other, USER/PROG, is connected to pin 0 of the ESP8266 and puts it in “Bootloader” mode when the device is restarted, or can be used by the user program after the board is started.

These devices are connected as follows:

● Red LED:	Pin 2
● Green LED:	Pin 5
● Blue LED:	Pin 4
● USER/PROG Button:	Pin 0
● RESET Button:	Pin RST

ADC

The ESP8266 board includes a 10-bit ADC than can sense an input from 0V to 1V. For convenience, the board integrates a voltage divider in that Pin, that allows the input to be between 0V and 3.3V.



Serial Programming

ESPino integrates an USB-Serial chip and a micro-USB connector that allows us to connect it to a PC. The PC will recognize it as a standard serial port, that will enable us to program it and to use serial consoles to send and receive messages to/from the board.

If the board isn't recognized automatically by the operating system, you should install the USB-Serial drivers, you can get them at espino.io.

Once the board is connected to the PC, we can program it in two ways:

1. With the Lua programming language of the integrated node-mcu firmware (without entering "Bootloader" mode)
2. With a custom firmware written in C or C++, like from the Arduino IDE with support for the ESP8266. It is necessary to enter "Bootloader" mode

Bootloader Mode

The "Bootloader" mode allows us to upload new firmware to the ESP8266, or to restore the node-mcu firmware that comes installed by default.

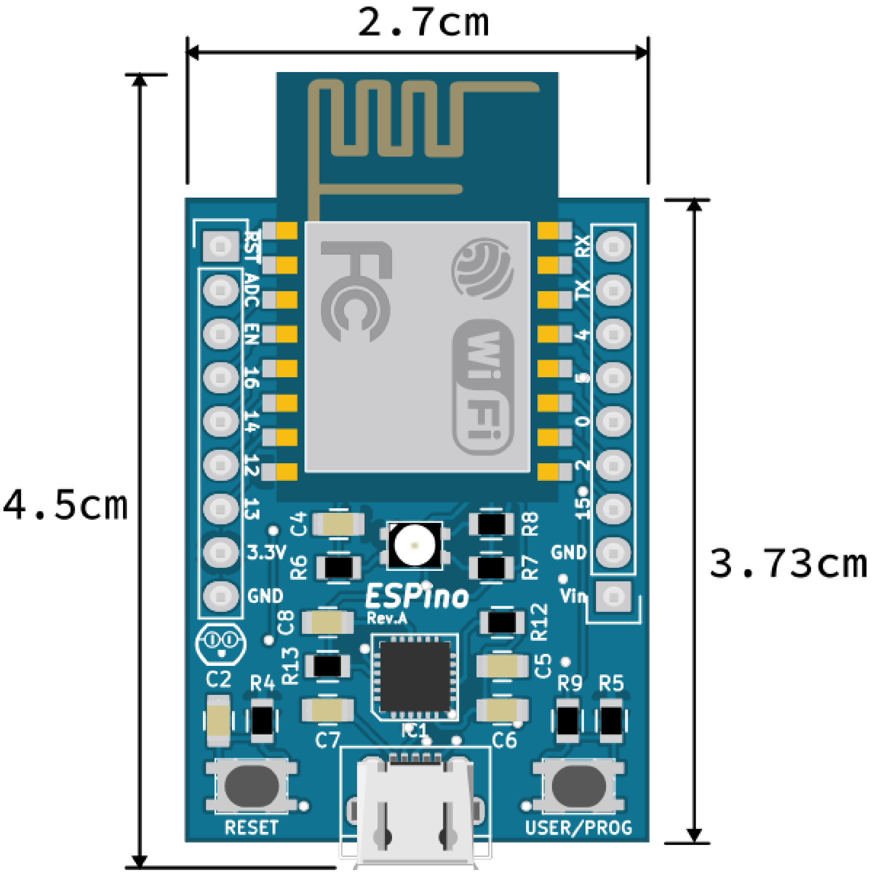
Steps for putting ESPino in Bootloader mode:

1. Press the RESET and USER/PROG buttons at the same time without releasing them
2. Release the RESET button, keeping USER/PROG pressed
3. Wait a second and release USER/PROG

Once in Bootloader mode, we can use a tool as "esptool" or the Arduino IDE with support for ESP8266 for uploading the firmware.

Physical Dimensions

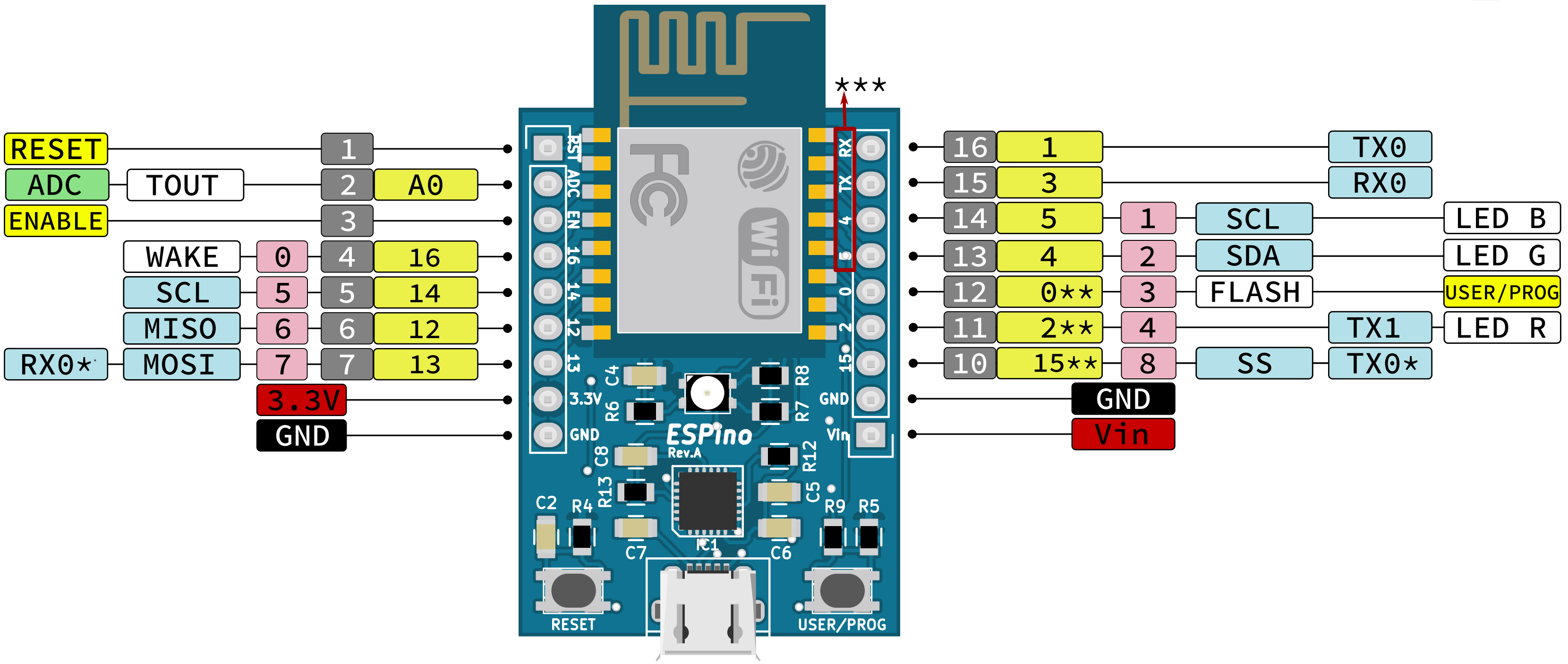
The pins of ESPino are designed for fitting in a standard breadboard with 2.54mm spacing.



Support

- For downloads, tutorials and more info, go to http://espino.io
- For support and questions, go to the Aquila Community Forum: http://community.aquila.io/
- Contact us at: info@makerlab.mx

ESPino
PINOUT DIAGRAM



Notes:

- Pins 0 to 15 can be PWM, depending on the firmware, it varies how many can be used at the same time

* Pins RX0*(13) and TX0*(15) can be used as serial port instead of TX0 and RX0. This is achieved in the Arduino IDE by calling Serial.swap() after Serial.begin()

** Pins 2 and 0 must be high and pin 15 must be low at startup. This is achieved with resistors in the board, however these could be not enough when connecting external devices on those pins.

*** In ESPino Rev. A, the RX and TX, 4 and 5 labels are inverted by mistake

- GND
- Power
- Control
- Physycal Pin
- Arduino IDE Pin
- Extra Function
- Analog Pin
- PWM Pin
- Communication
- node-mcu Pin

makerlab.mx

v0.2.0
Graphics inspired by works from pighixxx.com