Articles Embedded C & MCU

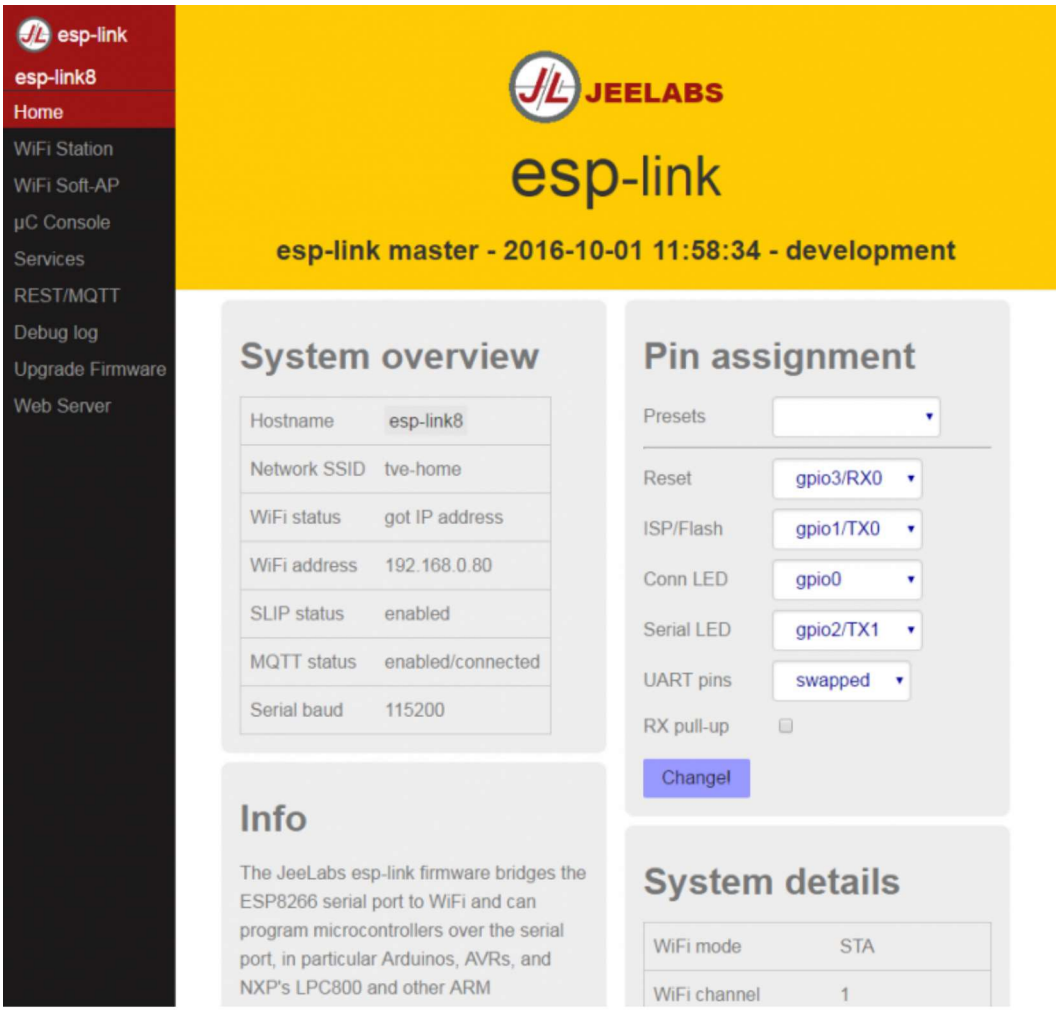
WiFi-UART Serial Bridge Using ESP8266 or ESP32

Yahya Tawil • 10th April 2018 18 86,345 4 minutes read

Hardware debugging and programming via UART/Serial connection has always been something so vital to hardware developers and even other team players like QA engineers. In certain circumstances, the device that needs to be debugged or programmed is not reachable by wires. I remember how QA engineers in my previous job had to come up with a Bluetooth adapter to bridge the serial debugging information of the tested products because it was fixed inside the car dashboard and can't be reached out. In this article, it's time to make some serial data fly over the air!

Serial Bridge Using ESP8266

One of the most well-known (appeared on Hackaday's blog) and well-designed projects to make ESP8266 as a WiFi-UART bridge is jeelabs's esp-link. Actually, this project is far beyond being a simple serial bridge as it also manages MQTT client pub/sub and REST HTTP requests in order to connect the MCU to the internet. Moreover, it can be used to flash the attached MCU. Esp-link has a very handy web interface stored inside the ESP.

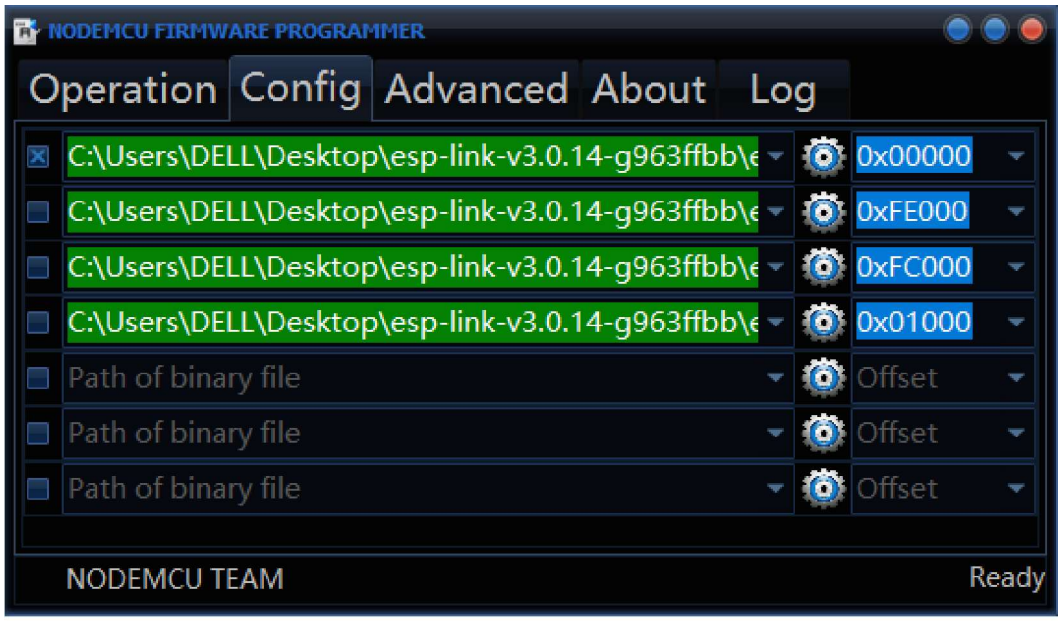


To make the test, we will use ESP8266 development board, specifically NodeMCU.

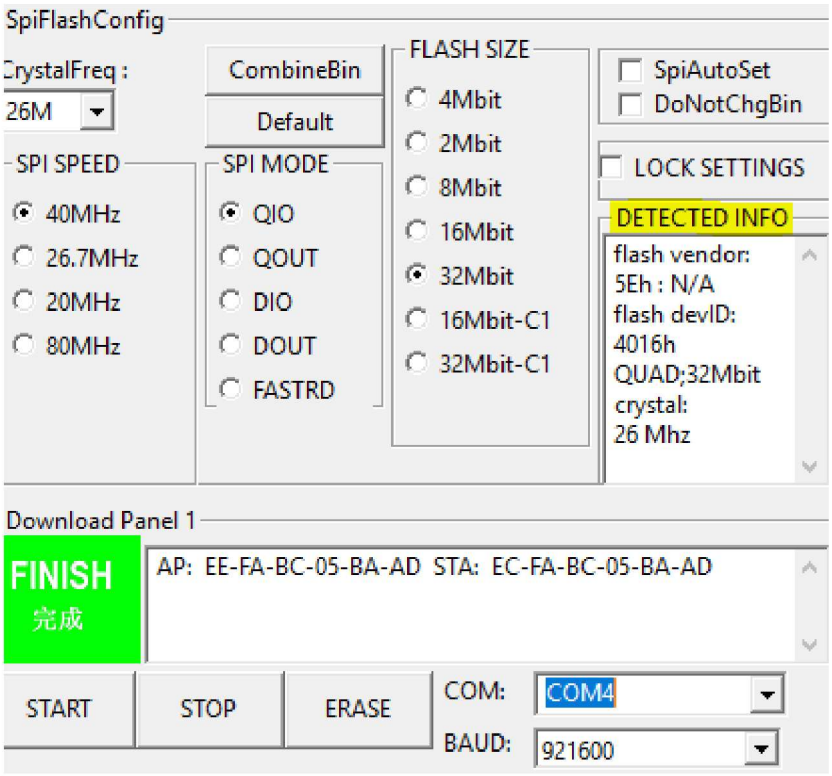
Prepare The Software

First, we need to download one of the release images of the project from releases page. The compressed file (A .tgz file is beside the download word in bold) should have 5 bin files :blank.bin, boot_v1.x.bin, esp_init_data_default.bin, user1.bin and user2.bin. The ESP8266 board should be connect after downloading one of the available download tools like: Flash download tool from Espressif or Nodemcu flasher.

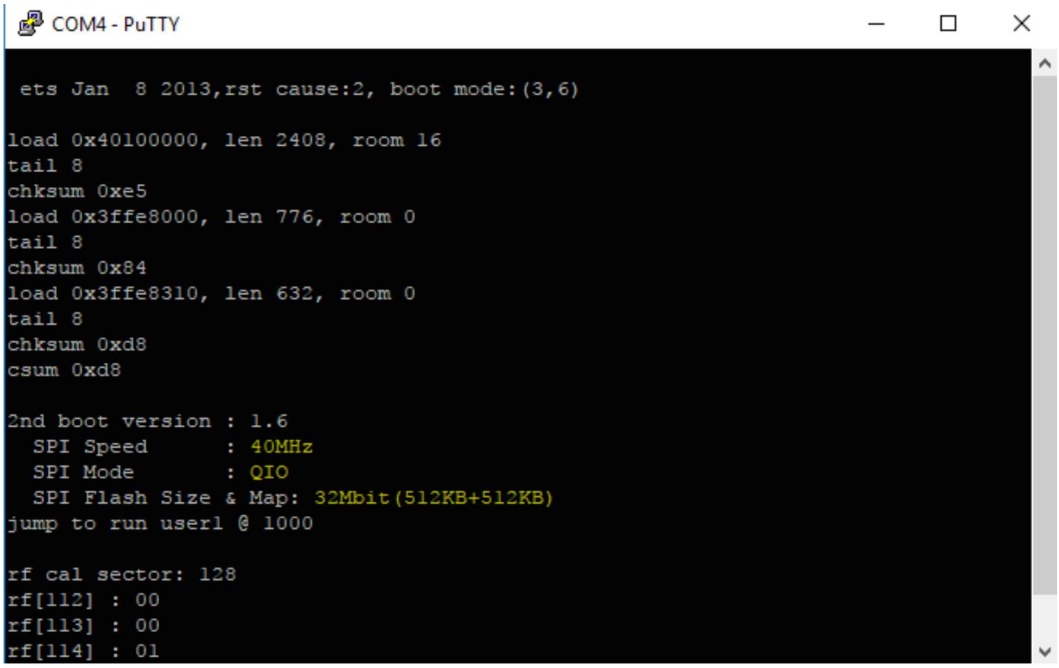
Now, follow the flashing guidelines available in Readme file according to your ESP8266 edition and set the bin files addresses according to your module memory size.



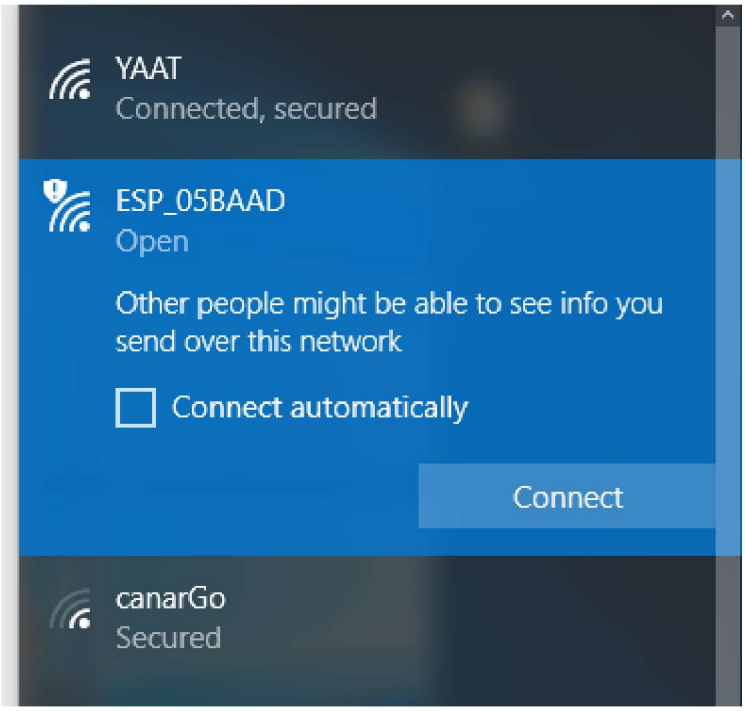
You can simply make the Flash download tool from Espressif find it for you by connecting it and pressing “start” to get the required specification from the boot message.



Alternatively, you can read the boot message using a serial console with 76600 baud.



After flashing the image files successfully, you should see a new WiFi network.



Connect to it and type this IP address into the browser http://192.168.4.1 . Now, you should see the home page.

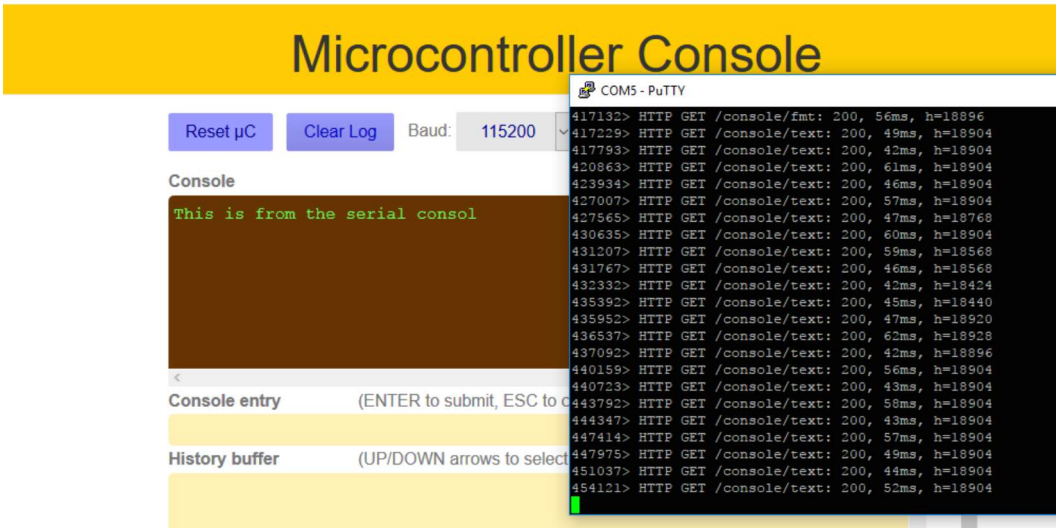
Prepare The Hardware

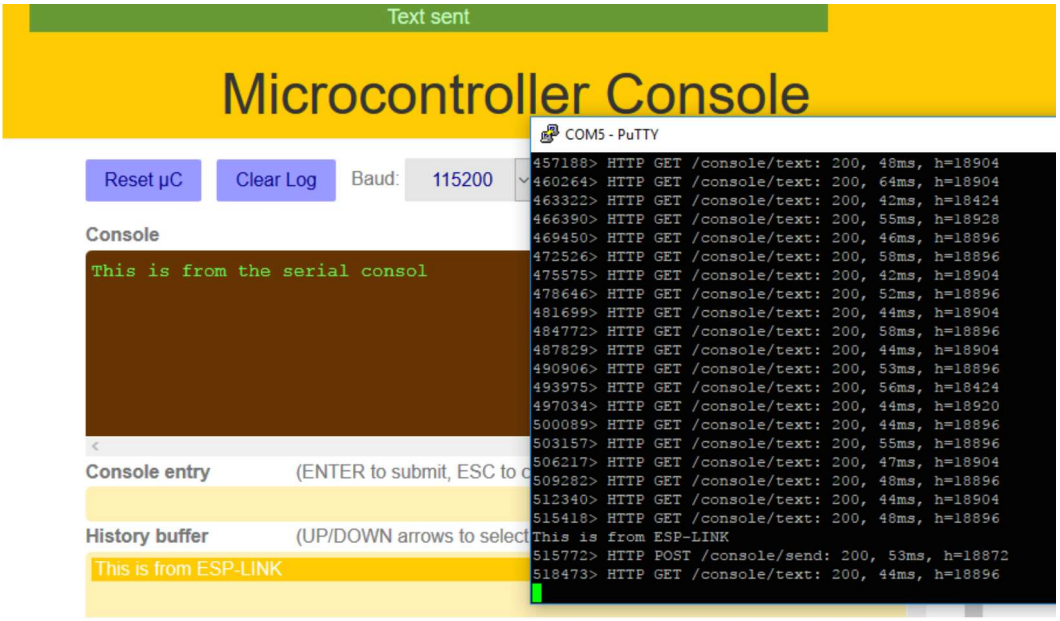
To test the serial bridge code, we will achieve the following connection: TX0 from ESP with RX of the device and RX0 with TX of the device that sends serial

information and needed to be bridged via WiFi. For the sake of the test, the device here would be a USB-TTL cable (USB-UART converter) and the data will be sent and received from a console (PUTTY for instance).

Image Courtesy of ElecFreaks

Now, let's see some serial data fly over the air!





Finally, I think a considered amount of time is needed to explore the other options of this amazing project!

Serial Bridge Using ESP8266 (Simpler)

The previous project could be too complicated to do the simple job of converting Serial connection to a WiFi connection. So I decided to port a project described in the rest of this article written in Arduino. The ported code works on ESP8266 and can be download from here:

INO **WiFi-Serial ESP8266**
1 file(s) 5.97 KB
DOWNLOAD

The explanation in the next section should be followed to know who to use it but with ESP8266, where simply by using a specific IP/Port via a Tellnet connection, the Serial data can be reached out.

Serial Bridge Using ESP32

A project via Github presents a WiFi to Serial bridge for the 3 UART ports available in ESP32. This project is written using Arduino IDE and supports ESP32 as an access point (AP) that broadcasts a specific WiFi network with predefined SSID and password in the code or a station. However, every UART

port on ESP32 is accessible after making a WiFi connection with ESP32 via a specific IP and port.

Let's prepare the needed software and hardware to test this project.

Prepare The Software

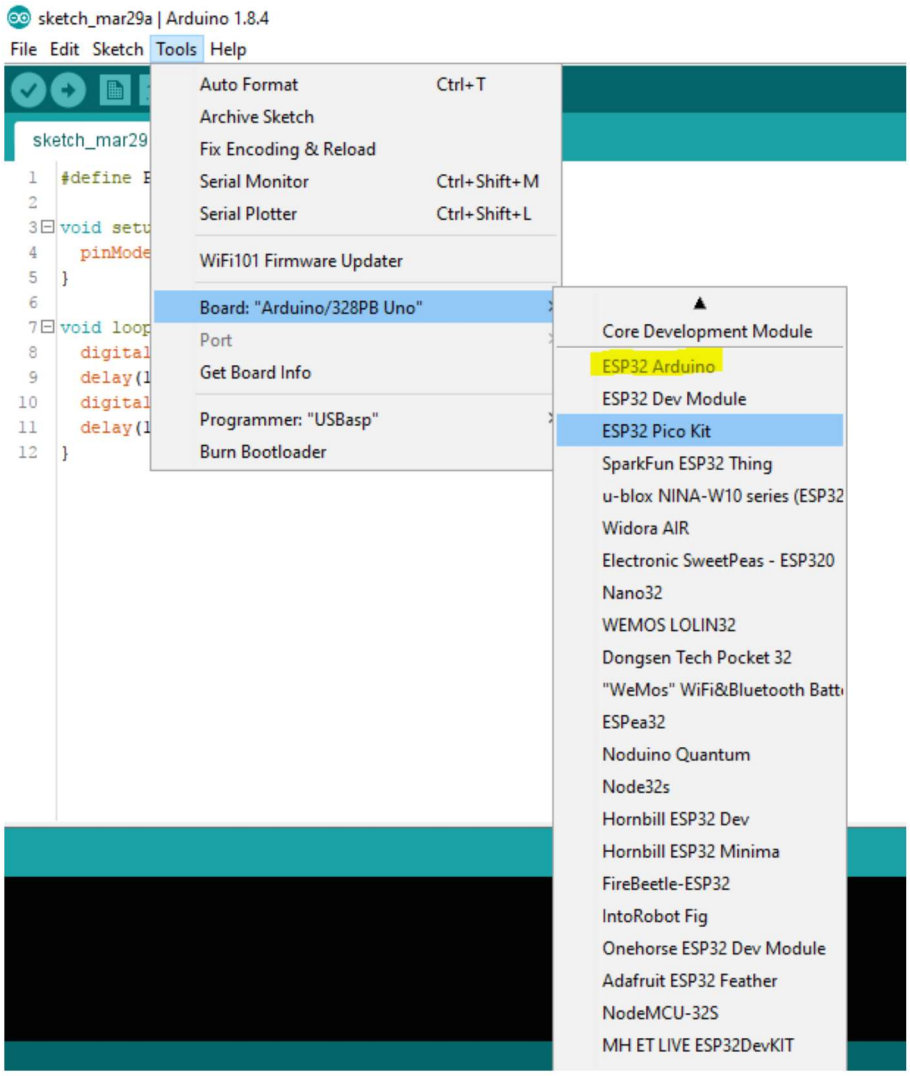
As the project uses Arduino SDK with ESP32 core, then we need to add Arduino core for the ESP32 to the Arduino environment. The official documentation of ESP32 core from Espressif has an installation guide, but the easiest way to add this core is to download the repo as a zip file then decompress it in the following directory:

(Windows) \Documents\Arduino\hardware\espressif\esp32

(Linux) folder is named "Sketchbook" and it is typically located in /home/

The next step is to get the compilation tool-chain xtensa-esp32-elf by executing one of the get.exe or get.py files available in the tools folder according to your machine type.

To test this all together, open the Arduino IDE and check the Tools>boards menu



Now, testing the compilation of a simple program is needed also. The following test code can be used.

```
#define PIN_LED 1 // built_in
void setup()
{
pinMode(PIN_LED, OUTPUT);
}
void loop()
{
digitalWrite(PIN_LED, HIGH);
delay(1000);
digitalWrite(PIN_LED, LOW);
delay(1000);
}
```

Prepare The Hardware

ESP32 has many different development boards. The one used here is Goouuu-ESP32 Development Board

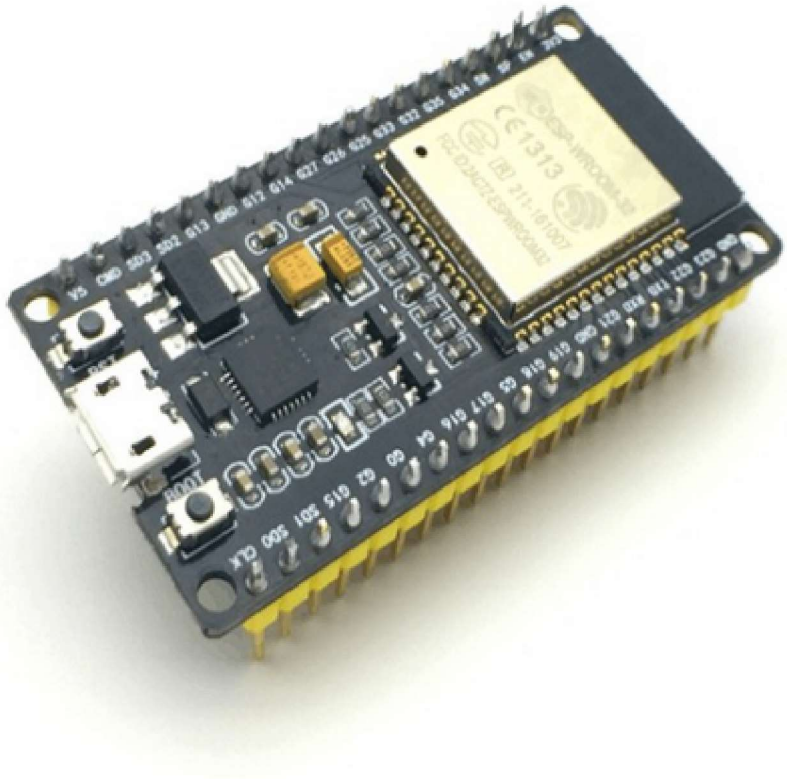


Image Source: Aliexpress

And has the following pinout:

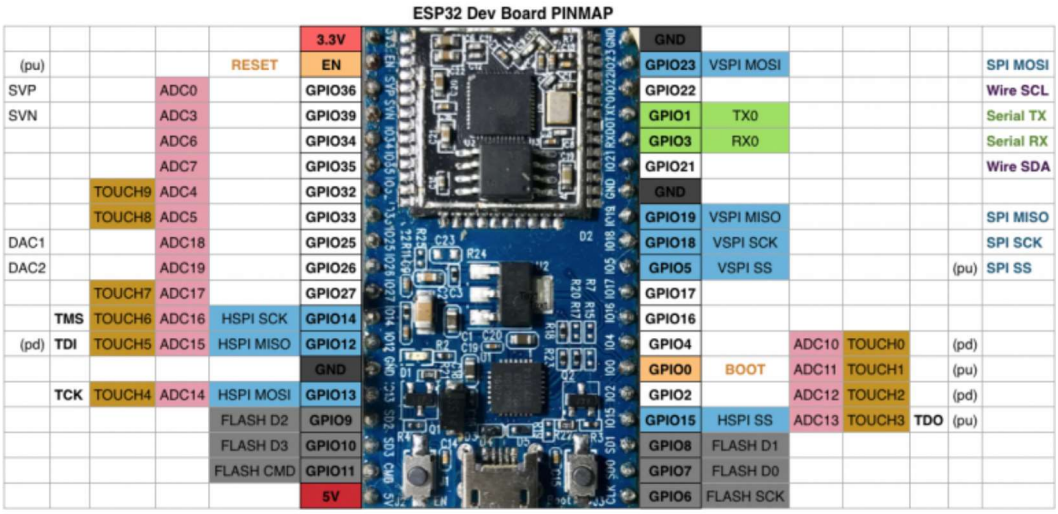
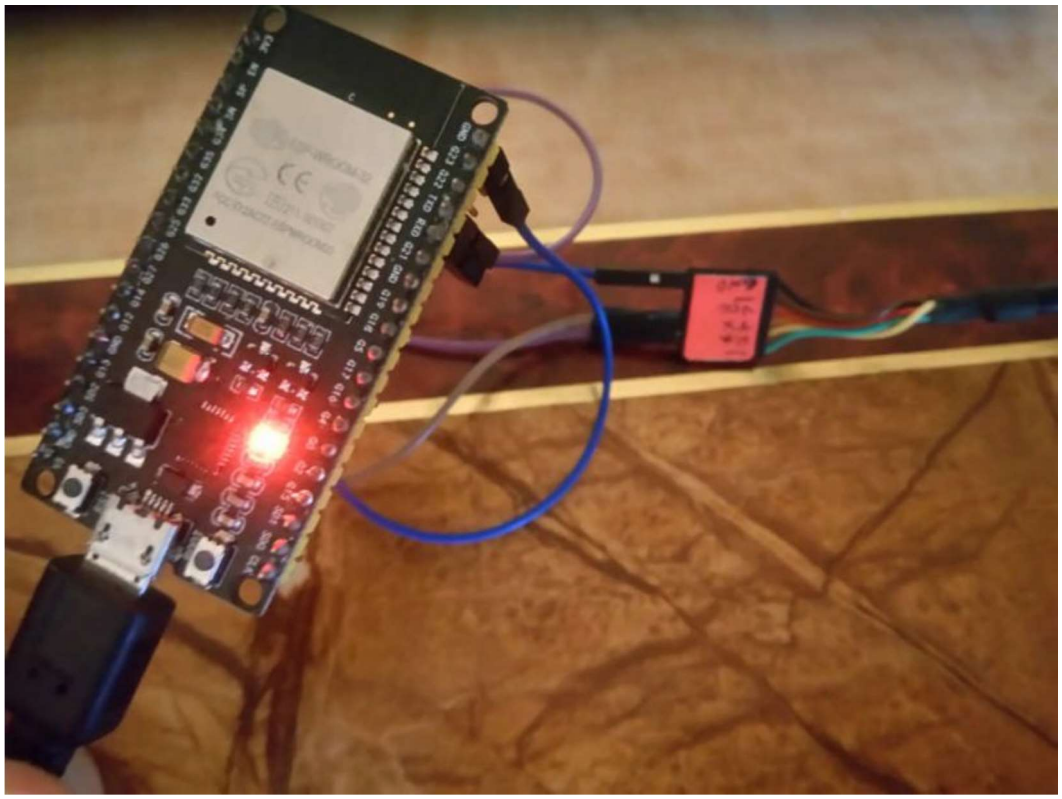


Image Source: Espressif Github Repo

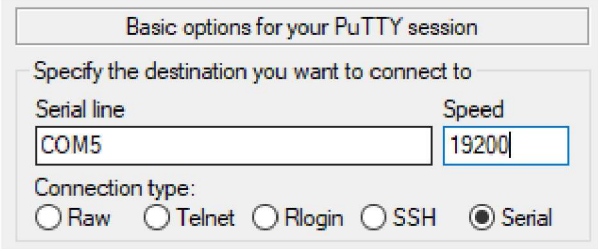
To test the serial bridge code, we will do the following connection: TX0 from ESP with RX of the device and RX0 with TX of the device that sends the serial information and needed to be bridged via WiFi. For the sake of the test, the device here would be a USB-TTL cable (USB-UART converter) and data will be sent and received from a console (PUTTY for instance).



Now download the Arduino code from Github and upload it to ESP32. Choose from Tools>boards>ESP Dev Module. Many things can be configured before uploading the code to ESP32 like choosing the working mode as a station or an access point and many other things.

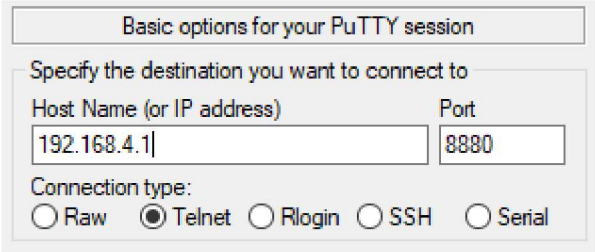
```
#define MODE_AP // phone connects directly to ESP

//#define MODE_STA // ESP connects to WiFi router
```

If no changes are done, the bridge will broadcast a WiFi network with SSID "LK8000" and password "Flightcomputer". Connect to the WiFi network and open the Putty program then use the following setups: one for the Serial connection for the USB-UART converter.

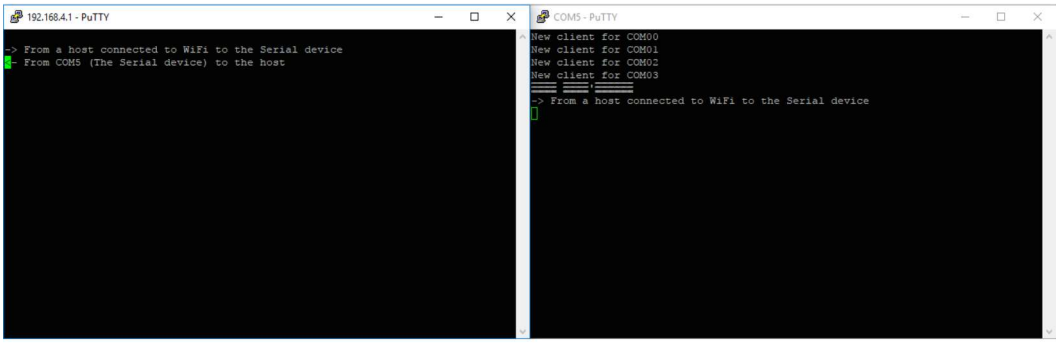


And another to connect to the WiFi server using one of the following IP/Port according to the used UART port.

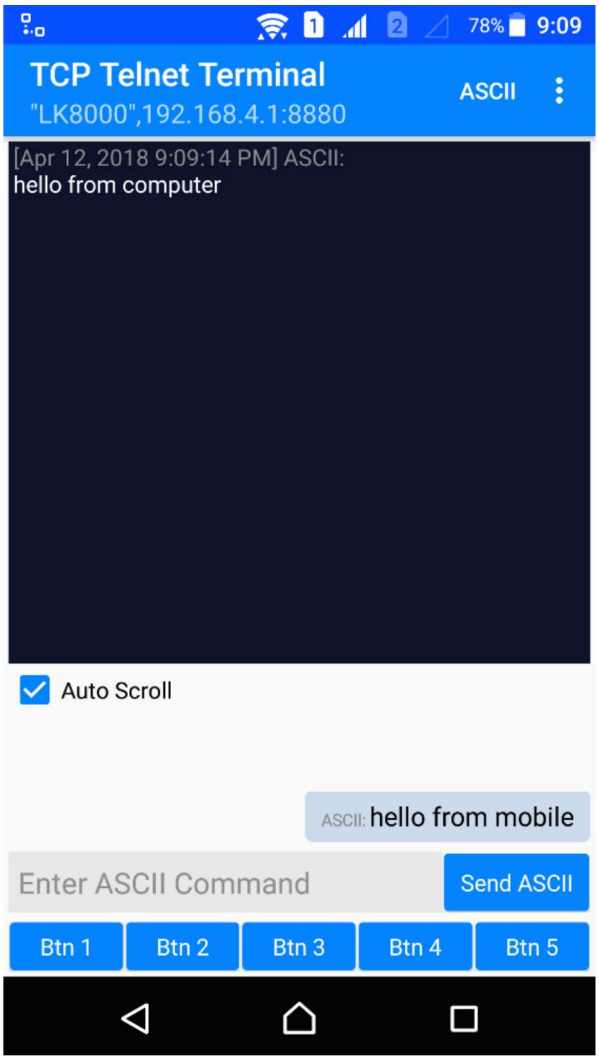
- 192.168.4.1:8880 for UART0
- 192.168.4.1:8881 for UART1
- 192.168.4.1:8882 for UART2



Now, open the two consoles and try the magic!



Note: Telnet connection can be done from a smartphone by using Telnet client application like Termius or any other working application.



Example

Do you need to know more about ESP32? Read this series "All About ESP32".

18 Comments

StDenits

6th May 2019 at 3:16 am

Hello.

Is it possible to use it for such a chain: UART ESP32-1 ESP32-2 UART.
I would like to connect two devices via wi-fi and send \ receive data.

Reply

Antonio

1st October 2019 at 11:34 pm

Hello, can do? I want. Same
Thanks.

Reply

Yahya Tawil

18th November 2019 at 11:23 pm

Yes it is doable, but there is a need to write a code for ESP to advertise an Access point to make the client connects to. I think you may find something similar like this one: http://www.geekstips.com/two-esp8266-communication-talk-each-other/ but still need modification to listen to UART and send via the connection.

Reply

Shamika Ghodke

8th October 2020 at 8:54 am

YES Sir, i have used in my project;
it can be done usart to wifi with both esp8266 and connect to same wifi thankyou

Reply

Eben

29th December 2019 at 6:35 pm

Hi!
Thank you very much for this. How do I use this to display my programatically generated debug serial messages from my sketch?
Typing in the wifi and COM terminal windows work as expected, but the messages that my sketch sends to the COM terminal is not mirrored in the wifi putty connection.

Is there any way to display those messages over the wifi terminal?

Reply

Flo

7th January 2020 at 3:05 pm

Hi,
I tried something like that but I blocked on some limitation.. I need to connect UART @1,2Mbps and send A LOT OF DATA in continue!
So my data are OK @ 250kbps but @ 1,2Mbps there is a lot of errors :/
Anybody did try that ?
What can I do to make it better ?
Maybe on IDF ?
Thanks for your help,

Reply

Marcus

28th January 2023 at 2:10 pm

Did you find a solution for your task? I would also be interested to run it with 921 kBd …
Please let me know!

Reply

Igor

17th March 2020 at 7:27 am

Hi. I tried your project on ESP32. In config.h changed the speed to 115200. There was a problem when transmitting large amounts of data . The first 12 kB file is transmitted normally. Then, after a pause, the same file is transmitted with errors. The further you go , the more mistakes you make . Then it stops transmitting even single bytes at all. It looks like the memory is filling up somewhere and not being cleared. I couldn ‘t figure out the code myself. Can you help? A very necessary device

Reply

Yahya Tawil

18th March 2020 at 3:17 am

It is better to open an issue on the original repository on Github (https://github.com/AlphaLima/ESP32-Serial-Bridge/blob/master/ESP32-Serial-Bridge.ino) . Anyway, I think it better to disable features in code you don’t want like bluetooth, OTA , and finally try to increase the buffer size in config.h

Reply

amgad

6th July 2020 at 11:53 am

Did you try this project to connect to cisco router console ??

Reply

Yahya Tawil

7th July 2020 at 1:50 am

No. Did you have any problem with it ?

Reply

CHRIS EFSYTATHIOU

6th January 2021 at 10:27 am

Hi.
I am trying to use the simple version (not ESP-LINK) but it does not compile when i uncomment the UDP definition
//#define PROTOCOL_TCP
#define PROTOCOL_UDP
It gives me the below error
/home/hendrix/paparazzi/wifi-serial/wifi-serial.ino: In function ‘void loop()’:
wifi-serial:175:13: error: ‘TCPClient’ was not declared in this scope
if (TCPClient[num][cln])
^
wifi-serial:199:14: error: ‘TCPClient’ was not declared in this scope
if(TCPClient[num][cln])
^
Using library ESP8266WiFi at version 1.0 in folder: /home/hendrix/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/libraries/ESP8266WiFi
exit status 1
‘TCPClient’ was not declared in this scope

Basically i am trying to build a transparent serial connection in order to connect to a datalink server that can use UDP instead of the normal serial port like /dev/ttyUSB0.
Chris

Reply

codebeat

30th May 2021 at 4:47 pm

Hai, great project. I am very interested in the WiFi-Serial ESP8266 code however the file is empty/invalid! Can you update this please? Thank you for your kindness 😉 Kind regards, Erwin.

Reply

Yahya Tawil

1st June 2021 at 5:00 pm

Thank you very much for reporting this. It is fixed now. The download manager blocked .ino files after last update.

Reply

Harry

18th July 2021 at 8:55 am

Hello
Loaded the simple version in a nodemcu
What is the password to use (because it is not password free at the moment)?

Reply

Harry

18th July 2021 at 1:20 pm

found out that the password was: 123456789 . Made the wifi connection with the AP.
Gott it working: putty terminal on a PC com-port (serial setting) and TCP telnet terminal on the wifi side (Serial_wifi 192.168.4.1:8880). Could send and receive ascii characters vice versa.
In the Arduino scetch I also entered my home network SSID and Password before compiling and uploading. But i did not see the unit appear on my home network.
How to accomplish that?
Also: how can I upload and download files via this setup?

Reply

S N

29th December 2021 at 2:52 pm

Hello,
I have EEG data available via a UART interface. I need to use the ESP Wifi to read data from UART and send it via WiFi Direct to a PC. I need to be able to copy the data and then use an application. Do I need a special application on the ESP device as well as the PC to effect this ? Or a regular TCP socket application will work ?

Thanks & Regards,
Sn

Reply

Wijnand

23rd February 2023 at 1:52 pm

In “Serial Bridge Using ESP8266 (Simpler)” I downloaded wifi-serial.ino and tested it with success. But wifi-serial waits until there is a newline before it send the string to the serial out. My question: is there a way to do this on character base?

Reply