AZ-Delivery esp826601

1. Pinout and Setup of the esp8266

The AZ-Delivery esp826601 has eight pins, two of which are used for the power supply (VCC and GND), two for the serial communication (RXD and TXD), for example with the Arduino, and two as GPIO (GPIO 0 and GPIO 2). Additionally, the Wi-Fi module has a reset pin, as well as CH_PD, to reboot.

8266 01's VCC and GND must be contacted with the power supply's VCC and GND or with the Arduino. In addition, it is always recommended to use an external voltage source when *esp8266 01* is used with the Arduino, as the Wi-Fi module may require a brief moment approx. 250mA for data transmission, especially for long-distance transmissions, which can be too great of a burden for the Arduino and can cause errors. It is important to note that the device only supports 3.3V.

During normal use, of the device, the reset pin can be switched to HIGH via a pull-up resistor, switched to LOW, via a pull-down resistor, and for CH-PD, switched to HIGH, via a pull-up resistor. If you want to restart the *AZ-Delivery esp826601*, then for a brief moment, you must switch the reset pin to HIGH; for instance, via an external reset button, which is connected with +3.3V via a resistor.

If you want to connect the *esp8266* with an Arduino or Raspberry Pi, then you must connect TXD with RX and RXD with TX.

You can connect sensors or other things with the Wi-Fi module via GPIO 0 and GPIO 2.

2. Utilization of the *esp8266 01* as a WLAN module for the Arduino

2.1 Principles of communication via AT commands

The *AZ-Delivery esp8266 WLAN module* has a default preinstalled firmware, with the help of which connected devices, for instance, Arduino, can communicate with via AT commands. The AT instruction set is relatively old (approx. 30 years) and was originally developed for the communication with modems. However, its simplicity and thriftiness make it ideally suitable for the communication with the *esp8266*.

All AT commands begin with „AT+" and are then followed by the actual command. It is also important that all AT commands end with a line break (CR and NL). For this reason, you should always use *printIn*, when you send a command via the serial connection between *esp8266* and Arduino, to enclose the line break and to tell the module that the command has to be executed.

2.2 Download of a Website

In this example, we would like to tell the *AZ-Delivery esp8266*, via AT command, to download a website. To do this, firstly the WLAN module needs to be set to "WLAN Client" mode via the *AT+CWMODE=1* command. Then it can be connected to a WLAN network via *AT+CWJAP="SSID","Password"*.

Now you can make a connection with the destination server via *AT+CIPSTART="TCP","website.de",80*. Then you have to tell the Wi-Fi module, via *AT+CIPSEND=NUMBER*, how long the inquiry, which you would like to send to the server, is. You still have to add a few characters, because the line breaks must be included at the end.

The inquiry to the server follows the principle *GET /Pfad_zur_Website.html*. Now you may have to possibly add blank space characters until the previously specified number of characters had been reached; whereupon the *esp8266* sends the inquiry automatically to the server, and then also automatically sends the answer (HTML code of the website) back to the Arduino. As long as the specified

number of characters is not reached, then the *esp8266* will not do anything. If the inquiry is longer than what is specified, then *"busy"* will be issued.

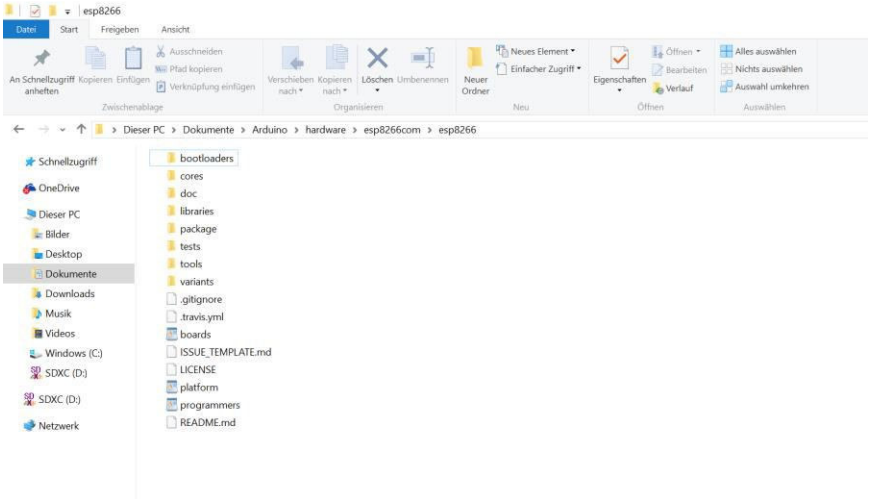
Due to the complexity of this procedure, it should be initially tried on the PC via the serial console, before executing the procedure in the Arduino Code, as troubleshooting will be very hard to arrange.

3. Autonomous operation of the *esp8266 01*

3.1 Preparation of the Arduino IDE

In addition to using the AZ-Delivery *esp8266 01* simply as a WLAN module for the Arduino, it is also possible to autonomously operate it, so that it carries out the core program. It can be conveniently programmed directly in the Arduino IDE with all of its customizations.

For that, you will have to download a folder on *Github* (https://github.com/esp8266/Arduino), which is needed for the Arduino IDE to recognize the *esp8266* and to be able to work with it. The contents of the .zip folder must be now saved in the Arduino folder, in a subfolder hardware under *esp8266com*.



3.2 Programming the esp826601

3.2.1 Download of a Website

This example is about programming the AZ-Delivery *esp8266 01* so that it can download a website. This will, subsequently, allow you to get any information from the downloaded website.

In order to download a website, the Wi-Fi module must first be connected to an existing Wi-Fi network, which has an internet connection. Then you should create an object from a *WiFiClient* type, which undertakes the communication with the server, and which sends the inquiry that the website should be sent to it.

Now the *WiFiClient* saves the received data in a string that displays the whole HTML code

of the website; not just the displayed text that you see, when you look at the website in the browser. This string can now be edited and searched for via the respective commands.

Based on this principle, there are already many example sketches, under *File* > *Examples* > *ESP8266WiFi* > *WiFiClient*.

3.2.2 esp8266 01 as a web server

The AZ-Delivery *esp8266 01* can also be used as a normal web server. However, it is not recommendable to use it for a normal internet website, as its hardware resources are not in favour of other factors, such as low power consumption or a low price. Nevertheless, it is ideally suited to provide information, for example from connected sensors; as well as to accept user input for connected devices (e.g. LED, motor, …).

In order to use the WLAN module as a web server, you need to connect it to an existing Wi-Fi network or, if there is none available, create one. Next, create an object from WiFiServer type, which waits for detailed inquiries on the specified port (typically 80), and sends the client the saved (HTML)-code/text.

There is a simple example sketch for that available under File > Examples > ESP8266WiFi > WiFiWebServer.