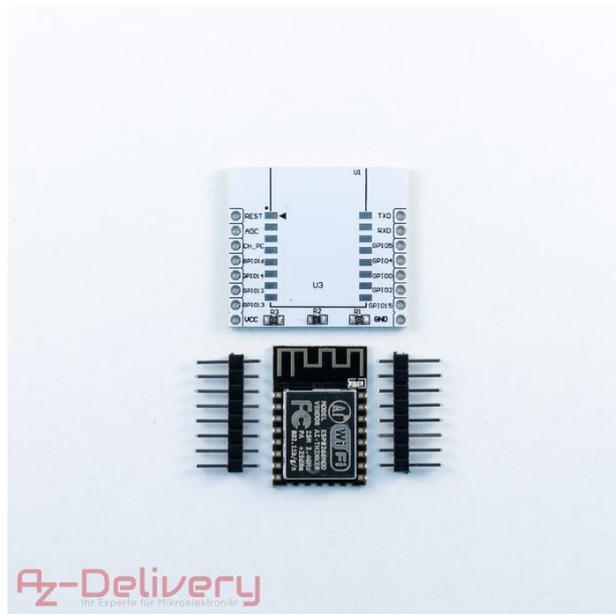


# Willkommen!

Und herzlichen Dank für den Kauf unseres AZ-Delivery ESP8266-12E mit Adapter Board. Auf den folgenden Seiten gehen wir mit dir gemeinsam das Auflöten des ESP8266 Chips auf das Adapter Board durch.

Viel Spaß!



Wichtig zu erwähnen ist, dass es sich hier um ein SMD Bauteil handelt und für Lötanfänger nicht geeignet ist!

Bevor wir zum Löten beginnen, überprüfen wir den Lieferumfang:

1x ESP8266-12E

1x Adapter Board (3 Widerstände vormontiert)

2x Stiftleiste (8 Pin)

Ist alles geliefert worden, bereiten wir unser Material vor. Benötigt werden noch:

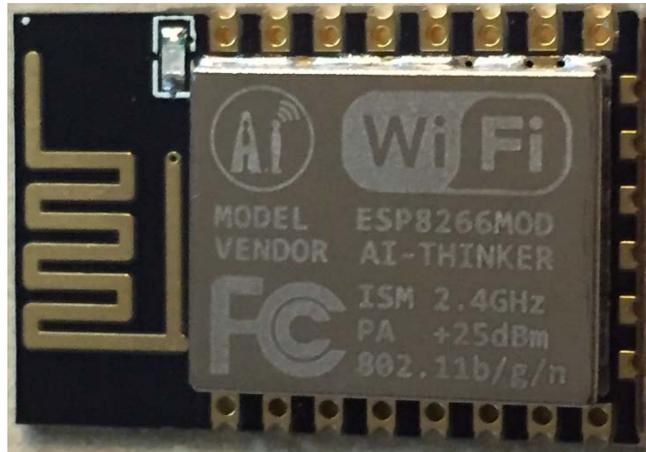
LötKolben mit feiner Spitze

Lötzinn für Elektronik (Flussmittel im Kern)

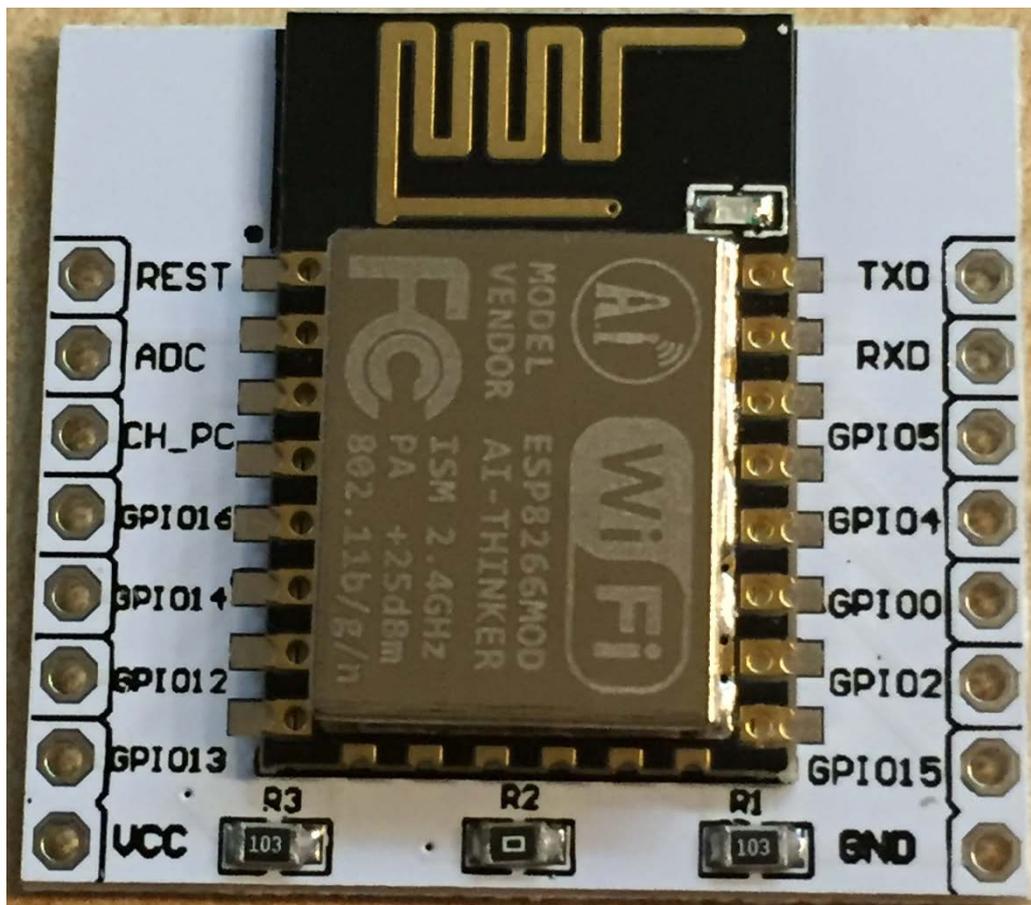
Evtl. Entlötlitze

## Löten der Platine:

Nachdem alles vorbereitet wurde, nehmen wir den ESP8266 aus der Verpackung



und legen ihn zur Kontrolle auf die Platine:

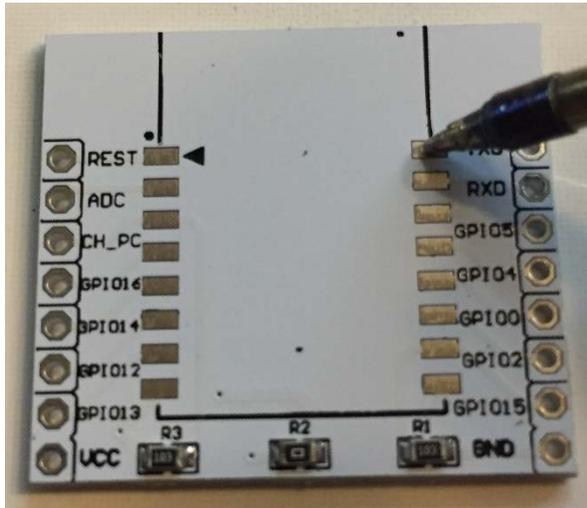


Die Kontakte sollten wie am Bild zu erkennen ist, passgenau übereinanderliegen.

Wir haben die richtige Platine mit dem korrekten Chip.

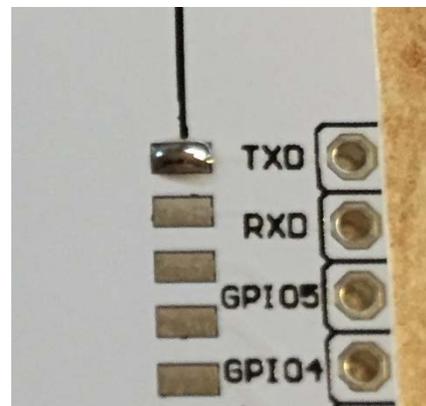
Nun nehmen wir den ESP-8266 wieder herunter und heizen unseren LötKolben auf. Je nach verwendetem Lötzinn sollte die Temperatur zwischen 300 - 330°C (bleihaltigem Lötzinn) oder 350 - 370°C (bleifreiem Lötzinn) liegen. Ist der LötKolben zu heiß, kann das Flussmittel zu schnell verdampfen und das Bauteil (ESP8266-Chip) zu heiß werden und an einem Hitzetod sterben. Auch bei niedriger und richtiger Temperatur heißt es deswegen schnell und zügig löten!

Hat unser LötKolben die Arbeitstemperatur erreicht,

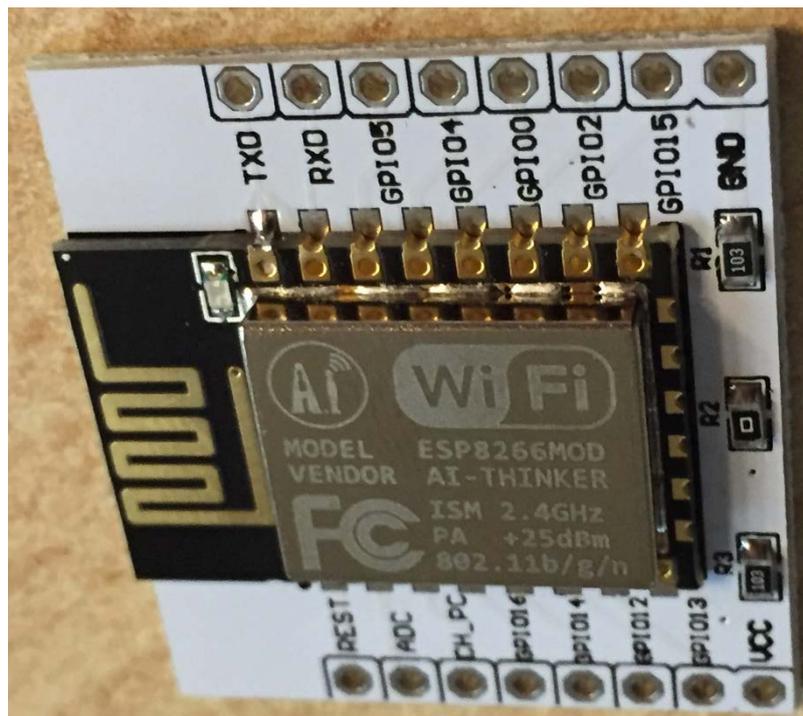


verzinnen wir eines der 16 LötPads auf der Platine, indem wir zuerst das Pad etwas erhitzen und dann etwas Lötzinn zuführen.

Das Ergebnis sollte so aussehen:

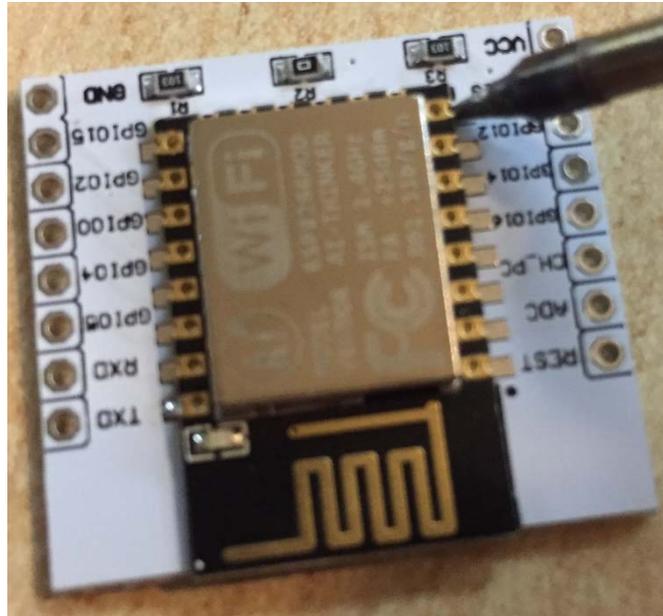


Anschließend legen wir den ESP8266 Chip auf die Platine und erhitzen unser verzinntes Pad noch einmal kurz. Mit leichtem Druck von oben auf den Chip, fixieren wir die Position und lassen den ESP8266 satt auf der Platine aufliegen.



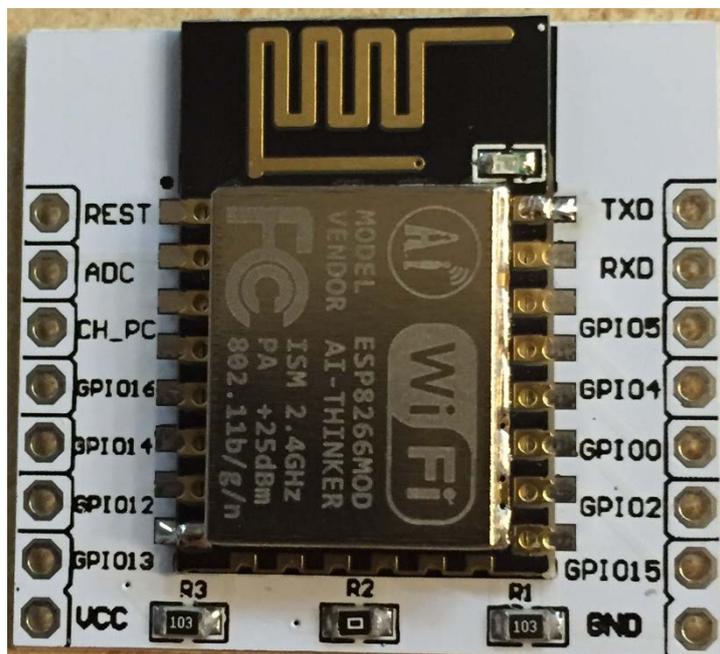
Jetzt kann eine kleine Positionsänderung durch Erhitzen der Lötstelle vorgenommen werden, aber Vorsicht: Nicht zu lange erhitzen -> Hitzetod.

Ist der Chip an seiner vorgesehenen Stelle angeheftet, so fixieren wir ihn, indem wir den gegenüberliegenden Kontakt verlöten.

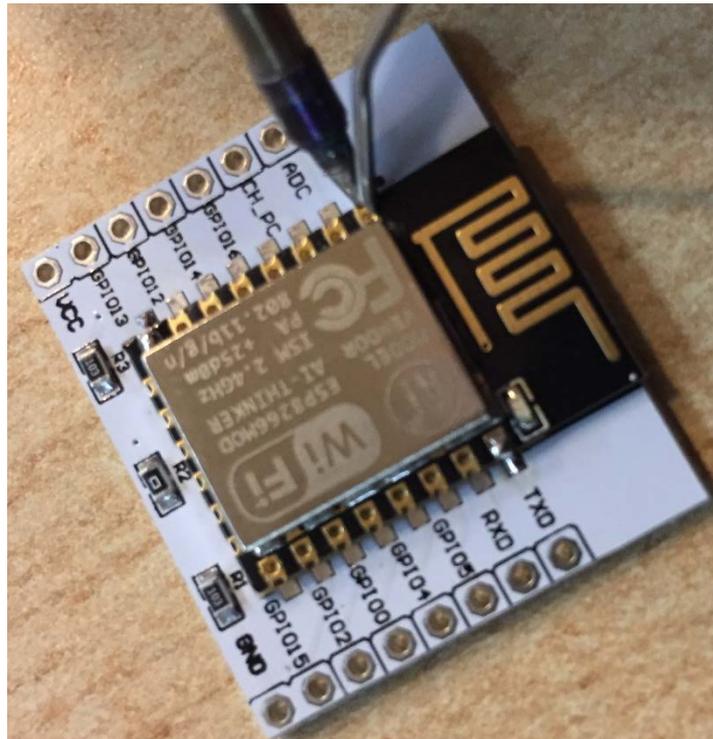


Zuerst wird der Kontakt und das Pad gleichzeitig erwärmt und kurz darauf etwas Lötzinn zugeführt. Bitte darauf achten, nicht zu viel Lötzinn zuzuführen. Es können dann Lotbrücken entstehen. Ist dies passiert, kann die Lotbrücke mit einer Entlötlitze wieder entfernt werden. Dazu einfach die Entlötlitze auf die Lotbrücke legen und mit dem LötKolben erhitzen. Die Entlötlitze saugt das Lötzinn auf. Eventuell mehrmals wiederholen.

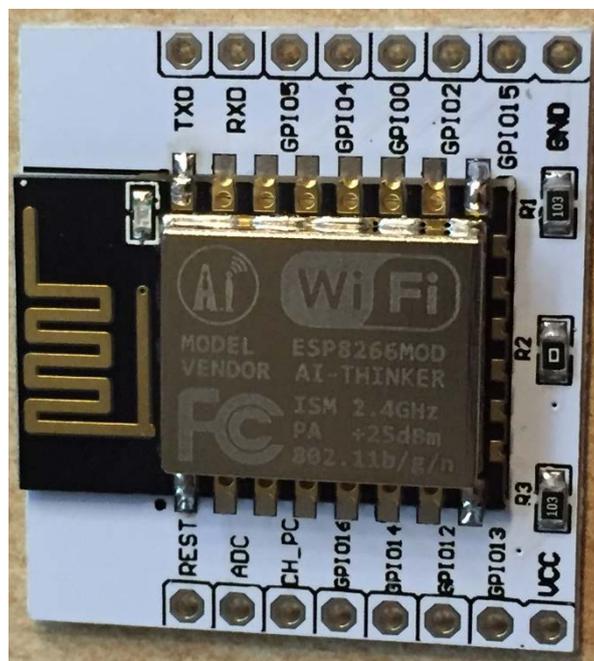
So sehen nun die 2 Lötstellen aus:



Jetzt löten wir die 2 anderen gegenüberliegenden Kontakte fest:

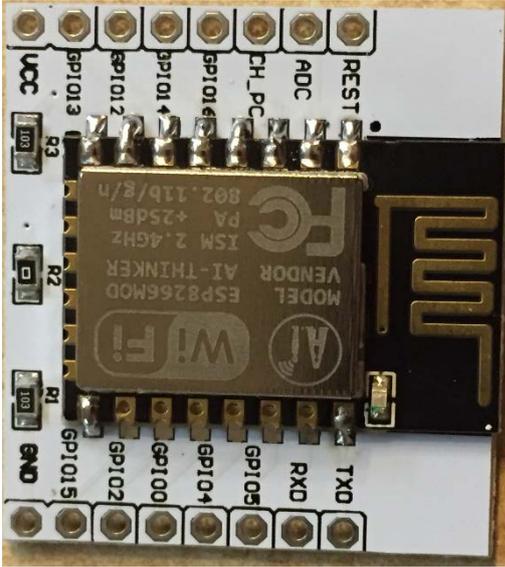
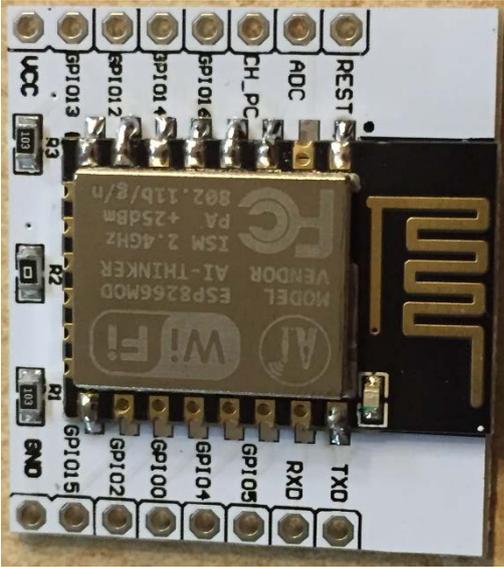
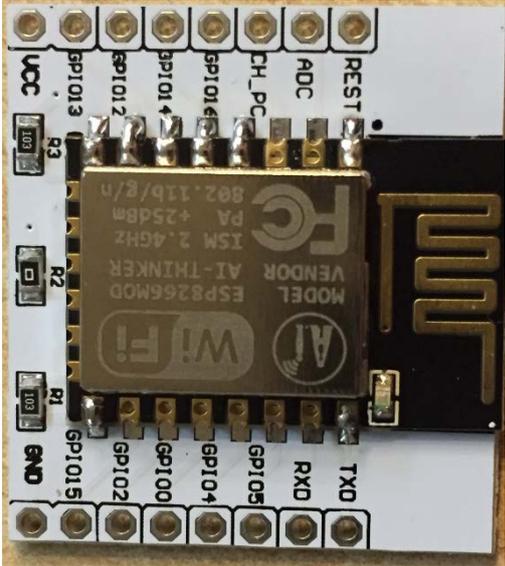
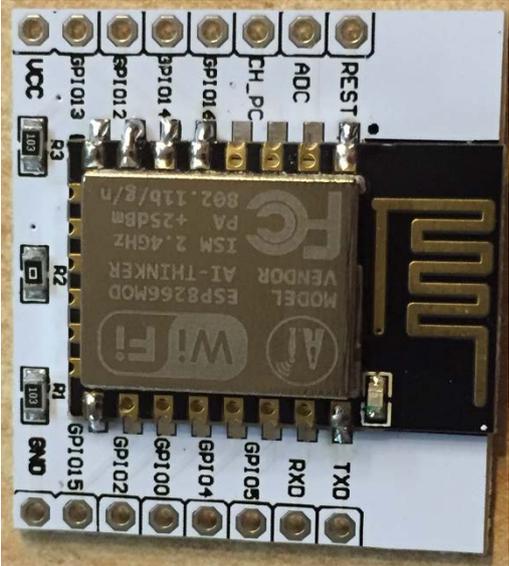
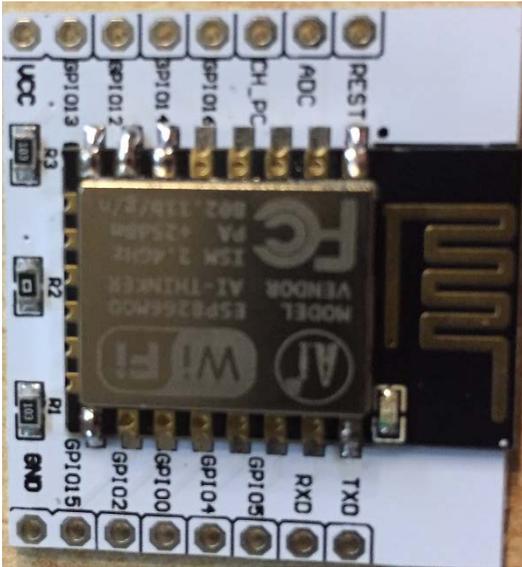
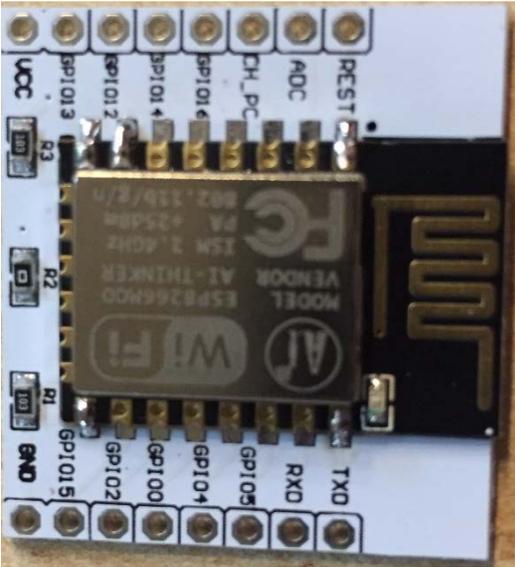


Zuerst Kontakt erhitzen und dann Lot zuführen.

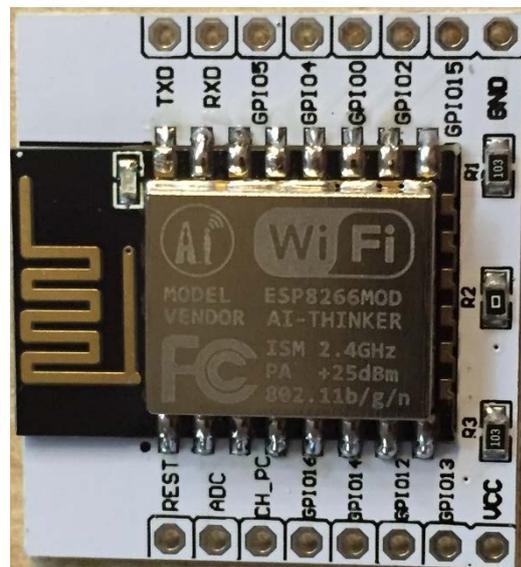
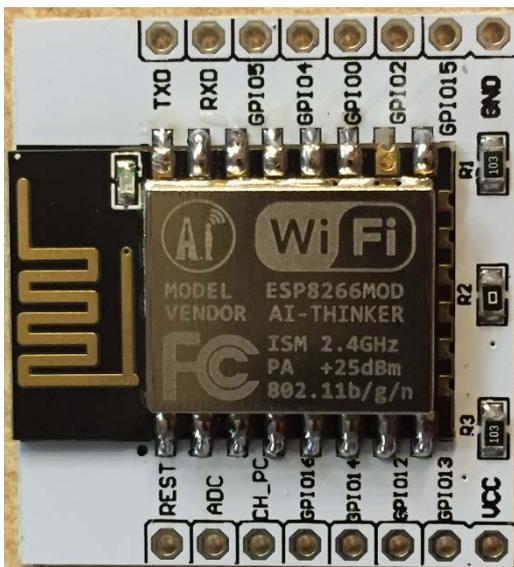
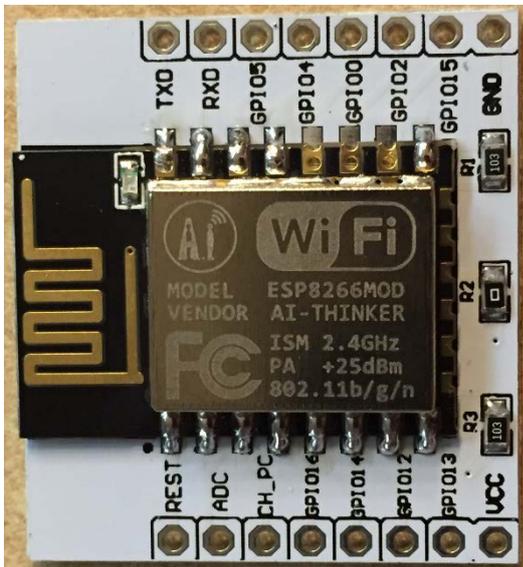
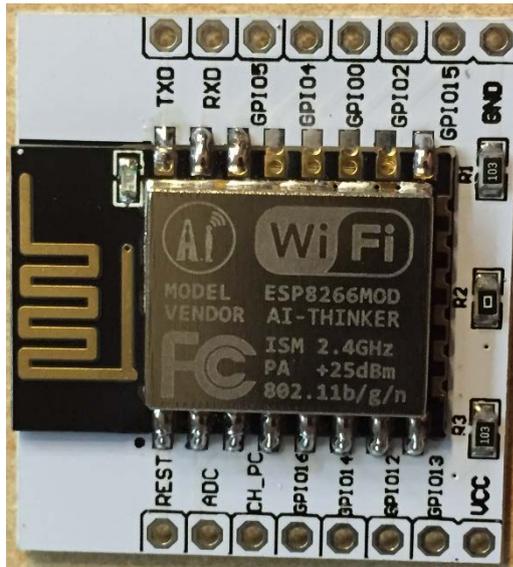
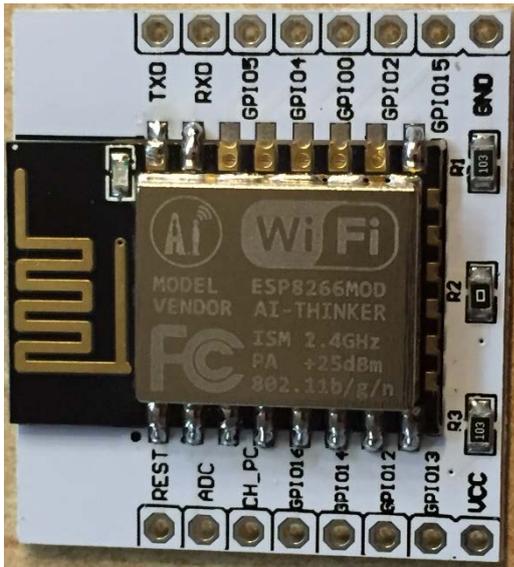


Jetzt sind vier Kontakte verlötet, der Chip sitzt nun fest und kann nicht mehr verrutschen werden.

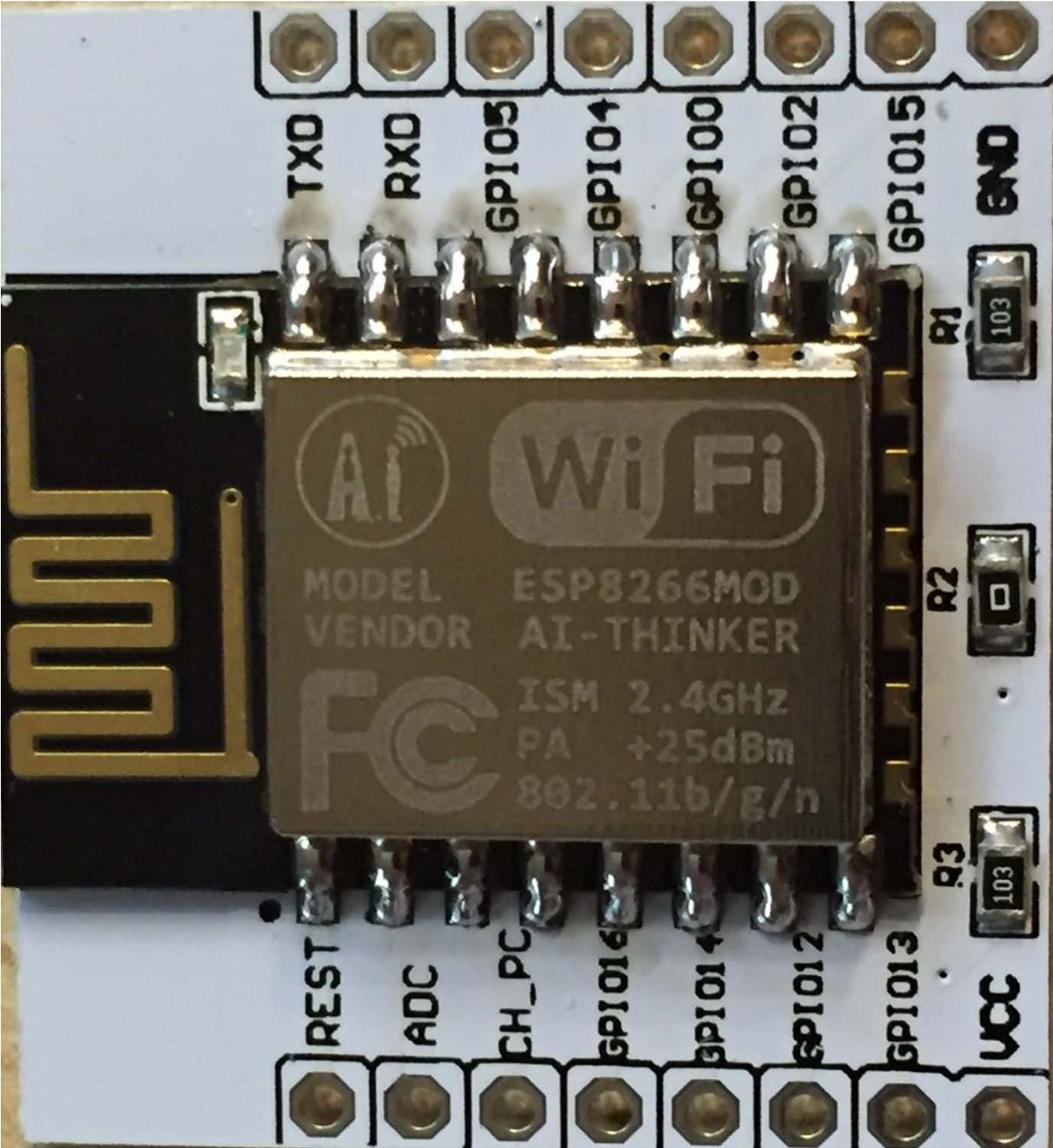
Nun kann ein Kontakt nach dem anderen auf einer Seite angelötet werden:



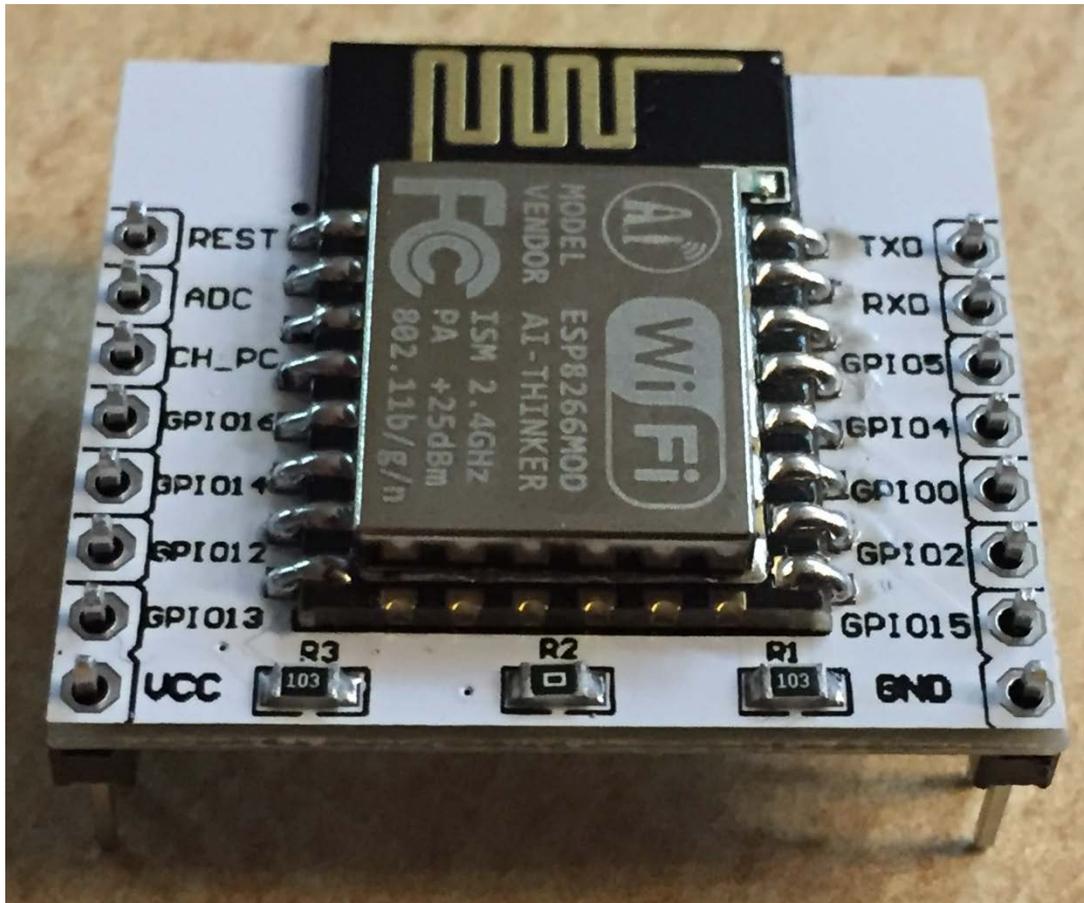
Ist nun die eine Seite fertig, machen wir mit der anderen Seite weiter:



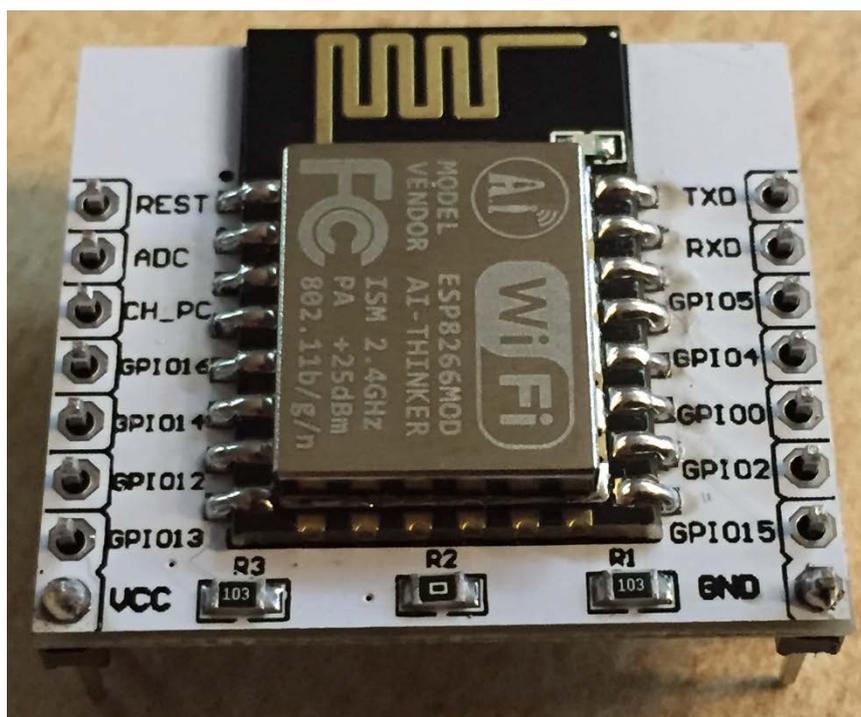
Jetzt sind alle Kontakte verlötet:



Nachdem wir alle SMD Kontakte gelötet haben, können wir die Stiftleisten auf beiden Seiten einlöten. Dazu nehmen wir die Stiftleisten zur Hand und stecken diese in die Platine:

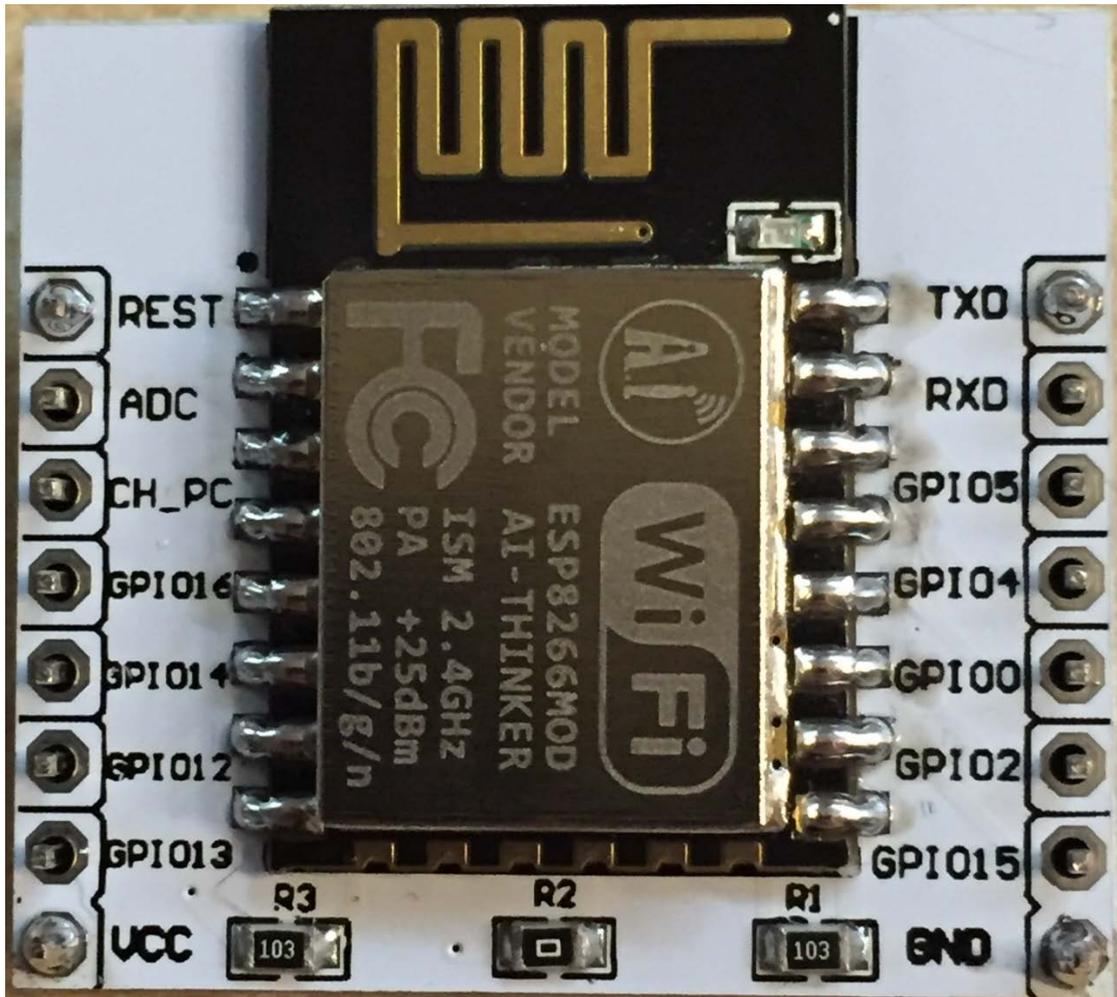


Auch hier löten wir auf jeder Seite nur je einen Pin fest:

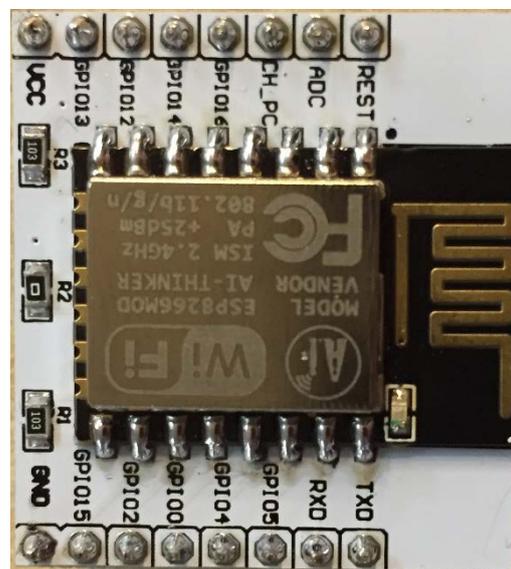
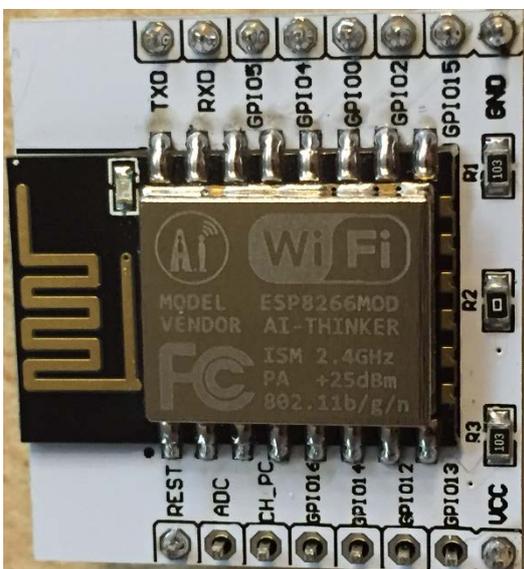


Bei Bedarf kann jetzt noch die Pinleiste etwas ausgerichtet werden.

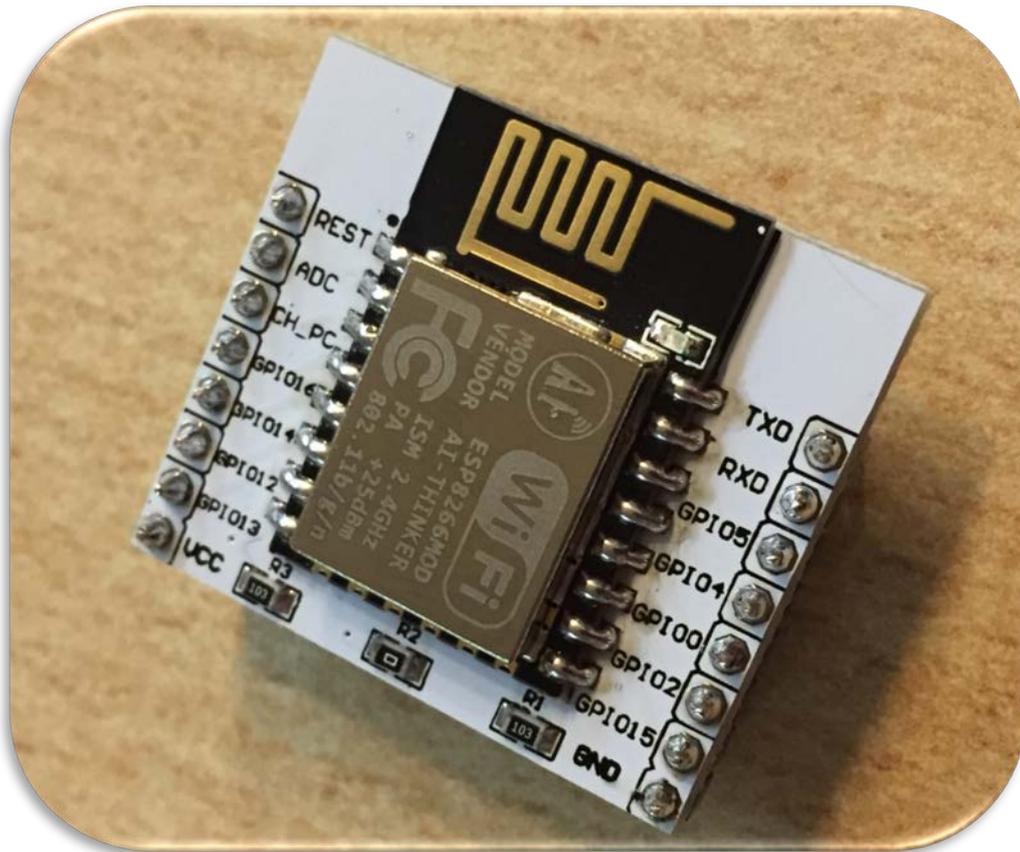
Sitzen beide Leisten richtig, löten wir den Pin auf der anderen Seite fest:



Löten wir auf jeder Seite die Pins fest:



Wir sind nun fertig mit dem Löten:



**Du hast es geschafft, jetzt geht es ans Programmieren, wie das funktioniert kannst du in den nächsten Seiten lesen.**

## **Programmieren des ESP8266-12E**

Der ESP8266-12E hat im Vergleich zu seinem kleinen Bruder (ESP8266-01) 4MB Flashspeicher und mehr GPIO Anschlüsse zur freien Verfügung.

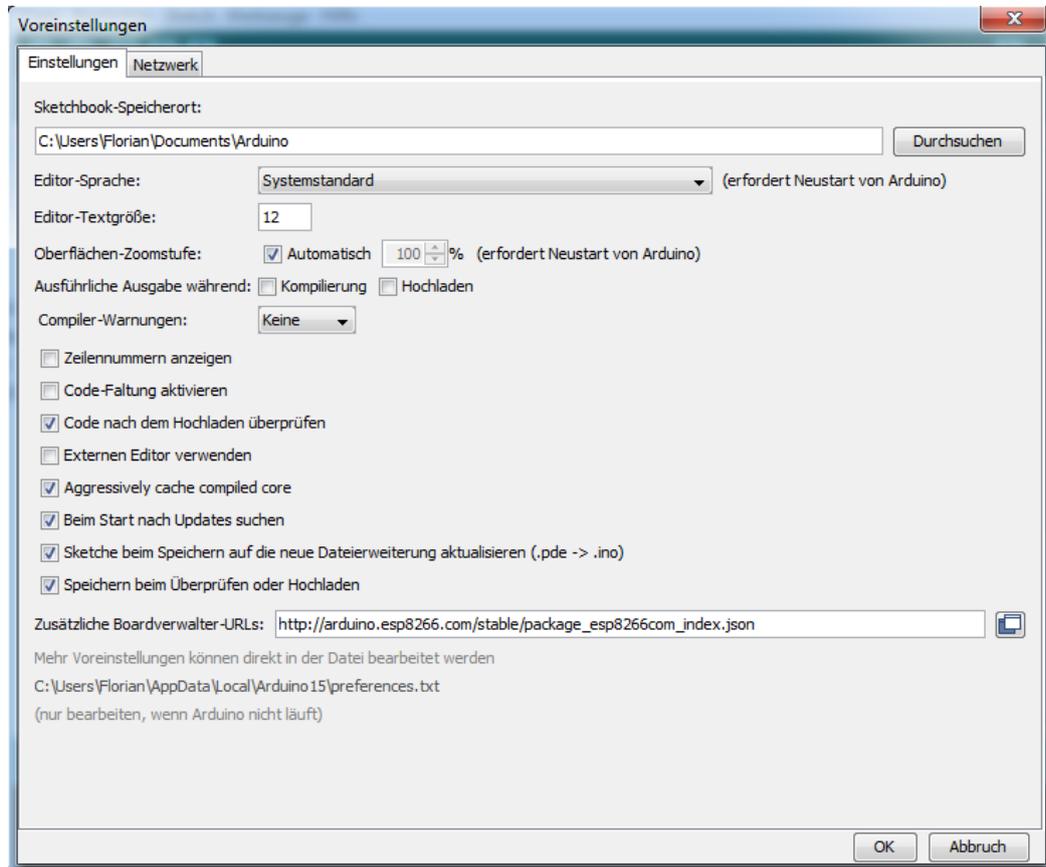
### **Vorbereiten der Software:**

Die Arduino Software sehen wir in diesem Schritt als installiert an, sollte diese bei dir noch fehlen, so kannst du diese unter <https://www.arduino.cc/en/Main/Software#> herunterladen und auf deinen PC installieren.

Außerdem die Treiber für den FT232RL solltest du auch schon installiert haben, wenn nicht, sehe dir das entsprechende ebook an.

Nachdem alle Grundvoraussetzungen getätigt wurden, starten wir nun mit der Einrichtung der Software. Die Arduino Software benötigt zunächst einmal alle Informationen zum ESP8266-12E, dies können wir tun, indem wir unter dem „Voreinstellungen“ > „Zusätzliche Boardverwalter-URLs“ folgende Adresse eingeben:

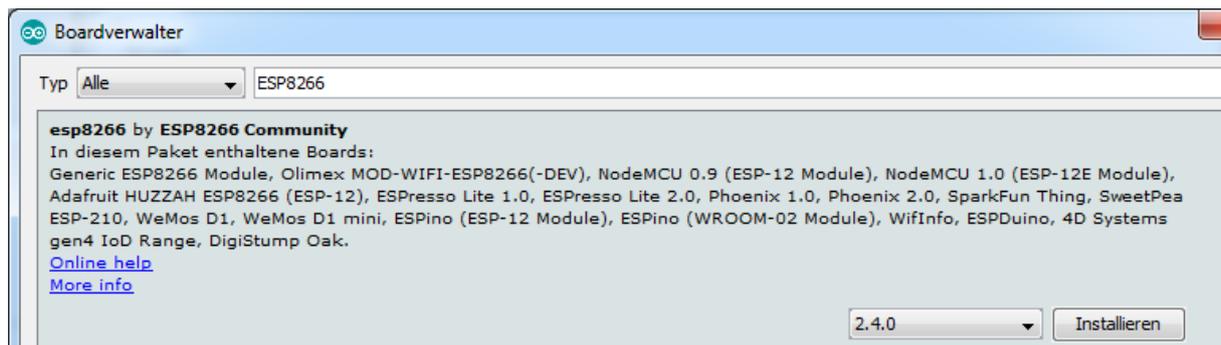
[http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)



Evtl. wenn du schon einen Link eingetragen hast, auf den Button  klicken und in dem Fenster eine neue Zeile hinzufügen.

Bestätigen wir die Eingabe mit „OK“.

Ist das erledigt, gehen wir auf „Werkzeuge“ > „Board“ > „Boardverwalter“ und installieren die ESP8266 Bibliothek. In dem Boardverwalter geben wir in der Suchleiste oben rechts „ESP8266“ ein, es wird das Paket von ESP8266 Community angezeigt. Dieses wählen wir aus und klicken auf Installieren.



Nach erfolgter Installation steht neben dem Paket nun INSTALLED.

**esp8266** by **ESP8266 Community** Version **2.4.0** **INSTALLED**  
In diesem Paket enthaltene Boards:  
Generic ESP8266 Module, Olimex MOD-WIFI-ESP8266(-DEV), NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Adafruit HUZZAH ESP8266 (ESP-12), ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, SparkFun Thing, SweetPea ESP-210, WeMos D1, WeMos D1 mini, ESPino (ESP-12 Module), ESPino (WROOM-02 Module), WifiInfo, ESPduino, 4D Systems gen4 IoD Range, DigiStump Oak.  
[Online help](#)  
[More info](#)

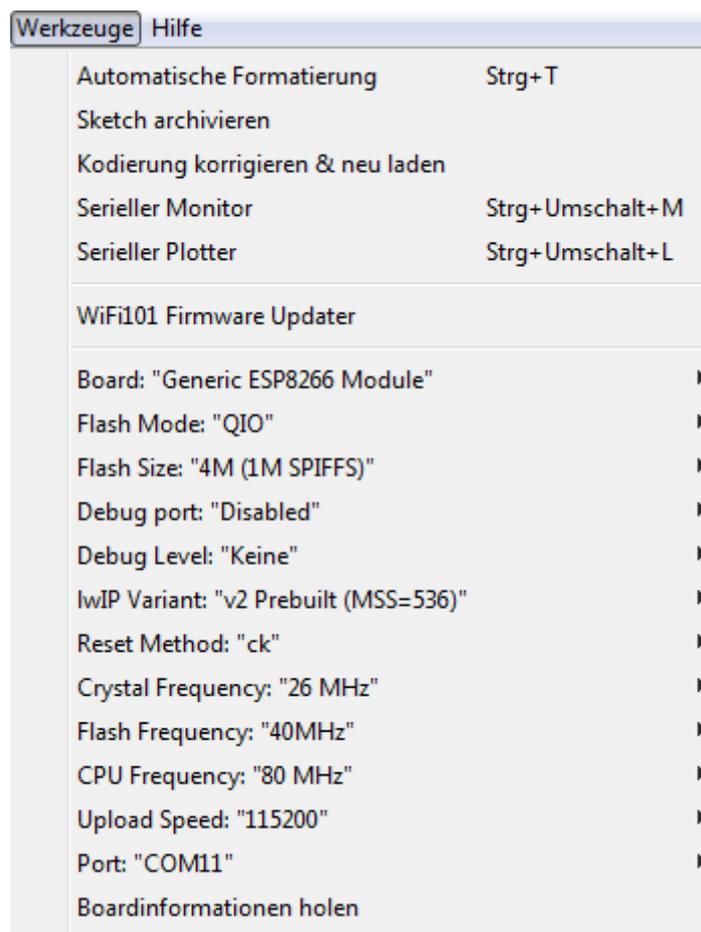
Als nächsten Schritt müssen wir das richtige Board auswählen:

Unter Werkzeuge >

Board: „Generic ESP8266 Module“

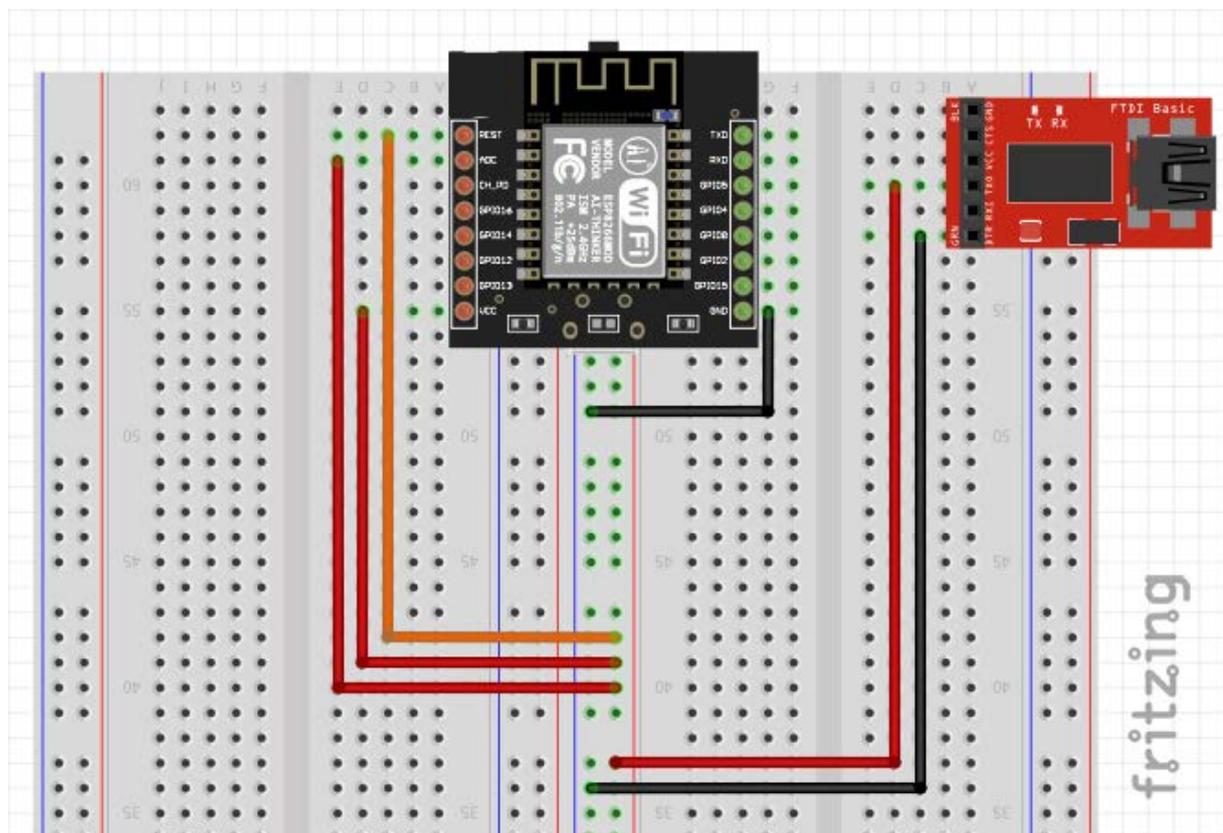
Flash Size: „4M (1M SPIFFS)“

Port: „COMxx“ (hier dein Port des Serial Adapters)



Jetzt sind alle Grundeinstellungen getätigt, jetzt geht es an die Verdrahtung.

## Verdrahten des Moduls mit dem Serial Adapter:



Die Grundbeschaltung des ESP8266-12E ist:

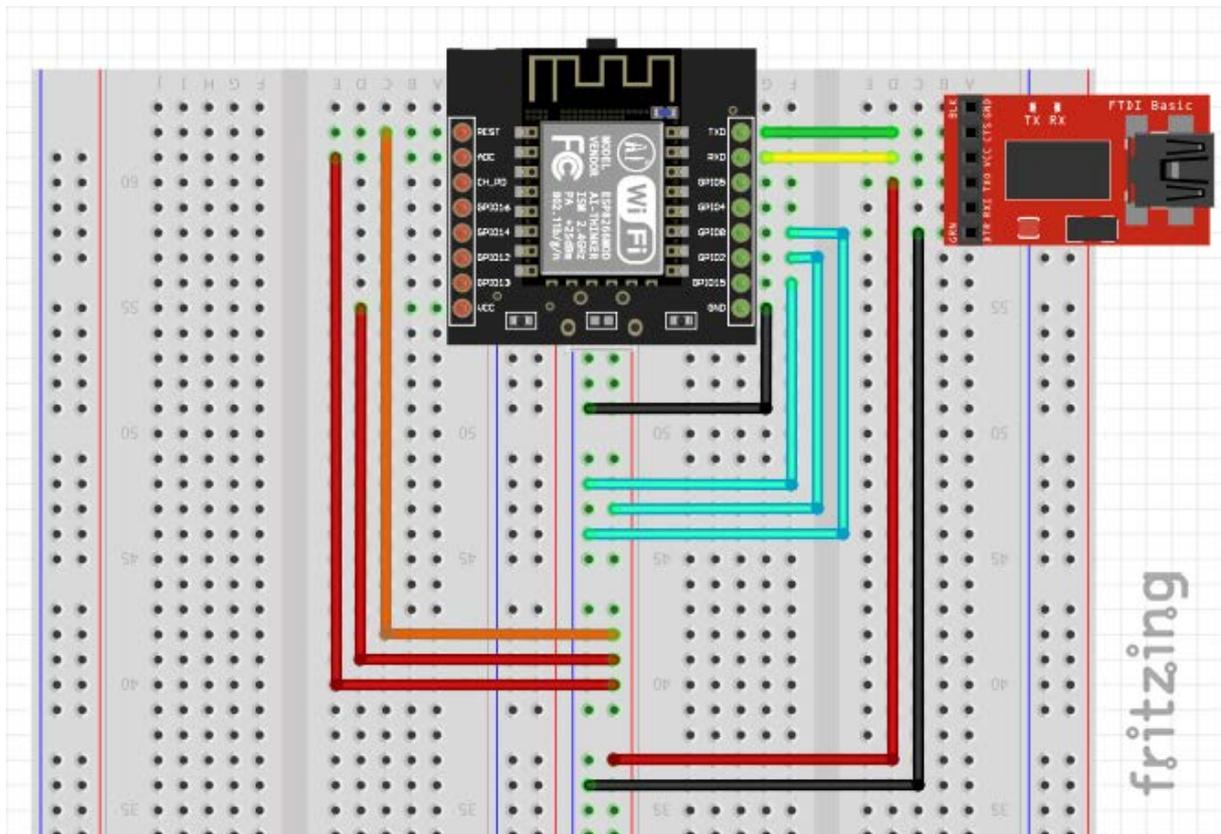
VCC mit 3,3V Spannungsversorgung verbinden	rote Leitung
GND mit Masse verbinden	schwarze Leitung
CH_PD (Chip Enable) mit 3,3V verbinden (High)	rote Leitung
Reset mit 3,3V verbinden (High)	Orange Leitung

**Wenn du den Reset auf GND (Masse) legst, dann startet der ESP8266 neu.**

**Wird der CH\_PD Pin nicht mit 3,3V bzw. auf High gelegt, ist der ESP8266 nicht aktiviert und startet somit auch nicht.**

Diese Beschaltung ist die Minimumbeschaltung um den ESP8266-12E zum laufen zu bringen und in Zukunft der Regeleinsatz des Chips.

Der Chip muss nun erst noch programmiert werden. Dazu muss der ESP8266 zuerst in den Programmiermodus gebracht werden. Dazu ergänzen wir folgende Steckbrücken:



GPIO 00	LOW	auf Masse legen	hellblaue Leitung
GPIO 02	HIGH	auf 3,3V legen	hellblaue Leitung
GPIO 15	LOW	auf Masse legen	hellblaue Leitung
RX		auf TX Serial Adapter	gelbe Leitung
TX		auf RX Serial Adapter	grüne Leitung

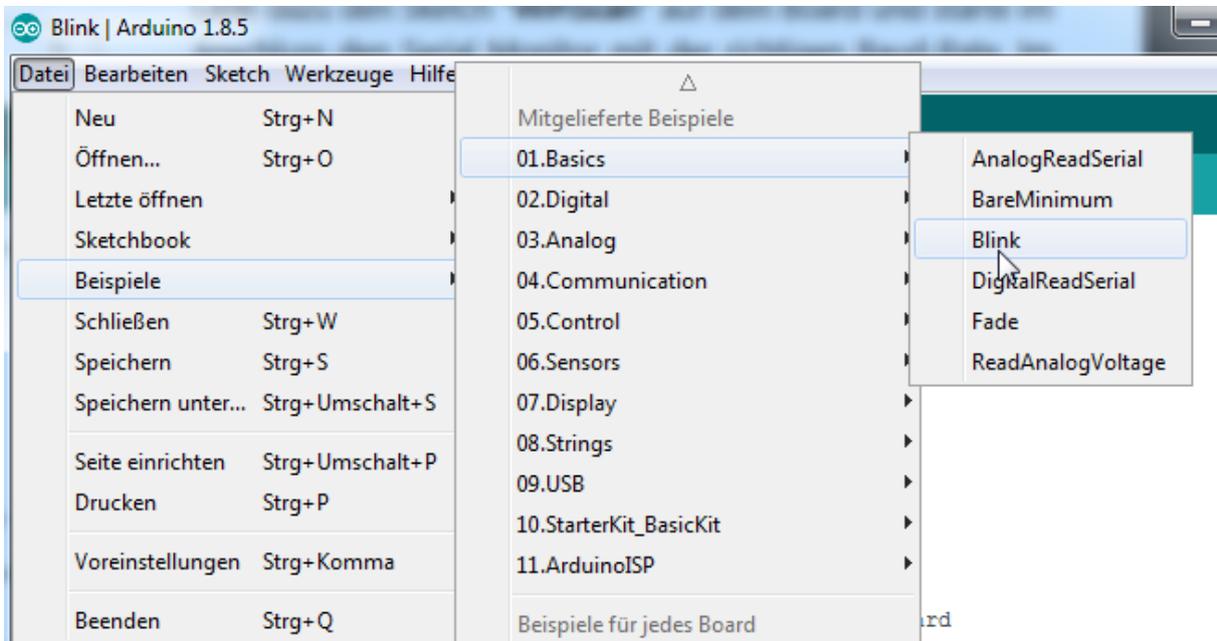
Wird der ESP8266 jetzt mit Spannung versorgt, startet er im Programmiermodus. Alternativ kann der Chip auch Resetet werden (Reset auf Masse legen) und der Chip startet im Programmiermodus neu.

Nach dem programmieren sollten die 3 hellblauen Leitungen wieder entfernt werden. (Sonst startet der ESP8266 wieder im Programmiermodus)

## Der Arduino Code:

Nachdem nun die Verdrahtung erledigt wurde, schreiben wir unseren ersten Code. Lassen wir die LED direkt auf dem ESP8266 blinken.

Wähle dazu unter Datei > Beispiele > 01.Basics > Blink aus.



Den nun angezeigten Code müssen wir noch etwas abändern:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(2, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(2, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);           // wait for a second
  digitalWrite(2, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);           // wait for a second
}
```

Und zwar funktioniert die Arduino typische Version „LED\_BUILTIN“ nicht.

Möchten wir aber trotzdem die eingebaute LED blinken lassen, so müssen wir LED\_BUILTIN mit 2 ersetzen. Der GPIO02 = LED\_BUILTIN.

Du kannst auch eine LED an jeden anderen GPIO anschließen

(GPIO15 => `digitalWrite(15, HIGH);`)

Nachdem wir den Code geändert haben klicken wir oben auf  und Verifizieren unser Programm:

Wenn alles stimmt und unser Programm keine Fehler enthält

```
Der Sketch verwendet 247055 Bytes (23%) des Programmspeicherplatzes. Das Maximum sind 1044464 Bytes.  
Globale Variablen verwenden 32868 Bytes (40%) des dynamischen Speichers, 49052 Bytes für lokale Varia
```

können wir es auf den ESP8622 hochladen. Dazu klicken wir oben auf 

Ist unser ESP8266 nicht im Programmiermodus, dann bekommen wir folgende Meldung:

```
error: espcomm_upload_mem failed  
error: Failed to open COM11  
error: espcomm_open failed  
error: espcomm_upload_mem failed  
error: espcomm_upload_mem failed
```

Dann einfach kurz Spannungsversorgung trennen und wieder verbinden oder Reset betätigen.

Dann nochmal das Hochladen  starten und es sollte nun klappen.

Nach dem Programmieren, wie schon beschrieben wurde, die hellblauen markierten Leitungen wieder trennen!

Nun blinkt die LED im Sekundentakt, aber dies ist nur ein kleiner Teil, was dein ESP8266 kann.

## **Du hast es geschafft, du kannst nun deine Projekte mit ESP8266 programmieren und verwirklichen!**

Ab jetzt heißt es Experimentieren.

Und für mehr Hardware sorgt natürlich dein Online-Shop auf:

<https://az-delivery.de>

Viel Spaß!  
Impressum

<https://az-delivery.de/pages/about-us>