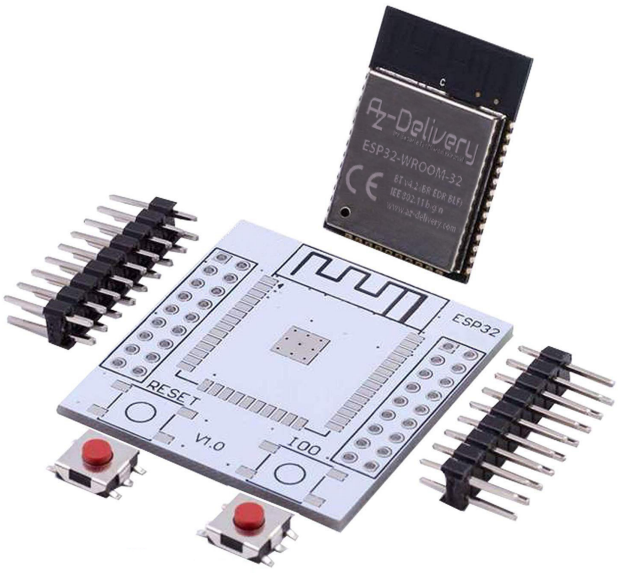
# Willkommen!

Und herzlichen Dank für den Kauf unseres AZ-Delivery ESP-32 mit Adapterkarte. Auf den folgenden Seiten gehen wir mit dir gemeinsam das Auflöten des ESP32 Chips auf die Adapterkarte durch.

Viel Spaß!




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

Bevor wir zum Löten beginnen, überprüfen wir den Lieferumfang:

1x ESP32

1x Adapter Board

2x Stiftleiste (2x9 Pin)

2x Microtaster

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

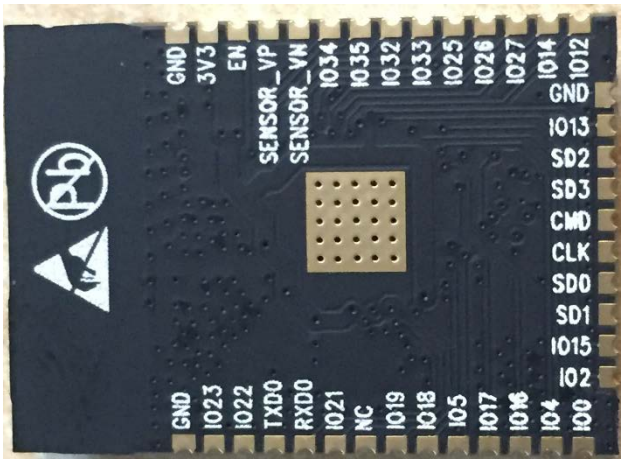
Lötkolben mit feiner Spitze

Lötzinn für Elektronik (Flussmittel im Kern)

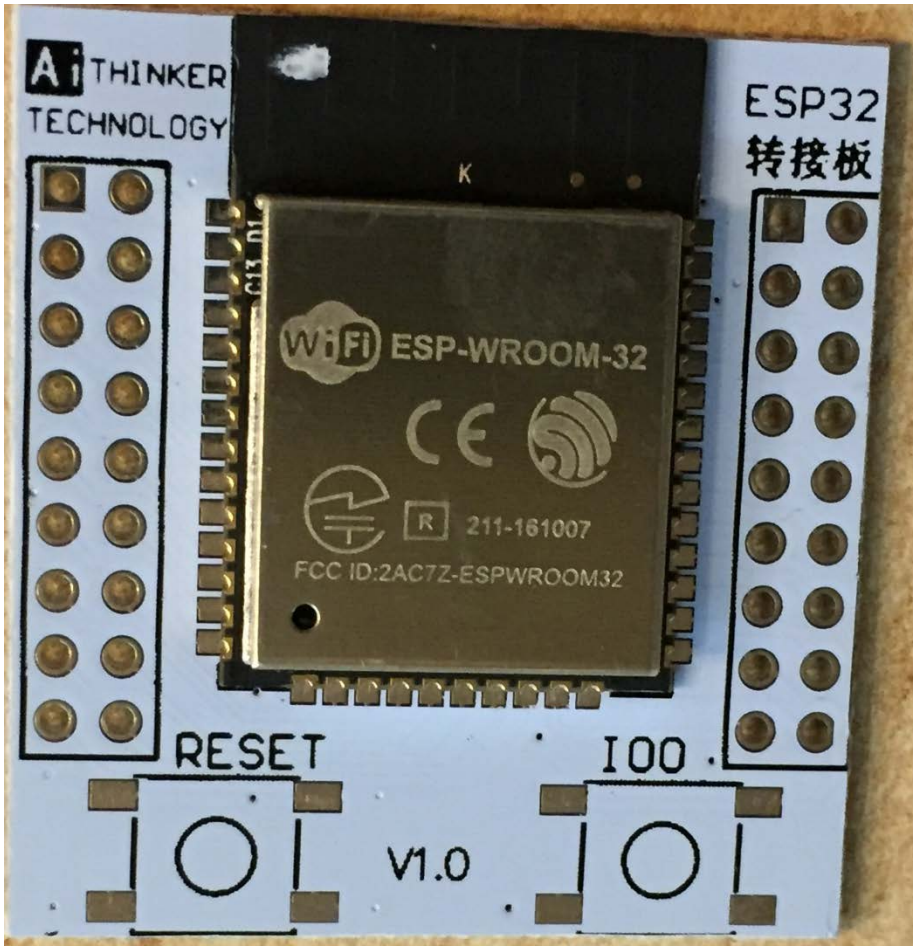
Evtl. Entlötlitze

## Löten der Platine:

Nachdem alles vorbereitet wurde, nehmen wir den ESP32 aus der Verpackung




und legen ihn zur Kontrolle auf die Platine:

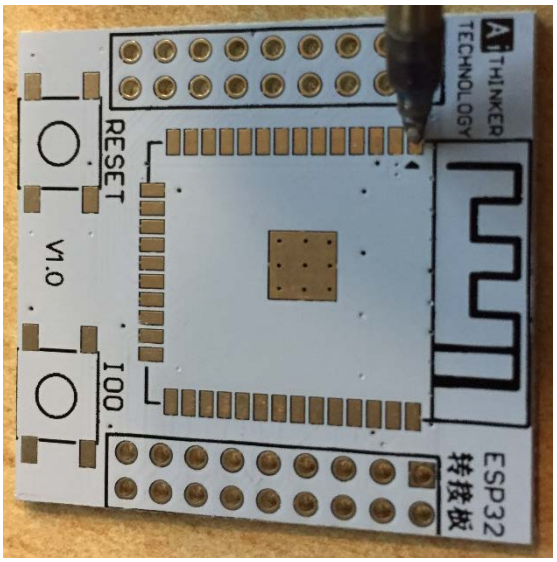



Die Kontakte sollten wie am Bild zu erkennen ist, passgenau übereinanderliegen.

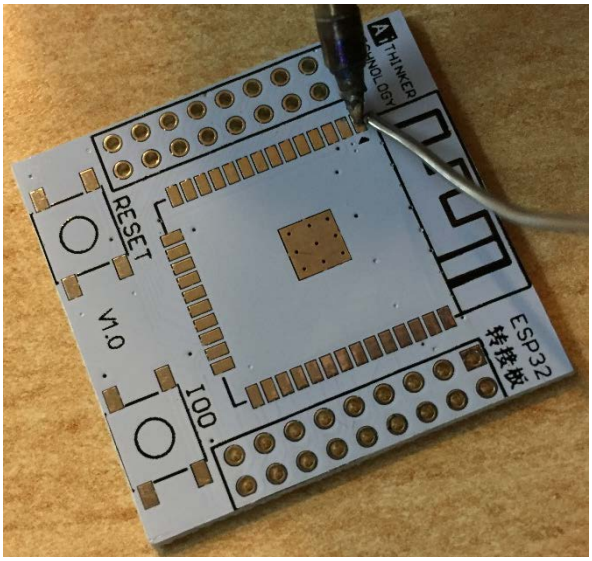
Wir haben also die richtige Platine mit dem korrekten Chip.

Nun nehmen wir den ESP32 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 (ESP32-Chip) zu heiß werden und an einem Hitzetod sterben. Auch bei niedriger und richtiger Temperatur heißt es deswegen schnell und zügig löten!

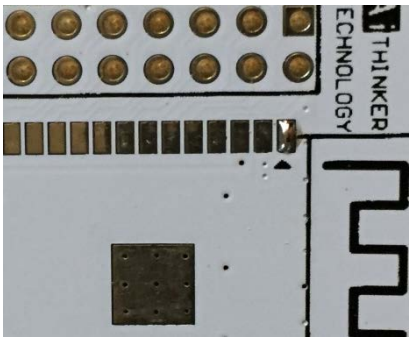
Hat unser Lötkolben die Arbeitstemperatur erreicht,




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



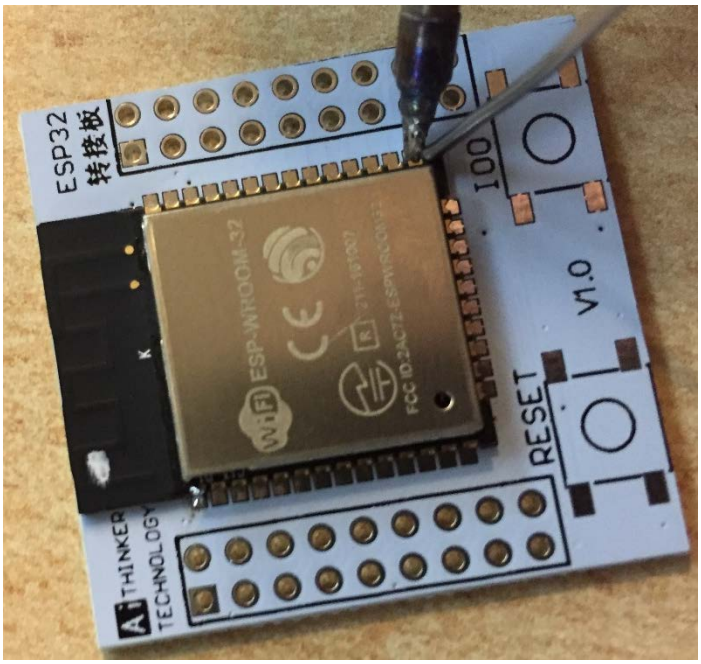

Das Ergebnis sollte so aussehen:




Anschließend legen wir den ESP32 Chip auf die Platine und erhitzen unser verzinntes Pad noch einmal kurz. Mit leichten druck von oben auf den Chip, fixieren wir die Position und lassen den ESP32 satt auf der Platine aufliegen.

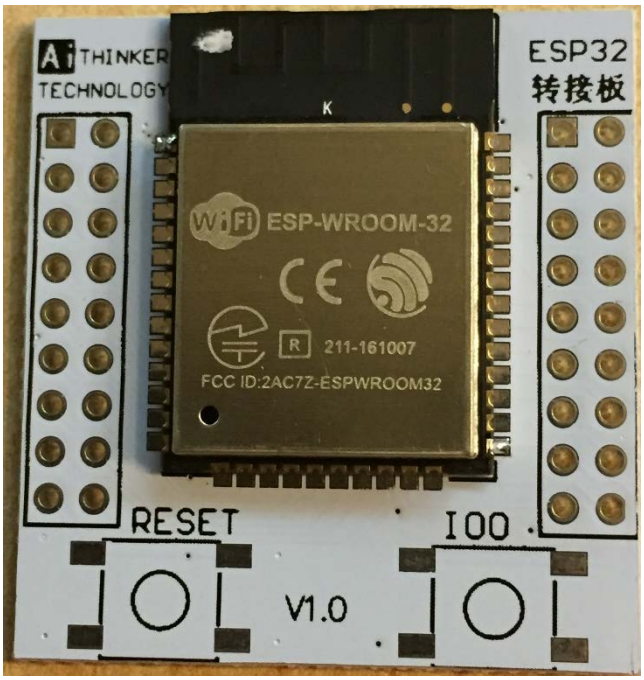
Jetzt kann noch eine kleine Positionsänderung durch erhitzen der Lötstelle vorgenommen werden, aber Vorsicht: Nicht zu lange erhitzen -> Hitzetod und Gefahr von Brückenbildung durch das Lötzinn.

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

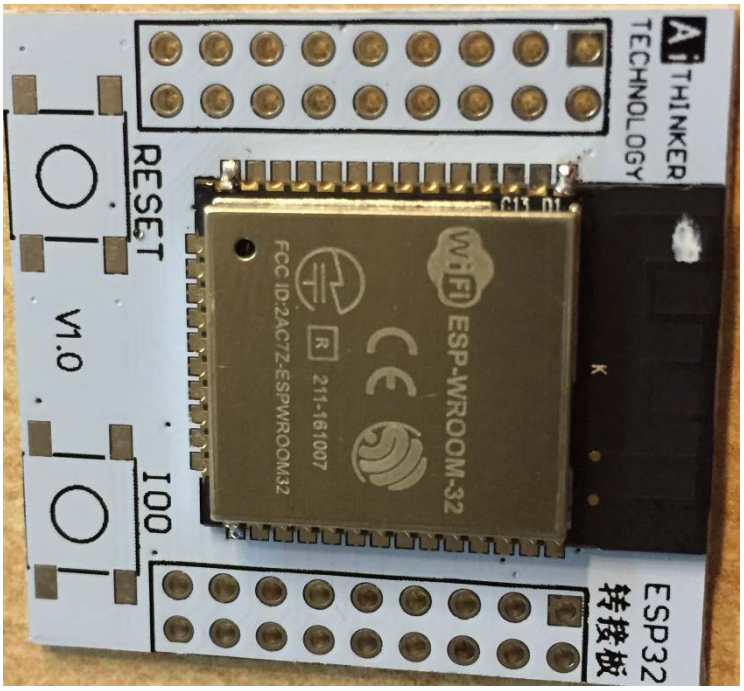



Zuerst wird der Kontakt und das Pad gleichzeitig erwärmt und kurz darauf etwas Lötzinn zugeführt. Bitte darauf achten, nicht zu viel Lötzinn zuzuführen. Es können dann Lotbrücken entstehen. Ist dies passiert, kann die Lotbrücke mit einer Entlötlitze wieder entfernt werden. Dazu einfach die Entlötlitze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

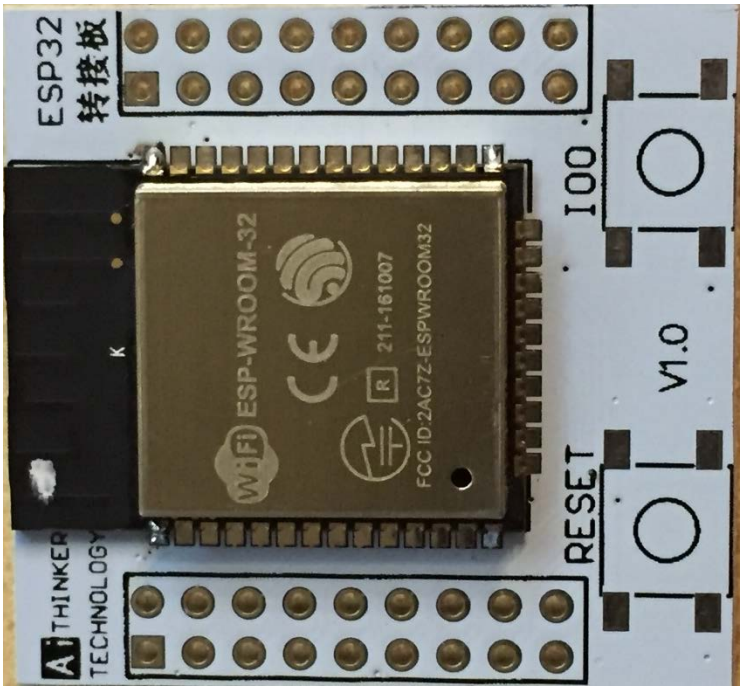
So sehen nun die 2 Lötstellen aus:



Jetzt löten wir die 2 anderen gegenüberliegenden Kontakte fest:

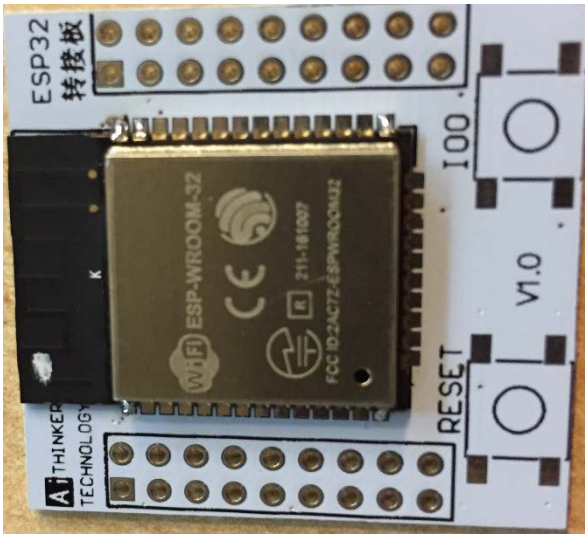


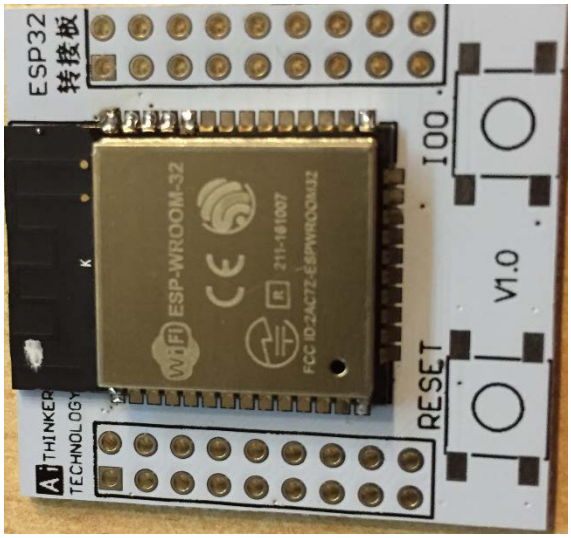

Zuerst Kontakt erhitzen und dann Lot zuführen.

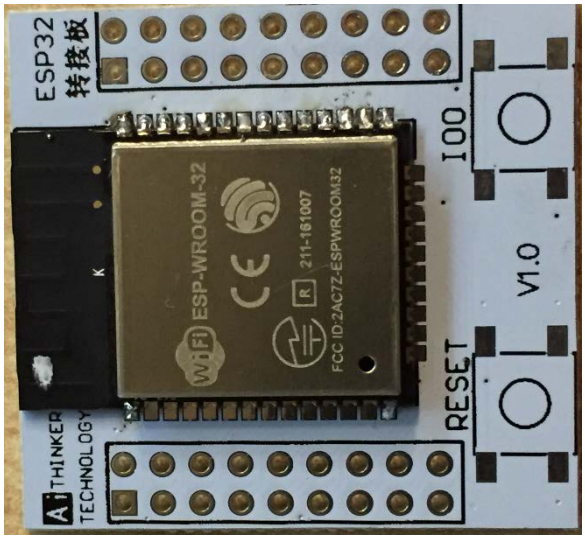



Jetzt sind vier Kontakte verlötet, der ESP32-Chip sitzt nun fest und kann nicht mehr verrutscht werden.

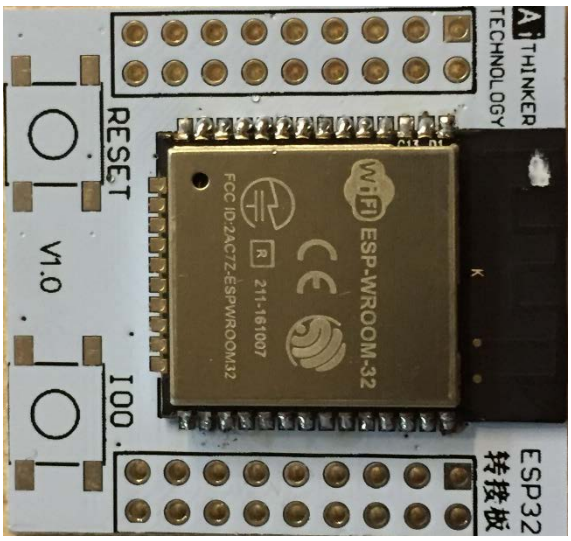
Nun kann ein Kontakt nach dem anderen auf einer Seite angelötet werden:



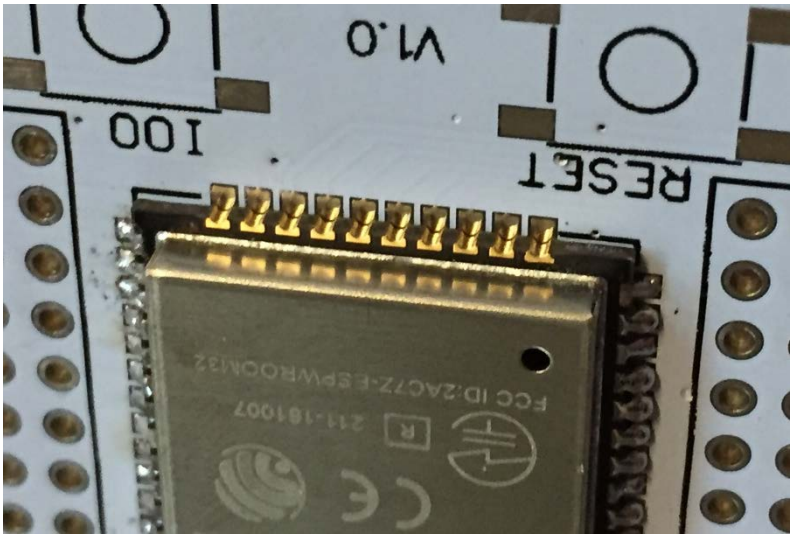



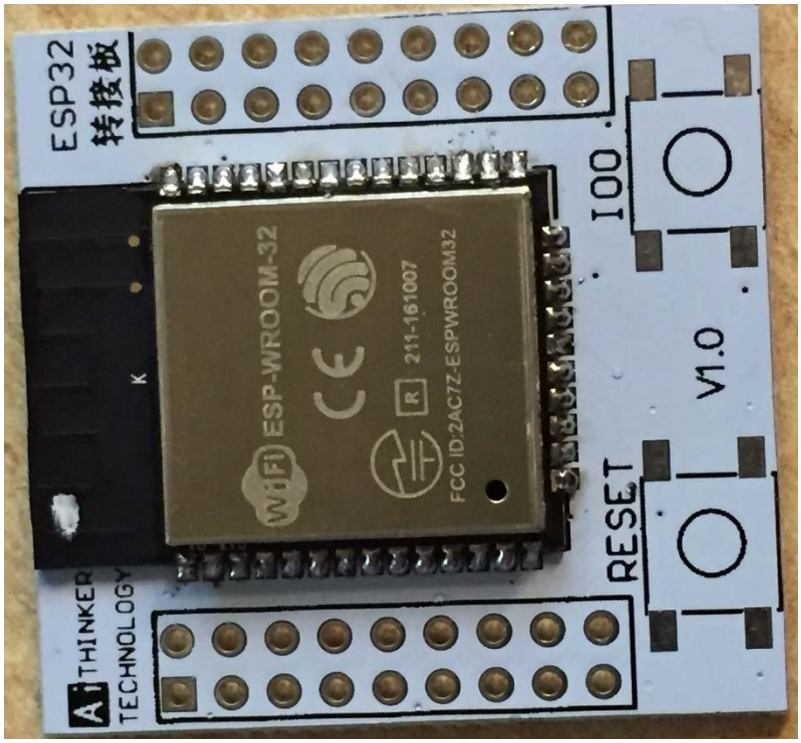

Ist nun die eine Seite fertig, machen wir mit der anderen Seite weiter:



Zuletzt am ESP-32 wird die 3. Seite, Kontakt für Kontakt, verlötet:

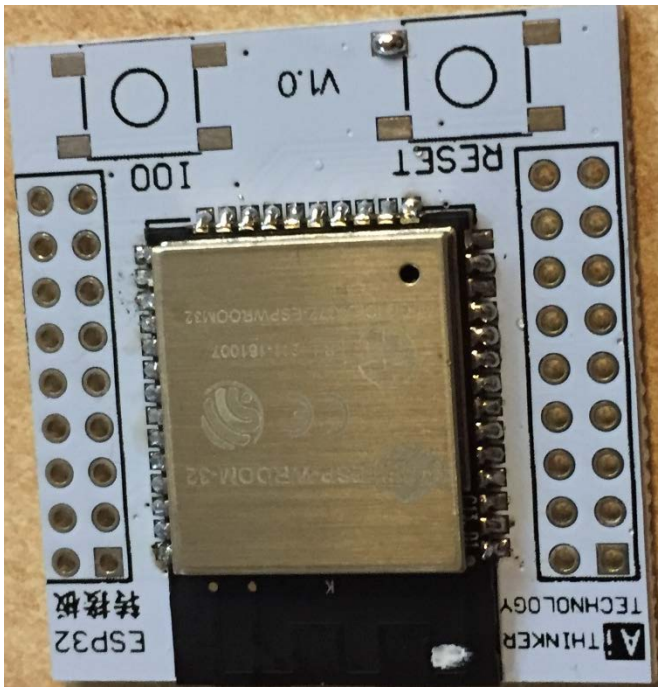



Jetzt sind alle Kontakte verlötet:

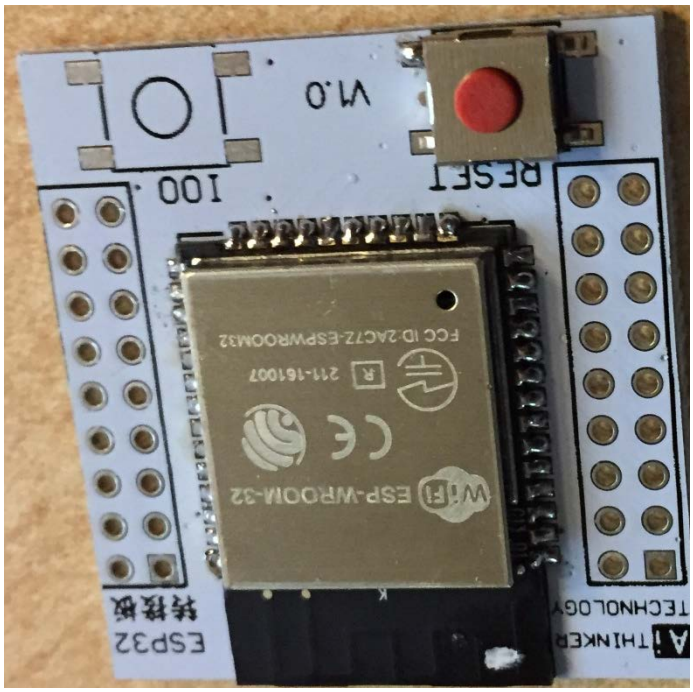


Sind alle Kontakte des ESP32-Chips verlötet, löten wir die 2 SMD Taster auf die Platine.

Auch hier verzinnen wir eines der Pads auf der Platine:



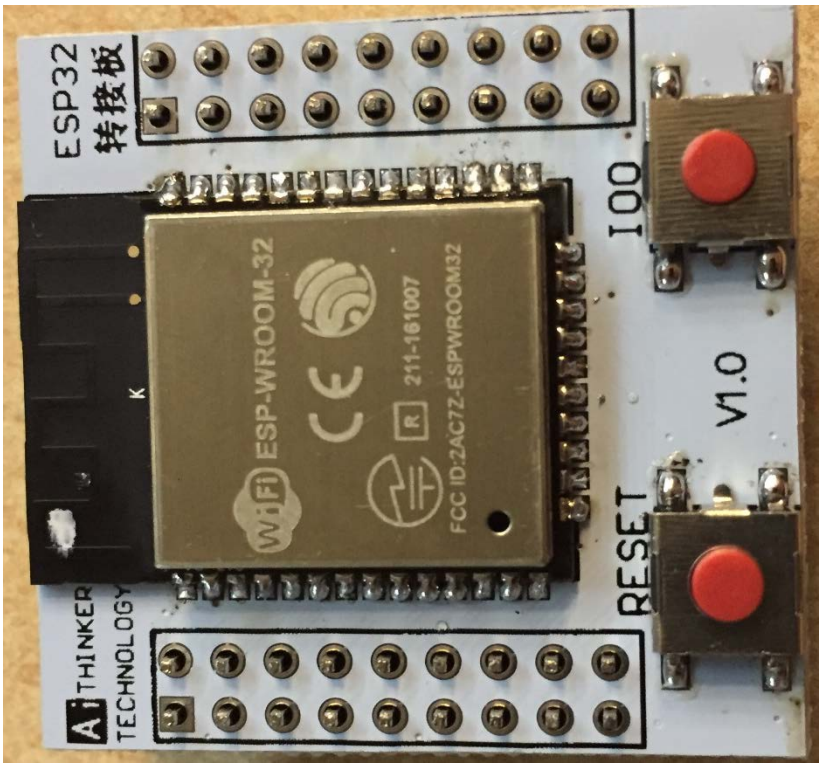

Anschließend setzen wir den Taster auf die verzinnte Stelle und fixieren ihn.



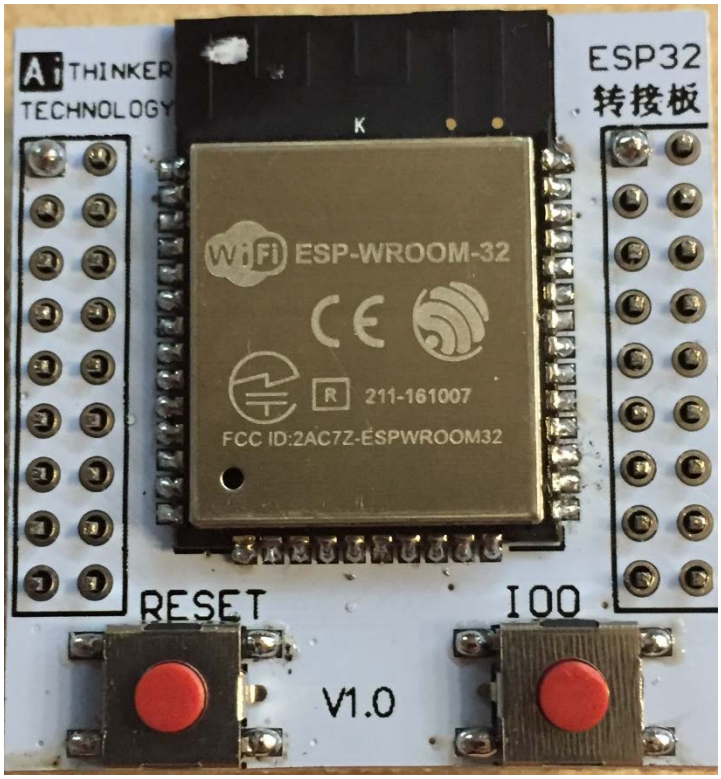

Jetzt kann wieder eine kleine Positionsänderung durchgeführt werden und auch hier gilt wieder nicht zu lange erhitzen. Auch der mechanische Taster kann Schaden

nehmen. Danach werden alle 4 Kontakte des Tasters gelötet und mit dem 2. Taster alle Schritte wiederholt.


Nachdem wir nun 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:

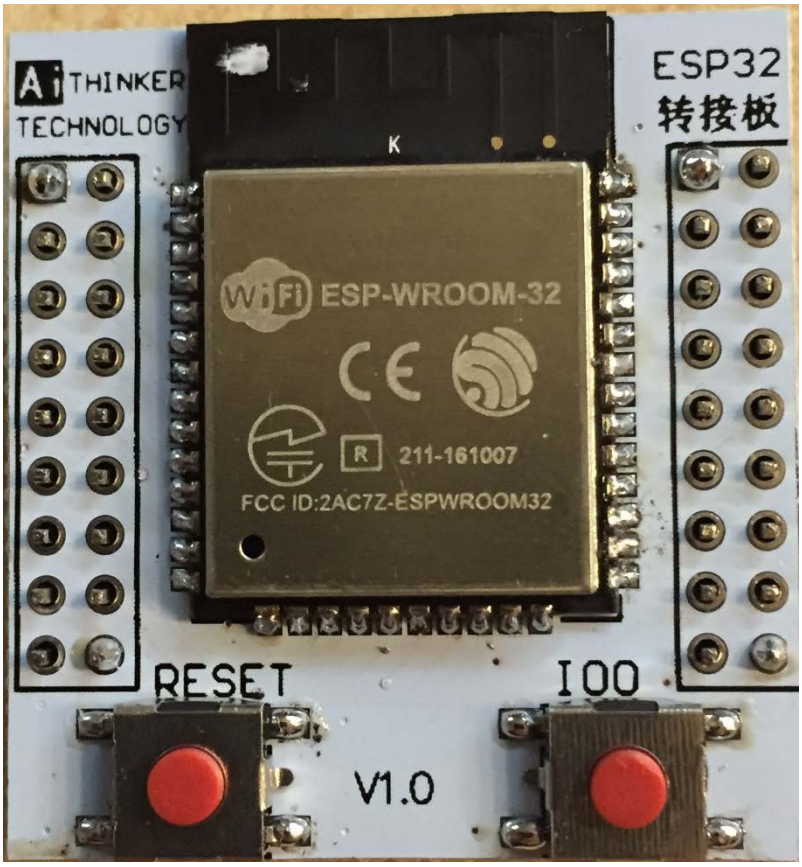


Auch hier löten wir auf jeder Seite nur je einen Pin fest:

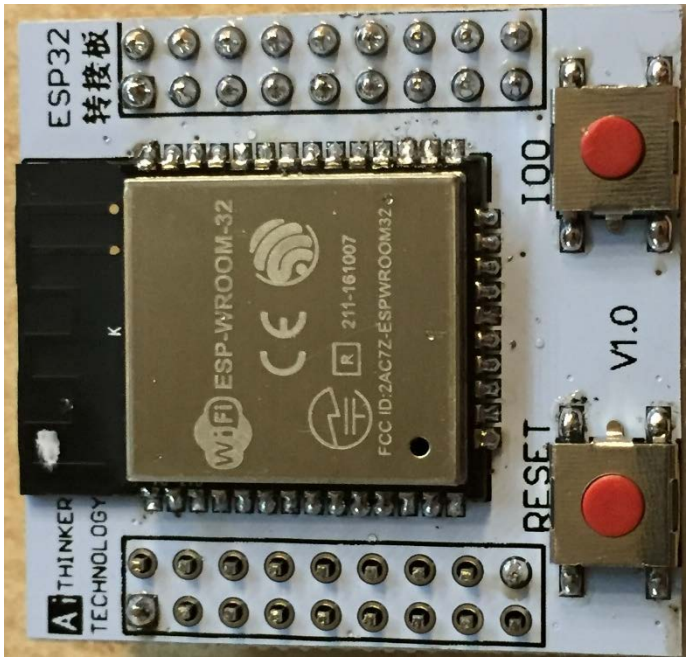


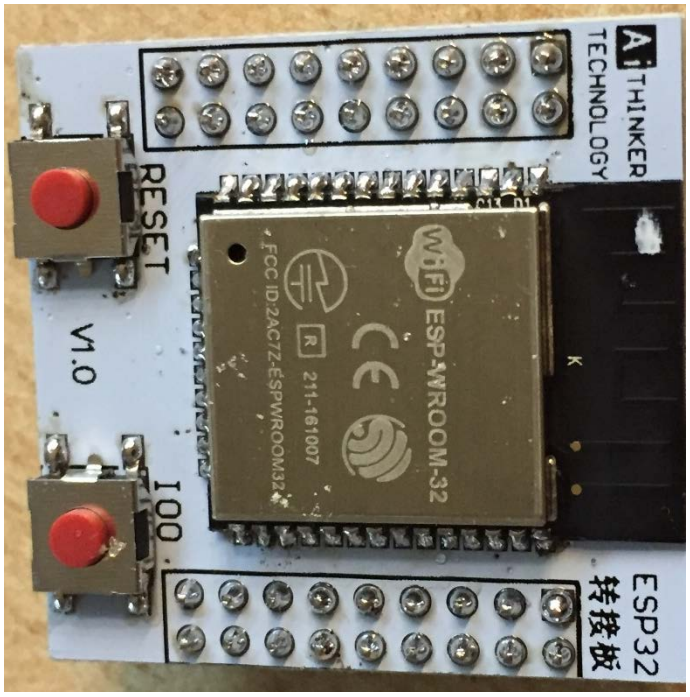

Bei Bedarf kann jetzt noch die Pinleiste etwas ausgerichtet werden.

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

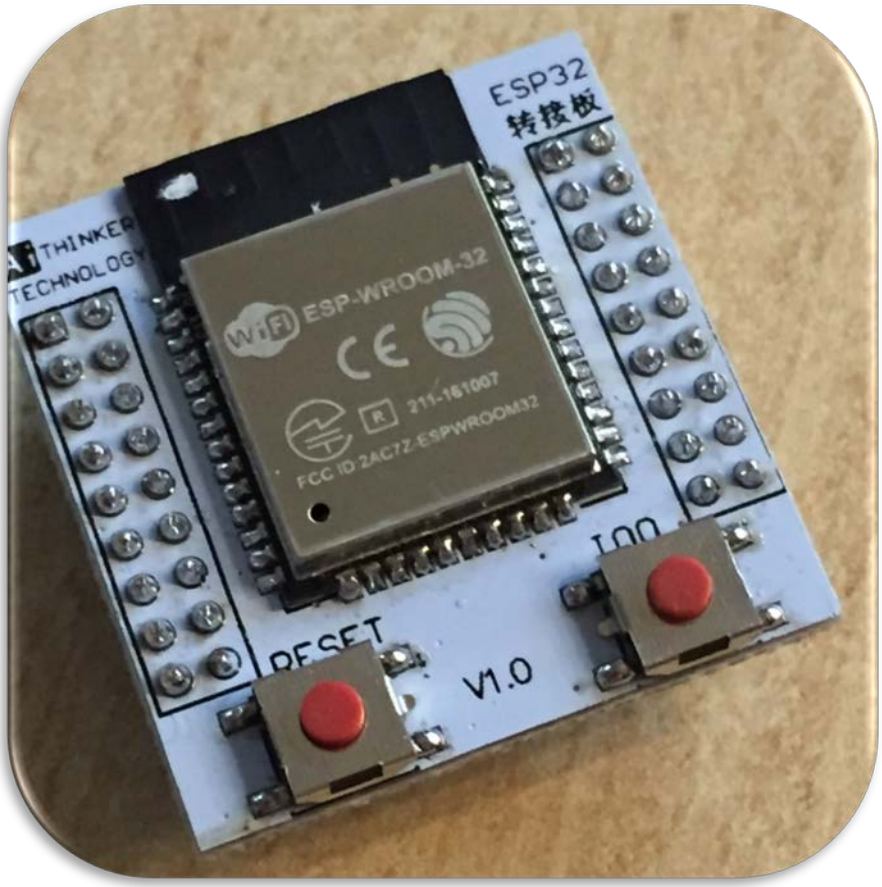


Löten wir auf jeder Seite die Pins fest:




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 ESP32

Der ESP32 hat im Vergleich zu seinem kleinen Bruder (ESP8266) mehr Leistung zur Verfügung und zusätzlich zum WLAN noch Bluetooth on Board.

## 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.

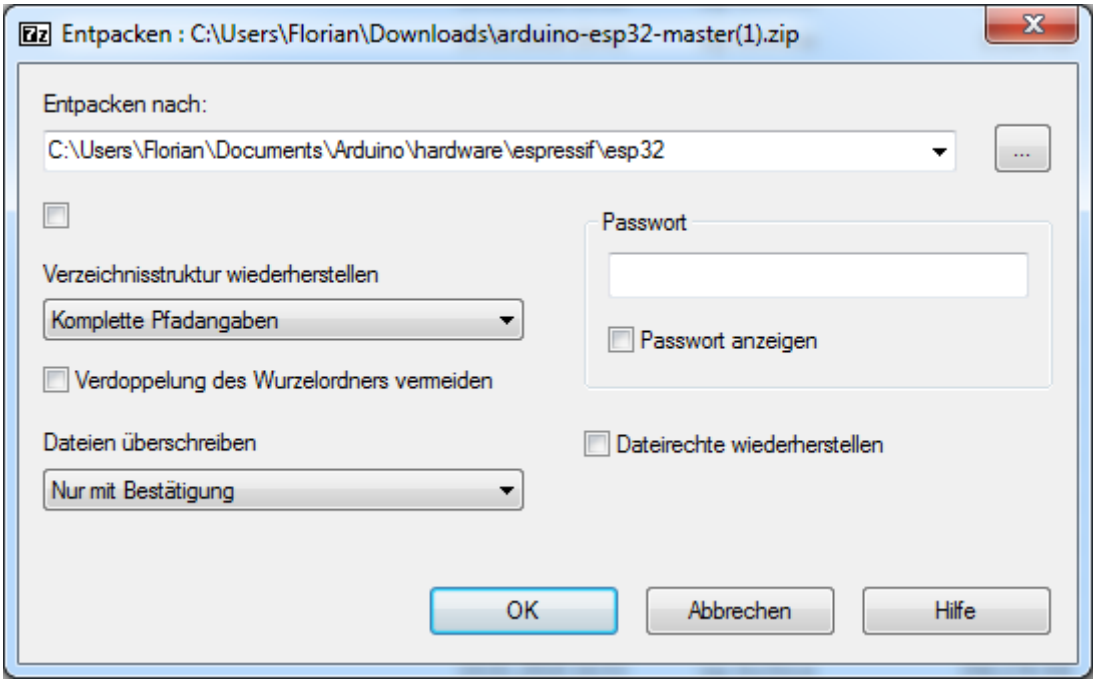
Außerdem die Treiber für den FT232RL solltest du auch schon installiert haben, wenn nicht, sehe dir das entsprechende ebook an.

Nachdem alle Grundvoraussetzungen getätigt wurden, müssen wir uns noch die benötigten Pakete für den ESP32 manuell herunterladen und in die Arduino Software einbinden. Laden wir dazu von GIT die aktuellen Daten herunter:

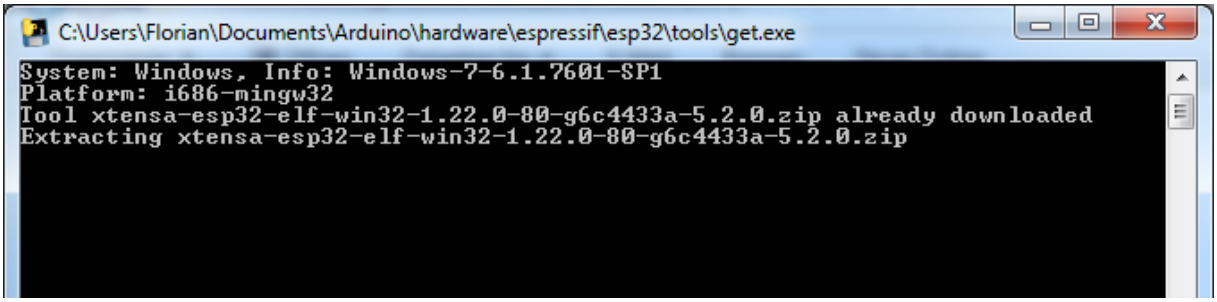
https://github.com/espressif/arduino-esp32/archive/master.zip

Diese Zip Datei entpacken (mit 7zip) wir in den Ordner: [Eigenes Userverzeichnis (C:\Benutzer\Florian\] \ Eigene Dokumente \ Arduino \ hardware \ espressif \ esp32

Hinweis: Sollten diese Ordner nicht existieren, dann lege diese einfach neu an.

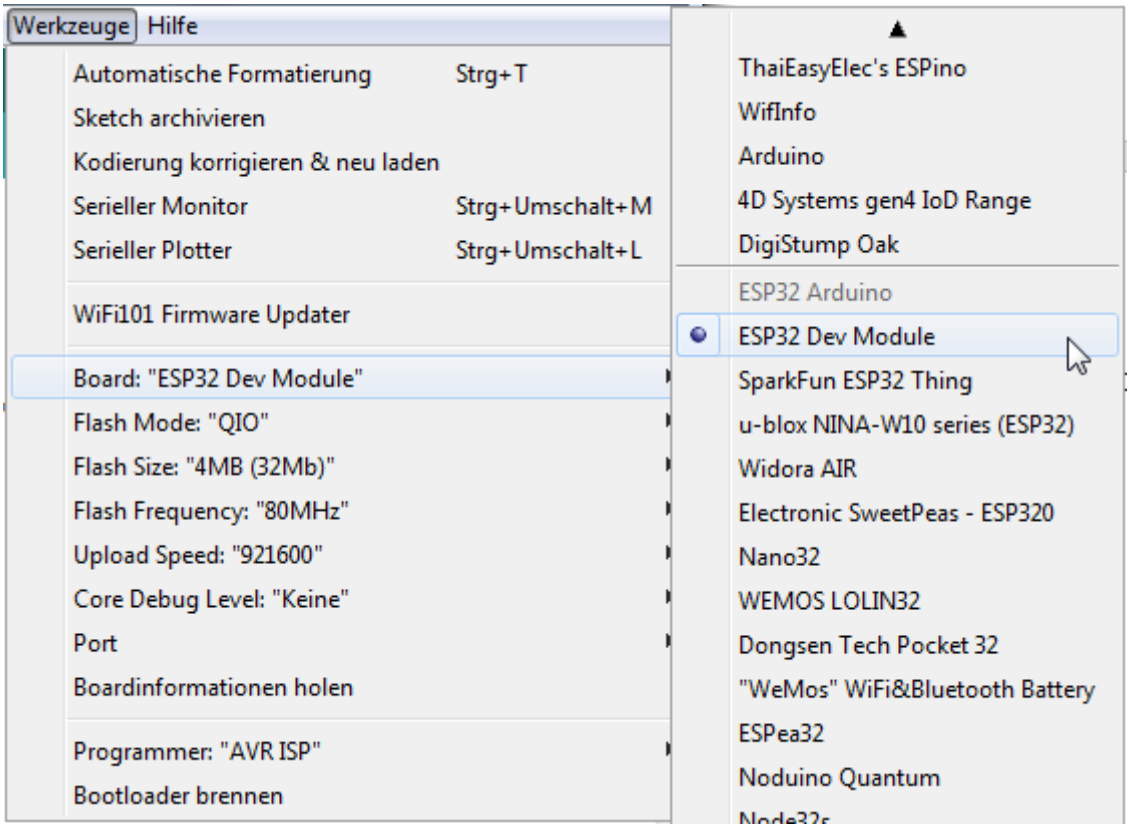



In dem Ordner tools, gibt es eine „get.exe“. Diese müssen wir einmal ausführen und alle benötigten Softwarepakete installieren und herunterladen lassen.



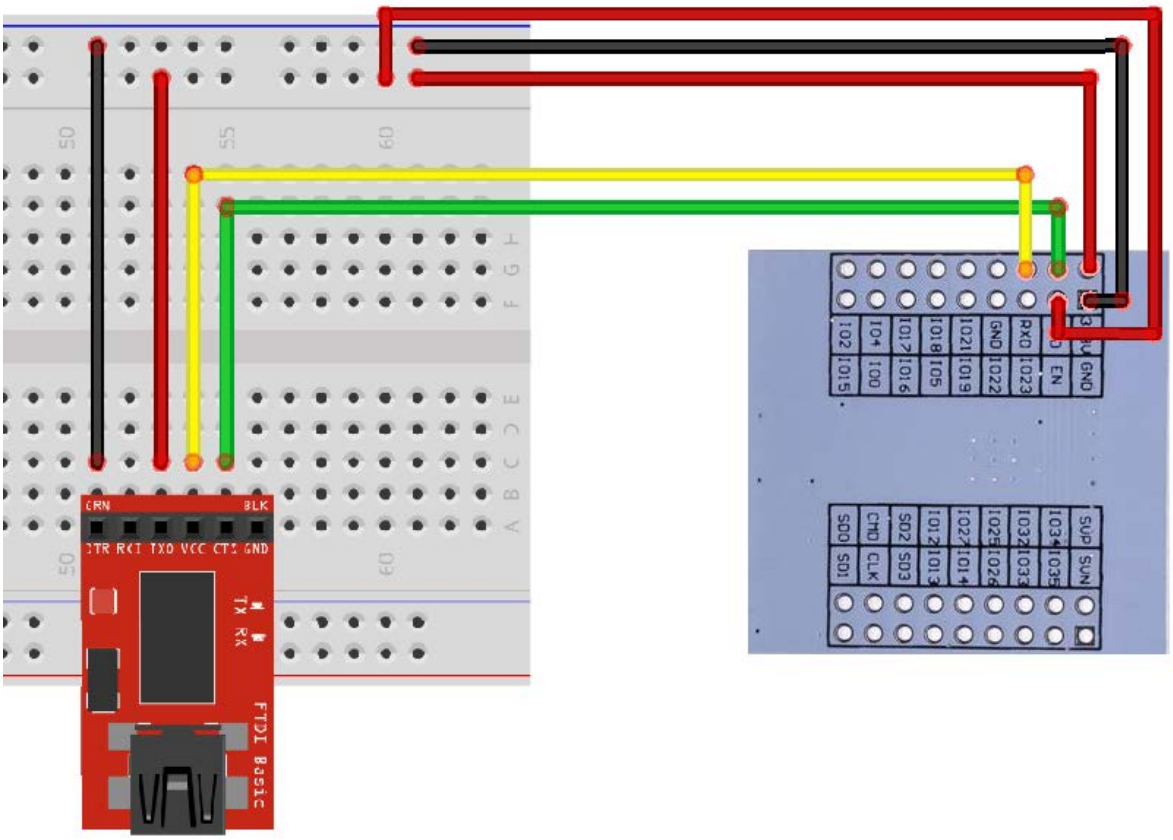

Dies dauert einen kurzen Moment, wenn alles abgeschlossen wurde, schließt das schwarze Fenster von selbst wieder.

Anschließend starten wir die Arduino Software und gehen unter Werkzeuge > Board und suchen uns das ESP32 Dev Module heraus.




Bei Port musst du nur noch den Com-Port deines Serial Adapters eintragen, diesen kannst du beim Gerätemanager auslesen und ggf. auch abändern.

# Verdrahten des Moduls mit dem Serial Adapter:




Die Adapterplatine ist leider für Laborsteckboards nicht geeignet, deswegen müssen wir mit entsprechenden Jumperleitungen eine Verbindung mit dem Serialadapter herstellen.

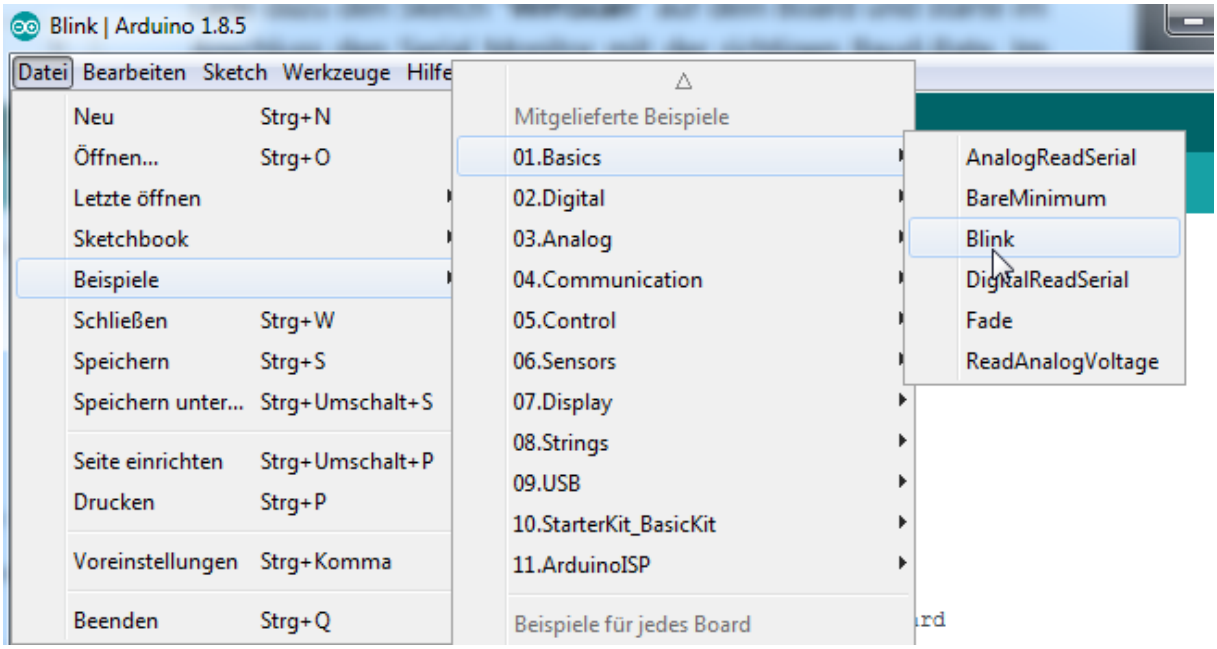
| | |
|---|---|
| VCC mit 3,3V Spannungsversorgung verbinden | rote Leitung |
| GND mit Masse verbinden | schwarze Leitung |
| EN (Chip Enable) mit 3,3V verbinden (High) | rote Leitung |
| TXD auf RX Serial Adapter | gelbe Leitung |
| RXD auf TX Serial Adapter | grüne Leitung |

**Bei der Version 1.1 muss zusätzlich noch ein Jumper bei GPIO0 und GPIO4 gesetzt werden**

## Der Arduino Code:

Nachdem nun die Verdrahtung erledigt wurde, schreiben wir unseren ersten Code. Lassen wir eine LED 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(21, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(21, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(21, 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.

Der ESP32 hat keine eingebaute LED. Möchten wir aber trotzdem eine LED blinken lassen, so müssen wir LED_BUILTIN mit 21 ersetzen. Der IO21 ist ein frei programmierber Pin des ESP32.

Du kannst aber auch eine LED an jeden anderen IO anschließen

(IO15 => `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 158969 Bytes (12%) des Programmspeicherplatzes. Das Maximum sind 1310720 Bytes.
Globale Variablen verwenden 10968 Bytes (3%) des dynamischen Speichers, 283944 Bytes für lokale Variablen verbleiben. Das Maximum sind 294912 Bytes.
```

können wir es auf den ESP32 hochladen. Dazu klicken wir oben auf 

Nach kurzer Zeit erscheint Connecting:

```
esptool.py v2.1
Connecting.....
```

Dann drücken wir beide Taster auf dem Board und lassen den Reset wieder los, aber IO0 bleibt gedrückt, solange bis Hard Reset angezeigt wird, dann wird der IO0 losgelassen(bei Version 1.1 kann hier jetzt der Jumper bei GPIO0 und GPIO4 entfernt werden) und der Reset nur kurz gedrückt. Die LED, welche zwischen dem Pin und Masse angeschlossen wurde (Vorwiderstand für 3,3V Spannung nicht vergessen!) beginnt zum Blinken.

```
esptool.py v2.1
Connecting....
Chip is ESP32D0WDQ6 (revision 1)
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 921600
Changed.
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 8192 bytes to 47...

Writing at 0x0000e000... (100 %)
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.0 seconds (effective 4369.0 kbit/s)...
Hash of data verified.
Compressed 14128 bytes to 9196...

Writing at 0x00001000... (100 %)
Wrote 14128 bytes (9196 compressed) at 0x00001000 in 0.1 seconds (effective 1056.3 kbit/s)...
Hash of data verified.
Compressed 160112 bytes to 81979...

Writing at 0x00010000... (16 %)
Writing at 0x00014000... (33 %)
Writing at 0x00018000... (50 %)
Writing at 0x0001c000... (66 %)
Writing at 0x00020000... (83 %)
Writing at 0x00024000... (100 %)
Wrote 160112 bytes (81979 compressed) at 0x00010000 in 1.5 seconds (effective 855.6 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 122...

Writing at 0x00008000... (100 %)
Wrote 3072 bytes (122 compressed) at 0x00008000 in 0.0 seconds (effective 1638.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting...
```

## Du hast es geschafft, jetzt kannst du deine eigenen Projekte programmieren.

Ab jetzt heißt es eigene Projekte verwirklichen.

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

https://az-delivery.de

Viel Spaß!
Impressum

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