# GRUPPENARBEIT: MIKROCONTROLLER

Onsite Xmas @ CMD+O

**T-Systems on site Services GmbH**

Version: 1.0

**TIME2RACE @ CMD+O**

T · · Systems ·

# IMPRESSUM

Tabelle 1 Impressum


| | |
|---|---|
| **Herausgeber** | T-Systems onsite Services GmbH<br>Fasanenweg 5<br>70771 Leinfelden-Echterdingen |

| Version | Letztes Review | Status |
|---|---|---|
| 1.00 | 20.11.2020 | |

| Kurzbeschreibung |
|---|
| Grundlegende Informationen zur Programmierung von Microcontroller mithilfe der Arduino IDE sowie eine Anleitung zur Integration des ESP32 in die Arduino IDE. |

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# 1 MIKROCONTROLLER

Einfach ausgedrückt handelt es sich bei einem Mikrocontroller um einen kleinen Computer, der kompakt auf einem Chip integriert ist (Ein-Chip-System oder System-on-a-Chip). Der Chip enthält einen Prozessor für die Ausführung von Programmen, verschiedene Arten von Speicher (Arbeits- und Programmspeicher) sowie Schnittstellen für die Kommunikation mit Peripheriegeräten (LEDs, Sensoren, usw.). Mikrocontroller sind in Leistung und Ausstattung auf die jeweilige Anwendung angepasst. Daher haben sie gegenüber „normalen“ Computern Vorteile bei den Kosten und der Leistungsaufnahme. Kleine Mikrocontroller sind in höheren Stückzahlen für wenige Cent verfügbar.

Einsatzbereiche von Mikrocontrollern sind meist in eingebetteten Systemen im Alltag, zum Beispiel in Kaffeemaschinen, Unterhaltungselektronik, Autos und Mobiltelefonen. Dort verrichten sie Steuerungsaufgaben anhand der Messwerte, die sie von ihren angeschlossenen Sensoren erhalten.

Bekannte Hersteller von Mikrocontrollern sind z.B.: Atmel, Intel oder Samsung. Die Programmierung erfolgt meist in den Programmiersprachen C oder Assembler.

# 2 ARDUINO

## 2.1 Arduino Plattform

Arduino ist eine Open-Source Elektronik Plattform, die aus einem Hardware- und Softwareteil besteht. Dadurch soll jedem interessierten die Möglichkeit gegeben werden auf dieser Basis Anpassungen für eigene Projekte zu erstellen und zu verwenden. Ziel ist es, einer breiteren Masse einen einfachen Zugang zu Elektronik zu ermöglichen. Die Webpräsenz ist zu finden unter www.arduino.cc.

## 2.2 Arduino Hardware

Arduino bietet eine breite Palette an vorgefertigten Entwicklerboards an, die direkt für eigene Projekte eingesetzt werden können. Für verschiedene Anwendungsgebiete sind spezielle Boardfamilien verfügbar, zu den bekanntesten zählt der Arduino Uno. Zudem sind für viele Boards die Layouts und Produktionsdateien unter Creative Commons freigegeben. Dadurch sind mittlerweile diverse Arduino kompatiblen Boards von Drittherstellern oder Privatpersonen entstanden.

Die meisten Arduino Boards basieren auf Atmel Mikrocontrollern, die um verschiedene Speicher, Schnittstellen und Funktionen erweitert sind, je nach Modell. Die Controller sind mit einem Bootloader vorprogrammiert, der das Uploaden von eigenem Code über eine USB-Schnittstelle ermöglicht.

Beispielhafter Aufbau eines original Arduino Boards:

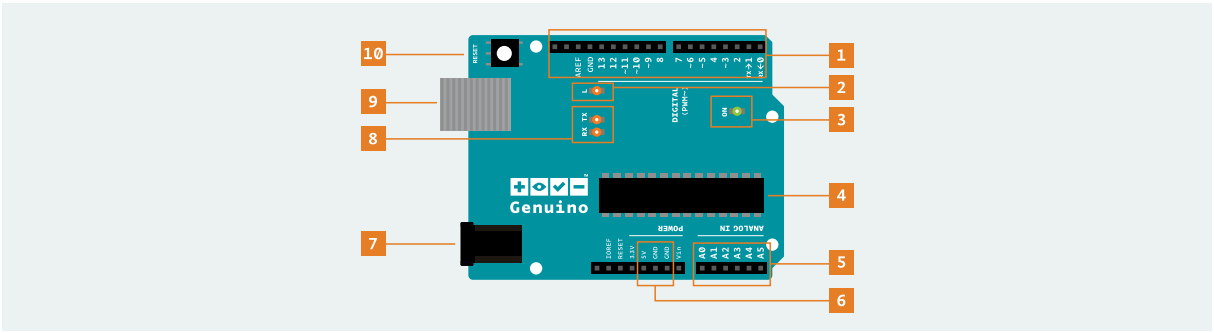


Abbildung 1: Aufbau eines Genuino Boards

1. **Digital Pins:** Können als Ein- und Ausgangspins für Peripherie genutzt werden.
2. **Pin 13 LED:** Einzige OnBoard-LED, kann z.B. für Debugging genutzt werden.
3. **Power LED:** Zeigt an ob der Arduino mit Strom versorgt wird.
4. **ATmega Mikrocontroller:** Das Herz des Boards.
5. **Analog In:** Analoge Eingangspins für Signale von z.B. Sensoren.
6. **GND und 5V Pins:** Anschlusspins, um Peripherie mit Strom über den Arduino zu versorgen.
7. **Stromanschluss:** Kann genutzt werden, um den Arduino mit Strom zu versorgen, wenn er nicht gerade per USB angeschlossen ist.
8. **TX und RX LEDs:** Werden genutzt, um die Kommunikation des Bords mit dem Computer anzuzeigen, z.B. beim Uploaden eines Programms.

9. **USB Port:** Wird genutzt, um den Arduino mit Strom zu versorgen, ein Programm hochzuladen oder über serieller Kommunikation zu debuggen.
10. **Reset Button:** Resettet den ATmega Mikrocontroller.

## 2.3 Arduino IDE

Neben den Hardwareboards bietet Arduino auch eine eigene IDE für die Entwicklung eigener Programme an. Die Arduino IDE wurde mit Java entwickelt und ist kostenlos für Windows, macOS und Linux verfügbar. Die Programmierung selbst erfolgt in einer C bzw. C++-ähnlichen Programmiersprache, wobei technische Details wie Header-Dateien vor den Anwendern weitgehend verborgen werden und umfangreiche Bibliotheken und Beispiele die Programmierung vereinfachen.

Bibliotheken und Boardkonfigurationen - welche zum Entwickeln auf unterschiedlicher Hardware nötig sind - lassen sich einfach über einen eigenen Dienst im Hintergrund herunterladen und aktualisieren, der IDE muss hierzu lediglich eine URL zum Downloadserver zur Verfügung gestellt werden. Viele dieser Bibliotheken werden von den Hardwareherstellern oder der Open Source Community kostenlos angeboten.

## 2.4 Struktur

Der grundlegende Aufbau der Arduino Programmiersprache ist relativ einfach und teilt sich in mindestens zwei Teile auf. Diese zwei benötigten Teile oder Funktionen umschließen Blöcke von Anweisungen.

```
void setup()
{
 setupAnweisungen();
}

void loop()
{
 anweisungen();
}
```

Hierbei ist setup() die Vorbereitung und loop() ist die Ausführung. Beide Funktionen sind notwendig damit das Programm ausgeführt werden kann. Die Setup Funktion sollte der Variablen Definition folgen, die noch davor aufgeführt werden muss. Setup muss als erste Funktion in einem Programm durchlaufen werden. Sie wird nur einmal ausgeführt und dient beispielsweise dem Setzen von PinMode oder der Initiierung der seriellen Kommunikation.

Nach der setup() Funktion folgt die loop() Funktion. Sie beinhaltet Programmcode, der kontinuierlich in einer unendlichen Schleife ausgeführt wird - Eingänge auslesen, Ausgänge triggern, etc. Diese Funktion ist der Kern von allen Arduino Programmen und erledigt die Hauptarbeit.

## 2.5 Serielle Kommunikation

```
Serial.begin(rate);
```

'Serial.begin(rate)' Öffnet den seriellen Port und setzt die Baud Rate (Datenrate) für die serielle Übertragung fest. Die typische Baud Rate mit dem Computer ist 9600 Baud. Andere Geschwindigkeiten werden jedoch auch unterstützt.

```
void setup() {
        Serial.begin(9600);             //öffnet seriellen Port
                                            // setzt die Datenrate auf 9600 bps
}
```

Serial.println(data) Schreibt Daten zum seriellen Port, gefolgt von einem automatischen Zeilenumbruch als Carrier Return und Linefeed. Dieser Befehl hat dieselbe Form wie 'Serial.print()', ist aber einfacher auf dem seriellen Monitor zu lesen.

```
Serial.println(analogValue); // sendet den Wert von 'analogValue'
```

Das folgende einfache Beispiel liest einen Wert vom analogen Pin 0 aus und sendet die Daten an den Computer einmal pro Sekunde.

```
void setup() {
        Serial.begin(9600); // setzt die Datenrate auf 9600 bps
}
void loop() {
        Serial.println(analogRead(0)); // sendet den Analogwert
        delay(1000); // pausiert fuer 1 Sekunde
}
```

.

# 3 ESP32

## 3.1 Chipvorstellung

Der ESP32 Chip ist ein kostengünstiger Mikrocontroller der Firma espressif, der besonders auf geringen Stromverbrauch ausgelegt ist. Sein besonderes Merkmal ist die direkte Integration von WLAN und Bluetooth Low Energy, was ihn im Moment auf dem Markt besonders auszeichnet. Andere Mikrocontroller bieten dies in der Regel nur über Hardwareerweiterungen durch sogenannte „Shields“ an.

Meist wird der ESP32 in verschiedenen Varianten von espressif selbst, aber auch von diversen Drittherstellern auf einem Entwicklungsboard zur Verfügung gestellt. Diese Boards unterscheiden sich untereinander zum Teil stark in der Anzahl und Belegung der Anschlusspins, sowie den technischen Spezifikationen, wie Stromverbrauch und unterstütze Schnittstellen sowie dem Preis.

Die Programmierung des ESP32 kann mit Hilfe verschiedener Plattformen wie Arduino oder MicroPython erfolgen. espressif bietet zudem ein eigenes Framework namens Espressif IoT Development Framework an.

Weiterführende Links:

- https://www.espressif.com/en/products/hardware/esp32/overview
- https://docs.espressif.com/projects/esp-idf/en/latest/

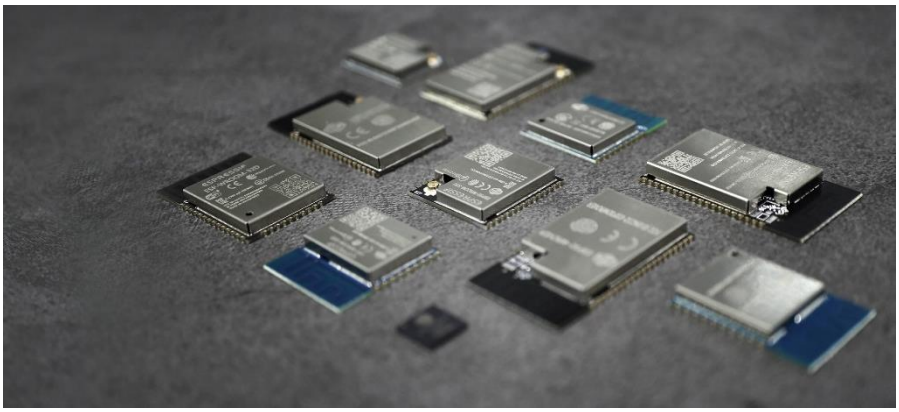

Abbildung 2: Verschiedene ESP32 Chips

## 3.2 ESP32-DevKitC

Für unseren Workshop nutzen wir das gängige Entwicklungsboard „ESP32-DevKitC“ vom Dritthersteller AZ-Delivery, welches ein klein wenig anders aufgebaut ist als das offizielle Modell von espressif:

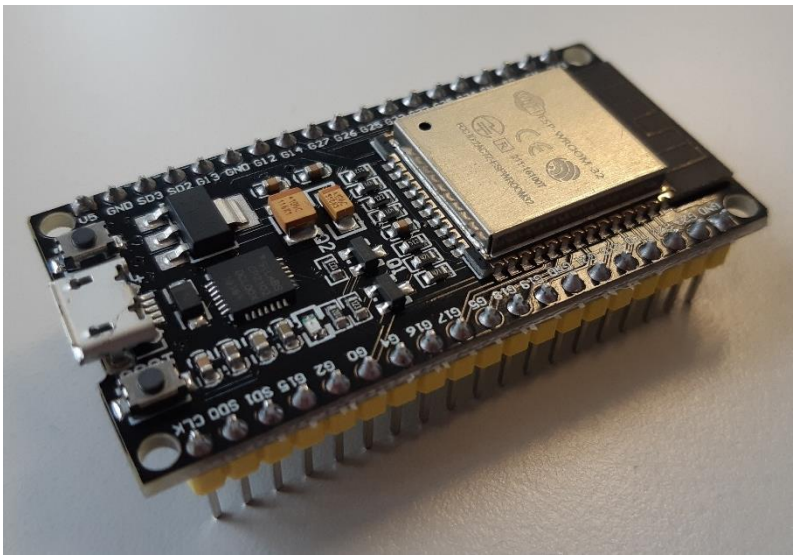

Abbildung 3: ESP32-DevKitC

Das Board verfügt über eine CPU mit 2 Tensilica-LX6-Kernen, getaktet mit bis zu 240 MHz, und 512 Kilobyte SRAM. Dazu integriert er eine Funkeinheit für WLAN (nach 802.11bgn) und Bluetooth (Classic und LE).

Die WLAN-Funktion unterstützt alle gängigen Verschlüsselungsmethoden, wie zum Beispiel WPA2. Außerdem kann er im WLAN auch als Access Point oder Sniffer agieren. Über 32 GPIO-Pins stehen unter anderem UART, I2C, SPI, DAC, ADC (12 Bit) zur Verfügung, alle GPIO-Pins können als Ein- oder Ausgabe benutzt werden.

Weiterführende Links:

- https://www.espressif.com/en/products/hardware/esp32-devkitc/overview
- https://docs.espressif.com/projects/esp-idf/en/latest/hw-reference/get-started-devkitc.html

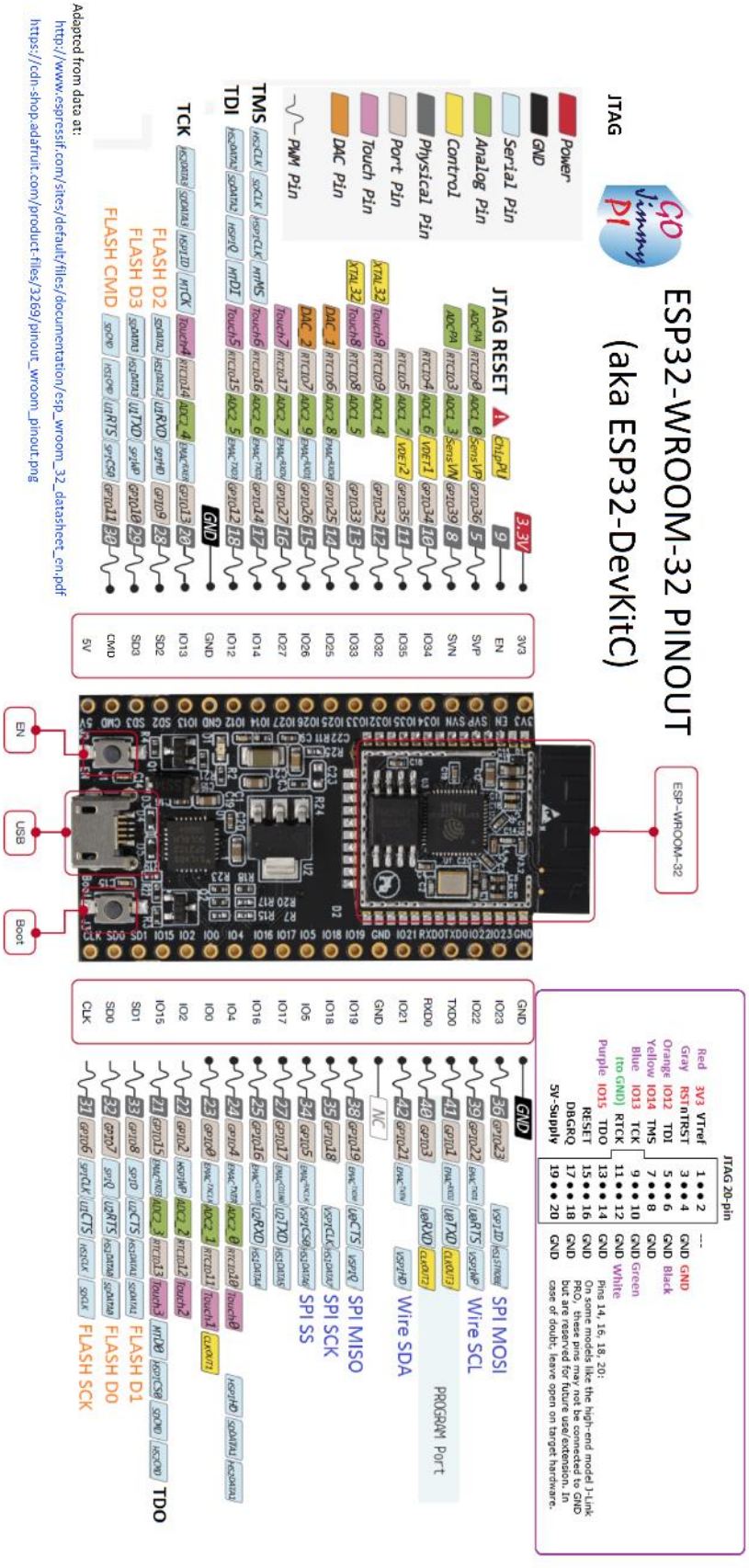


Abbildung 4: Pinout ESP32-DevKitC