ESP32

Bei der Verwendung von fertigem Code aus dem Internet, muss darauf geachtet werden auf welcher ESP API dieser basiert. Mit der Umstellung von v2 auf v3 haben sich einige Befehle verändert. Hier gibt es zwei Möglichkeiten:

1. Die Bibliothek in der IDE anpassen
2. Den Code umschreiben

Weitere Informationen gibt es hier: Migration from 2.x to 3.0

ESP Typen Vergleich

Modell	CPU / Takt	RAM / PSRAM	Flash (üblich)	WLAN	Bluetooth	Zigbee / Thread	USB	GPIOs*	Besondere Funktionen
ESP32	Dual-Core Xtensa LX6 @ 160–240 MHz	~520 KB SRAM, optional PSRAM	4–16 MB	2.4 GHz b/g/n	BT 4.2 BR/EDR + BLE	nein	nein	~34	Sehr viele Peripherals, RMT, Hall-Sensor
ESP32-S2	Single-Core Xtensa LX7 @ 160–240 MHz	320 KB SRAM, optional PSRAM	4–16 MB	2.4 GHz b/g/n	kein BT	nein	USB-OTG	~43	Touch-Sensoren, USB-Device/Host, verbesserte Sicherheit
ESP32-S3	Dual-Core Xtensa LX7 @ 160–240 MHz	512 KB SRAM + optional PSRAM	4–16 MB	2.4 GHz b/g/n	BLE 5.0 (LE)	nein	USB-OTG	~45	AI-Beschleuniger (Vektor-Instr.), Kamera-Interface
ESP32-C3	Single-Core RISC-V @ 160 MHz	400 KB SRAM	4 MB	2.4 GHz b/g/n	BLE 5.0 (LE)	nein	USB-Seriell (teilweise)	~22	Sehr stromsparend, sichere Boot-Kette
ESP32-C6	Single-Core RISC-V @ 160 MHz	400 KB SRAM	4 MB	Wi-Fi 6 (2.4 GHz)	BLE 5.0 (LE)	Zigbee + Thread (802.15.4)	USB (variiert)	~28	Matter-fähig, moderne Funkplattform
ESP32-H2	Single-Core RISC-V @ 96 MHz	256 KB SRAM	extern	kein WLAN	BLE 5.2 (LE)	Zigbee + Thread	kein USB	~24	Extrem low-power, Matter-IoT-Knoten
ESP32-P4	Dual-Core RISC-V @ bis 400 MHz	bis 768 KB SRAM	extern	kein WLAN	kein BT	kein Zigbee/Thread	USB-OTG	~50+	Hochleistungs-MCU ohne Funk, GPU-ähnliche Beschleuniger

Kurzbeschreibung aller ESP32-Serien

ESP32 (Classic Series)

Die ursprüngliche ESP32-Familie mit Dual-Core Xtensa-CPU, WLAN 2.4 GHz und Bluetooth Classic/BLE. Sehr leistungsfähig, viele GPIOs, große Modulauswahl (WROOM/WROVER). Ideal für allgemeine IoT-,

Sensor-, Display- und Steuerungsprojekte.

ESP32-S2 Serie

Single-Core Variante mit Fokus auf Sicherheit (integrierter Hardware-Schutz) und USB-OTG. Hat KEIN Bluetooth. Enthält Touch-Sensoren und verbesserte Peripherie. Gut geeignet für USB-Geräte, HID, Webserver, Tastaturen, Sicherheitssysteme.

ESP32-S3 Serie

Dual-Core Xtensa, WLAN + BLE5.0, USB-OTG und ein AI-Vektor-Beschleuniger für Sprach-/Bildverarbeitung. Häufig in neuen Dev-Boards. Unterstützt Kamera-Interfaces und schnelle Peripherie. Moderner Nachfolger des ESP32 Classic mit mehr Möglichkeiten.

ESP32-C2 Serie

Sehr günstige Low-End-Serie auf RISC-V Basis. WLAN + BLE5.0, aber wenig RAM und wenig GPIOs. Für Massenprodukte und einfache IoT-Sensoren optimiert.

ESP32-C3 Serie

RISC-V Single-Core, extrem stromsparend, WLAN + BLE5.0, hohe Sicherheit. Gilt als „ESP8266-Nachfolger“. Ideal für kleine IoT-Geräte, Smart-Home-Module, Sensoren und Akkuprojekte.

ESP32-C5 Serie

Wi-Fi 6 Unterstützung, ohne Bluetooth. Selten und kaum verwendet, da der C6 eine vollständige Ablösung darstellt.

ESP32-C6 Serie

RISC-V, WLAN 6 + BLE5.0 + Zigbee + Thread 802.15.4. Perfekt für Matter-fähige Smart-Home-Geräte. Eine der modernsten und flexibelsten Funkplattformen von Espressif.

ESP32-H2 Serie

Kein WLAN! Stattdessen Zigbee + Thread + BLE5.2. Gemacht für extrem stromsparende, batteriebetriebene Smart-Home-Knoten (Sensoren, Schalter). Ideal für zukünftige Matter/Thread Installationen.

ESP32-P4 Serie

High-Performance MCU ohne Funk. Dual-Core RISC-V bis 400 MHz, große Menge an Peripherie, USB-OTG, viele GPIOs. Einsatz als Displaycontroller, Rechen- oder Grafikprozessor, HMI-Panel. Kann mit externen Funksystemen kombiniert werden.

ESPHome

ESPHome Themen werden hier behandelt: ESPHome

Cheap Yellow Display (CYD)

Empfohlenes GitHub Repository: ESP32-Cheap-Yellow-Display

Empfohlene Libraries:

https://github.com/Bodmer/TFT_eSPI

Linksammlung bei SPI Problemen (Touch, SPI und LCD brauchen jeweils einen SPI Bus -> So nicht möglich beim ESP32):

https://github.com/Bodmer/TFT_eSPI/pull/3186

https://forum.arduino.cc/t/question-about-spi-bus-assignment-for-tft-espi-touch-screen-and-sd-card-for-cyd/1352613