Pclika

Entwicklerhandbuch

Von der Platine zur KI — vollstandige Einrichtung und Workflow

DOCUMENT · Pclika ESP32-S3 + pclika-bridge + MCP

VERSION · 0.1.0 · Juni 2026

CLIENTS · Claude Code · Codex · Cursor · VS Code · OpenCode

STACK · Python 3.10+ · ESP-IDF v5.x

CONTACT · pclika.com · starinvc@gmail.com

SEAL · PCK-MMXXVI-C4A32096

Diese Anleitung fuhrt Sie Schritt fur Schritt vom Kauf der Hardware bis zum ersten KI-gesteuerten Hardwarebefehl. Keine Hardwarerfahrung erforderlich.

00 ÜBERBLICK

Pclika verwandelt ein ESP32-S3-Board in eine KI-gesteuerte Hardwareplattform. Der KI-Assistent ruft Hardwarefunktionen uber Standard-MCP-Werkzeuge (JSON-RPC 2.0) per USB in Echtzeit auf.

SCHRITT	TITEL	BESCHREIBUNG
01	Kit kaufen	Im offiziellen Shop oder uber MCP bestellen
02	Hardware anschliessen	USB-C an PC, I2C-Sensoren an JST SH
03	Firmware flashen	Ein Befehl: idf.py flash — ~90 Sekunden
04	Bridge installieren	pip install pclika-bridge — fertig
05	Verbindung pruefen	pclika-bridge --list-ports dann device_info
06	KI-Client konfigurieren	Claude Code / Codex / Cursor / VS Code / OpenCode
07	Anweisungen laden	System-Prompt einmalig einfugen
08	Projekt beschreiben	Anforderung in normaler Sprache schreiben
09	Iterieren und deployen	KI schreibt Firmware, Sie pruefen und flashen
10	Skalieren	Module hinzufugen, SSE-Mehrclient-Modus nutzen

01 KIT KAUFEN

Der einfachste Einstieg ist das Pclika Starter-Kit — Platine und Sensorpaket, getestet mit allen Beispielen.

OFFIZIELLE KAUFKANÄLE

Benutzershop (offiziell)	**https://mall.pclika.com/**
MCP Self-Service-Kauf	**https://hc.pclika.com/**

Starter-Kit ¥299 (Empfohlen)

ARTIKEL	MENGE	HINWEISE
Pclika ESP32-S3 Baseboard v0.1	1 Stk.	Haupt-MCU · USB-C · Wi-Fi · BLE
Sensor DHT22 Temp./Feuchte	1 Stk.	I2C · Beispiel env-monitor
Lichtsensor BH1750	1 Stk.	I2C · Lux-Messung
USB-C-Kabel (1 m)	1 Stk.	Datenkabel — kein reines Ladekabel
JST SH 4P Kabel × 3	3 Stk.	I2C-Sensorverbindungen

DIY / EIGENE PLATINE

KiCad-Designdateien und JLCPCB-Bestellanleitung befinden sich im GitHub-Repository.

OPTIONALE MODULE

MODUL	ANWENDUNG
Servo SG90 × 4	PWM-Ausgange — Beispiel servo-control
OLED SSD1306 128x64	I2C-Display — Werkzeug display_text
Kamera OV2640	Kamera — Beispiel vision-snapshot
RS-485-Adapter	Modbus RTU — Beispiel industrial-gateway
FPGA iCE40UP5K M.2	FPGA-Erweiterung — HDL Bridge

02 HARDWARE ANSCHLIESSEN

PHYSISCHE VERBINDUNGEN

GERÄT	VERBINDUNG	HINWEIS
USB-C-Kabel	Platine J1 → USB PC	Stromversorgung + serielle Verbindung
DHT22/BME280	SDA=GPIO21 SCL=GPIO22 JST SH 4P	Beispiel env-monitor
OLED SSD1306	Gleicher I2C-Bus, Adresse 0x3C	Werkzeug display_text
Servo	PWM0–PWM3-Anschlusse	Werkzeug servo_move

LED-STATUS DER PLATINE

LED-ZUSTAND	BEDEUTUNG
Blau dauerhaft	Eingeschaltet — Firmware lauft
Langsames Blinken	Bridge wartet auf MCP-Befehle
Schnelles Blinken	Aktive MCP-Sitzung — Aufrufe werden empfangen
Rote LED an	Fehler — serial_log_read prufen

>> Verwenden Sie ein USB-Datenkabel — reine Ladekabel haben keine Datenleitungen und erscheinen nicht in der Portliste.

03 FIRMWARE FLASHEN

VORAUSSETZUNGEN

- Python 3.10+ (python3 --version)

- ESP-IDF v5.x (docs.espressif.com/projects/esp-idf)

- idf.py im PATH (idf.py --version)

- Pclika-Platine uber USB-C-Datenkabel verbunden

KLONEN UND FLASHEN

```
git clone https://github.com/Pclika/mcp-platform
cd mcp-platform/firmware
idf.py set-target esp32s3
idf.py flash monitor
```

VORKOMPILIERTE FIRMWARE (ohne ESP-IDF)

Laden Sie die .bin-Datei von GitHub Releases herunter und flashen Sie mit esptool:

```
pip install esptool
esptool.py --port /dev/ttyUSB0 --baud 460800 \
  write_flash -z 0x0 pclika-firmware-v0.1.bin
```

>> Windows: Falls idf.py flash den Port als belegt meldet, schliessen Sie Arduino IDE, PuTTY usw.

04 BRIDGE INSTALLIEREN

```
pip install pclika-bridge

# verify
pclika-bridge --version
pclika-setup  --list
```

WAS INSTALLIERT WIRD

BEFEHL / PAKET	ZWECK
pclika-bridge	CLI-Befehl — MCP-Server starten (STDIO oder SSE)
pclika-setup	CLI-Befehl — Auto-Konfiguration aller KI-Clients
pclika_bridge	Python-Bibliothek — in eigenen Skripten importierbar

VERBINDUNGSMODI

MODUS	BESCHREIBUNG	OPTIMAL FÜR
STDIO (Standard)	Jeder KI-Client startet seinen eigenen Bridge-Prozess	Einfach, sofort einsatzbereit
SSE / Adaptiv	Ein Bridge-Prozess fur alle Clients	Ideal fur Mehrclient-Betrieb

05 VERBINDUNG PRÜFEN

VERFÜGBARE PORTS AUFLISTEN

```
pclika-bridge --list-ports
# /dev/ttyUSB0   Pclika ESP32-S3  (USB VID:PID=303A:1001)
```

SCHNELLER FUNKTIONSTEST

Bridge im Debug-Modus starten und device_info manuell aufrufen:

```
pclika-bridge --debug
> {"jsonrpc":"2.0","id":1,"method":"tools/call",
   "params":{"name":"device_info","arguments":{}}}
< {"mcu":"ESP32-S3","firmware":"0.1.0","ip":"192.168.1.x","mac":"AA:BB:CC:..."}
```

>> Bei DeviceNotConnected: Kabel prufen, blaue LED prufen, Port mit --list-ports abgleichen.

06 KI-CLIENT KONFIGURIEREN

pclika-setup ausfuhren zur automatischen Konfiguration aller erkannten Clients:

```
pclika-setup
# Detected: Claude Code, Cursor, VS Code
# Writing ~/.cursor/mcp.json         ... OK
# Writing .vscode/mcp.json           ... OK
# Running: claude mcp add pclika...  ... OK

# SSE mode (multi-client):
pclika-bridge --serve &
pclika-setup --sse
```

CLIENT	KONFIGDATEI / BEFEHL	PFAD
Claude Code	claude mcp add pclikaPlatform pclika-bridge	run once in terminal
Cursor	cursor.mcp.json	~/.cursor/mcp.json
VS Code	vscode.mcp.json	.vscode/mcp.json
Codex	codex.config.toml	~/.config/codex/config.toml
OpenCode	opencode.jsonc	~/.config/opencode/config.json

07 ANWEISUNGEN LADEN

Fugen Sie den folgenden System-Prompt zu Beginn einer Sitzung in Ihren KI-Client ein (oder als dauerhaften System-Prompt). Dies teilt der KI mit, welche Werkzeuge verfugbar sind.

SYSTEM-PROMPT — kopieren und einfügen

```
You are an AI assistant with direct hardware control via MCP tools.
Connected: Pclika ESP32-S3 board on local USB serial port.

ALWAYS call device_info() first to confirm connection.

Available MCP tools (server: pclikaPlatform):
  sensor_read     - read I2C/UART sensor (temp, humidity, light, distance, IMU)
  display_text    - write text to SSD1306 OLED or ST7789 TFT
  servo_move      - move servo channel (0-3) to angle (0-180 degrees)
  gpio_write      - set GPIO pin HIGH (1) or LOW (0)
  gpio_read       - read GPIO pin state
  led_control     - set R/G/B (0-255) on 4x WS2812B LEDs
  wifi_scan       - list nearby Wi-Fi networks (SSID, dBm, channel)
  button_read     - read BOOT/RESET/USER button state
  device_info     - MCU model, firmware version, IP, MAC
  serial_log_read - UART diagnostic log (default 20 lines, max 200)

Safety: GPIO is 3.3V. Servos need 5V via PWM header.
On DeviceNotConnected error -- stop and report, do not retry in a loop.
```

>> Claude Code: in CLAUDE.md oder --system-prompt. Cursor/VS Code: Einstellungen → KI → System-Prompt. Codex/OpenCode: --system-Flag oder system_prompt-Feld in der Konfiguration.

08 PROJEKT BESCHREIBEN

Nachdem der Prompt geladen ist, beschreiben Sie einfach, was Sie bauen mochten. Die KI plant die Implementierung, ruft Hardwarewerkzeuge auf und generiert Firmware-Code.

BEISPIEL-ANFORDERUNGEN

TITEL	ANFORDERUNG	FÄHIGKEITEN
Hello World	Eingebaute LED dreimal grun blinken lassen, dann blau, dann ausschalten.	Test des gesamten Stacks in 10 Sekunden.
Env-Monitor	BME280 alle 5 s lesen, Temp. und Feuchte auf OLED anzeigen. LED rot bei >28 Grad C.	Sensor + Display + Logik.
Servo-Sweep	Servo 0 von 0 bis 180 Grad in 10-Grad-Schritten bewegen, 500ms Pause, Winkel protokollieren.	PWM + Feedback.
Wi-Fi-Scan	WLAN-Netze scannen, sortierte Tabelle SSID/dBm/Kanal ausgeben, offene Netzwerke markieren.	HF-Daten + KI-Analyse.
Eigenes Projekt	Pflanzenbewasserungs-Monitor: ADC0 jede Minute lesen, bei <30% rote LED 5x blinken lassen + Zeitstempel protokollieren.	ADC + LED + Log.

09 DER VOLLSTÄNDIGE ENTWICKLUNGSZYKLUS

Nach der Einreichung der Anforderung fuhrt die KI diesen Workflow automatisch aus:

#	PHASE	WAS PASSIERT
1	Verbindungsprufung	KI ruft device_info() auf, bevor sie irgendetwas tut.
2	Sensorprufung	Fur Sensor-Projekte liest die KI einmal zur Bestatigung.
3	Planung	KI erklart den Implementierungsplan und bittet um Bestatigung.
4	Firmware-Code	KI schreibt kommentierten ESP-IDF-C-Code mit exakten GPIO-Pins.
5	Flash-Anweisung	KI gibt den idf.py-Befehl an und was im Monitor zu erwarten ist.
6	Live-Test	KI ruft MCP-Werkzeuge auf, um laufende Firmware zu prufen.
7	Iteration	Bei Fehlern liest KI serial_log_read() und schlagt Korrekturen vor.

VOLLSTÄNDIGES SITZUNGSPROTOKOLL (hello-mcp)

```
You: Make the LED pulse cyan 5 times then show 'HELLO' on the OLED.

[AI] device_info()              => ESP32-S3 online, firmware 0.1.0
[AI] led_control(r=0,g=200,b=160) x5  => cyan pulse
[AI] led_control(r=0,g=0,b=0)    => off
[AI] display_text('HELLO', line=0)     => OLED updated
AI: Done. LED pulsed cyan 5x. 'HELLO' is shown on OLED line 0.
```

10 SKALIEREN

MULTI-CLIENT SSE-MODUS

Wenn Cursor und VS Code gleichzeitig offen sind, starten Sie eine geteilte Bridge:

```
# Terminal 1 — keep running
pclika-bridge --serve
# => Listening on http://localhost:3141/sse

# Terminal 2 — write SSE config to all clients
pclika-setup --sse

curl http://localhost:3141/health
# {"status":"ok","port":"/dev/ttyUSB0","connected_clients":2}
```

WEITERE MODULE HINZUFÜGEN

MODUL	FUNKTION	MCP-AUFRUF
BME280	Temp. · Feuchte · Druck	`sensor_read("bme280")`
MPU6050	Beschleunigung · Gyroskop	`sensor_read("imu")`
VL53L0X	Distanzsensor (ToF)	`sensor_read("distance")`
SHT31	Hochprazise Temp./Feuchte	`sensor_read("sht31")`
PCA9685	16-Kanal-PWM	`servo_move(ch, angle)`
OV2640	Kamera — JPEG-Aufnahme	`camera_capture()`

FPGA-ERWEITERUNG (fortgeschritten)

iCE40UP5K-FPGA-Modul in den M.2-Slot stecken. HDL Bridge fur 6 zusatzliche MCP-Werkzeuge:

```
pip install pclika-hdl-bridge
pclika-setup   # adds pclikaHDL server automatically
```

ANHANG A — KURZREFERENZ DER MCP-WERKZEUGE

WERKZEUG	PARAMETER	GIBT ZURÜCK / AKTION
`sensor_read`	sensor_id: "temp_humidity"|"bme280"|"light"|"distance"|"imu"|"sht31"	Gibt Sensordaten als JSON zuruck
`display_text`	text: str · line: int (0–7 OLED-Zeilen)	Auf SSD1306/ST7789 schreiben
`servo_move`	channel: int (0–3) · angle: float (0–180)	Servo auf Winkel drehen
`gpio_write`	pin: int · value: int (0 oder 1)	GPIO hoch oder niedrig setzen
`gpio_read`	pin: int	GPIO-Zustand lesen (0 oder 1)
`led_control`	r/g/b: int (0–255)	4x WS2812B LED-Farbe setzen
`wifi_scan`	(keine Parameter)	Liste der WLAN-Netze in der Nahe
`button_read`	"BOOT"|"RESET"|"USER"	Tastenzustand (0=los, 1=gedruck)
`device_info`	(keine Parameter)	MCU, Firmware, IP, MAC
`serial_log_read`	lines: int (Standard 20, max 200)	UART-Log fur Diagnose lesen

ANHANG B — FEHLERBEHEBUNG

DeviceNotConnected	Hinweis: 90% der Falle: reines Ladekabel

1. Leuchtet die blaue LED?
2. Datenkabel verwenden (kein reines Ladekabel).
3. --list-ports ausfuhren.
4. Linux: sudo usermod -aG dialout $USER

Port sichtbar, Bridge hangt	Hinweis: idf.py monitor fur Boot-Meldungen

1. Falsche Firmware — neu flashen.
2. Falsche Baudrate — --baud 115200 verwenden.
3. Port belegt — Arduino IDE schliessen.

KI-Client: Server nicht gefunden	Hinweis: Claude Code: /mcp fur Serverstatus

1. pclika-setup erneut ausfuhren.
2. KI-Client nach Konfigurationsanderung neu starten.
3. PATH prufen: which pclika-bridge

sensor_read gibt null zuruck	Hinweis: I2C-Scanner zum Finden der Adresse

1. Verdrahtung prufen: SDA=GPIO21, SCL=GPIO22.
2. serial_log_read(50) ausfuhren.
3. Sensoradresse prufen (BME280=0x76/0x77).

servo_move: keine Bewegung	Hinweis: SG90 direkt mit USB 5V versorgen

1. Servo benotigt 5V, nicht 3.3V.
2. Kanale: 0=GPIO38, 1=39, 2=40, 3=41.
3. Winkel muss 0.0–180.0 sein.

SSE: Clients verbinden nicht	Hinweis: pclika-setup --sse uberschreibt alle Konfigs

1. pclika-bridge --serve lauft es?
2. http://localhost:3141/health prufen.
3. Firewall — Ausnahme fur Port 3141.

SUPPORT UND FEEDBACK

Ein Problem? Eine Funktionsanfrage? Unser Support-Portal ist immer geöffnet.

https://pclika.com/Support

KANAL	KONTAKT
Web-Portal	https://pclika.com/Support
E-Mail	starinvc@gmail.com
GitHub-Issues	github.com/Pclika/mcp-platform/issues
Benutzershop	https://mall.pclika.com/
MCP-Self-Service	https://hc.pclika.com/

Ihr Feedback gestaltet direkt die nächste Version. Jeder Fehlerbericht und jede Anfrage zählt.

github.com/Pclika/mcp-platform

pypi.org/project/pclika-bridge

Shop: https://mall.pclika.com/ · MCP Self-Service: https://hc.pclika.com/

Pclika