ChatGPT

ESP32 Microcontroller + MicroPython/Arduino Cheatsheet

This guide covers essential commands, setup steps, and quirks for working with the **ESP32** microcontroller using **MicroPython** or **Arduino IDE**. Ideal for robotics, embedded systems, and IoT projects.

For full official documentation from Espressif Systems, visit:\ https://docs.espressif.com/projects/esp-idf/en/latest/esp32/

Getting Started: Out-of-the-Box Setup

1. **Install USB Drivers** (if needed):
2. macOS usually has drivers built-in.
3. Windows users: CP210x USB to UART driver
4. **Connect via USB** and check port:

```
ls /dev/tty.*  # macOS/Linux
```

1. **Install MicroPython or Arduino IDE**:
2. See instructions in Flashing & Setup section below.
3. **Install a tool to communicate with the board**:
4. `screen`, `minicom`, `PuTTY`, or `Thonny` (GUI-based editor for MicroPython).

Flashing & Setup

Flash MicroPython Firmware

```
# Install esptool
pip install esptool

# Erase flash
```

```
esptool.py --port /dev/ttyUSB0 erase_flash

# Flash MicroPython (adjust .bin path)
esptool.py --chip esp32 --port /dev/ttyUSB0 write_flash -z 0x1000 esp32-
xxxxxx.bin
```

Connect to REPL (Terminal Interface)

```
# Connect using screen (Linux/macOS)
screen /dev/ttyUSB0 115200

# Or use minicom or PuTTY (Windows)
```

Uploading Files (MicroPython)

```
# Install ampy
pip install adafruit-ampy

# Upload script
ampy --port /dev/ttyUSB0 put main.py

# Run script live
ampy --port /dev/ttyUSB0 run main.py
```

⚙ GPIO (MicroPython)

```python
from machine import Pin
led = Pin(2, Pin.OUT)
led.value(1)  # Turn on
```

Wi-Fi Connection (MicroPython)

```python
import network
sta = network.WLAN(network.STA_IF)
sta.active(True)
sta.connect('SSID', 'password')
while not sta.isconnected():
    pass
print('Connected, IP:', sta.ifconfig())
```

Serial & UART (MicroPython)

```
from machine import UART
uart = UART(1, baudrate=9600, tx=17, rx=16)
uart.write('Hello')
```

I2C Sensors (MicroPython)

What is I2C?

> I2C (Inter-Integrated Circuit) is a two-wire communication protocol used to connect digital sensors and peripherals to the ESP32. It uses SDA (data) and SCL (clock) lines and allows multiple devices to share the same two wires.

For an in-depth reference, visit the I2C section of the ESP-IDF docs

Typical Devices

- Temperature sensors (e.g. BME280, TMP102)
- OLED displays
- Accelerometers and gyroscopes (e.g. MPU6050)

Wiring Example

- SDA → GPIO21 (default)
- SCL → GPIO22 (default)

Setup and Scan

```
from machine import I2C, Pin

i2c = I2C(0, scl=Pin(22), sda=Pin(21))
print(i2c.scan())  # List device addresses found on the bus
```

Reading a BME280 Sensor (with `bme280.py` driver file)

```
import bme280
bme = bme280.BME280(i2c=i2c)
temp, pressure, humidity = bme.read_compensated_data()
print(temp, pressure, humidity)
```

Quirks:

- Always scan first to confirm your device is wired properly.
- Use pull-up resistors (\~4.7kΩ) on SDA and SCL lines if signals are unstable.
- Not all sensors work at 3.3V—double-check datasheets.

⚙ Motor Control (MicroPython)

Common Motor Types:

- **DC Motors** (via motor drivers like L298N, DRV8833)
- **Servo Motors** (SG90, MG996R, etc.)
- **Stepper Motors** (28BYJ-48, NEMA 17)

Example: Controlling a Servo Motor

```
from machine import Pin, PWM
from time import sleep

servo = PWM(Pin(14), freq=50)

# Function to set angle (0 to 180 degrees)
def set_angle(angle):
    duty = int(40 + (angle / 180) * 115)
    servo.duty(duty)

set_angle(90)  # Move to center
sleep(1)
set_angle(0)   # Move to 0 degrees
sleep(1)
set_angle(180) # Move to 180 degrees
```

Example: Control DC Motor with L298N

```
from machine import Pin
from time import sleep
```

```
in1 = Pin(5, Pin.OUT)
in2 = Pin(18, Pin.OUT)

# Forward
in1.value(1)
in2.value(0)
sleep(2)

# Backward
in1.value(0)
in2.value(1)
sleep(2)

# Stop
in1.value(0)
in2.value(0)
```

Arduino IDE Setup (Alternative to MicroPython)

1. Open Arduino IDE
2. Go to **Preferences** > add ESP32 URL:

   ```
   https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/
   package_esp32_index.json
   ```

3. Board Manager > install **esp32 by Espressif Systems**
4. Select board: `ESP32 Dev Module`

Digital Output (Arduino)

```
void setup() {
  pinMode(2, OUTPUT);
}

void loop() {
  digitalWrite(2, HIGH);
  delay(1000);
  digitalWrite(2, LOW);
  delay(1000);
}
```

🛠 Tips & Quirks

- ESP32 can run MicroPython or Arduino — choose based on project needs.
- GPIO2 is usually connected to onboard LED.
- Some USB cables are **power only** — no data. Always verify.
- Watch for **boot mode issues** if GPIO0 is grounded during reset.
- Use `Ctrl+A` then `K` to exit `screen` (REPL).
- MicroPython REPL can be buggy on Windows—try PuTTY with correct baudrate.

Power & Sleep

```python
import machine
machine.deepsleep(10000)  # sleep for 10 seconds
```

Let this evolve as you explore the ESP32! Log your tricks, favorite libraries, and board-specific behaviors.