ESP8266 D1 Board

User Manual

Open-Source Development Board · Arduino Compatible · Ready for Secondary Development

Document Version: **V1.0**

Release Date: **July 2026**

Contents

1 Overview

1.1 Product Overview

The ESP8266 D1 Board is an open-source development board based on the ESP-8266 Wi-Fi module. It follows the Arduino UNO board layout and pin-header arrangement, integrates a Micro-USB serial chip and a 5V/1A switching power supply, and lets you write and upload programs directly with the Arduino IDE — a cost-effective choice for getting started with IoT development.

The board ships with its hardware and software resources fully open, allowing secondary development. You can use it directly as a Wi-Fi-enabled Arduino board, or build on it by trimming the low-level firmware and expanding peripheral interfaces. It suits IoT education, smart-home prototyping, and wireless data acquisition scenarios.

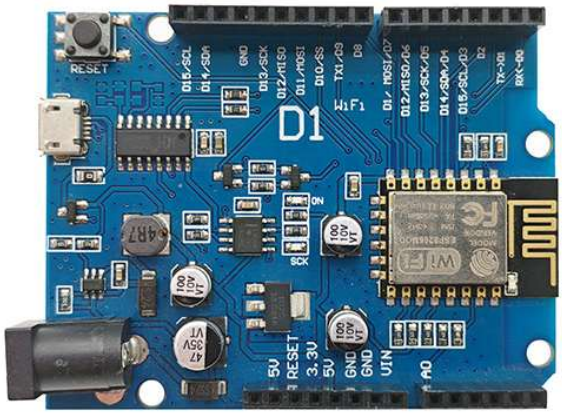

Figure 1-1 ESP8266 D1 UNO development board (top view)

1.2 Key Advantages

- **Open source:** All hardware and software resources are open; users may freely modify and extend the board.
- **Mature ecosystem:** Arduino-compatible — program it directly with the Arduino IDE, with a gentle learning curve and abundant example code.

- **Built-in Wi-Fi:** Based on the ESP-8266 Wi-Fi module, the board connects to networks wirelessly out of the box — no extra communication module needed.
- **Wide power input:** On-board 5V/1A switching power supply accepts a 5–12 V input, giving flexible powering options.
- **Easy upgrades:** OTA over-the-air upload is supported, so firmware updates no longer require repeated cable plugging.

1.3 Intended Users

- Beginners in IoT and embedded development;
- Product prototype developers who need to validate Wi-Fi features quickly;
- Teachers and students running Arduino or ESP8266 courses and experiments.

2 Product Specifications

The main technical specifications of the ESP8266 D1 are listed in the table below.

Item	Specification
Core module	ESP-8266 Wi-Fi module
Programming	Arduino IDE (Arduino-compatible)
Digital I/O	11 I/O pins
Analog input	1 ADC channel, 0–3.3 V input range
Wireless upgrade	OTA over-the-air upload supported
On-board power	5V/1A switching power supply (input 5–12 V)
Flash	4 MB
SRAM	32 KB
DRAM	80 KB
Clock frequency	80 MHz / 160 MHz

Table 2-1 Technical specifications

- **Logic level note:** On-board I/O operates at 3.3 V; use proper level shifting when connecting 5 V devices.

3 Interfaces and Hardware

3.1 On-Board Resources

The figure below shows the layout of the board's functional blocks. The Arduino UNO-compatible I/O expansion headers run around the edges, while the power, USB-serial, and core module areas sit in the middle.

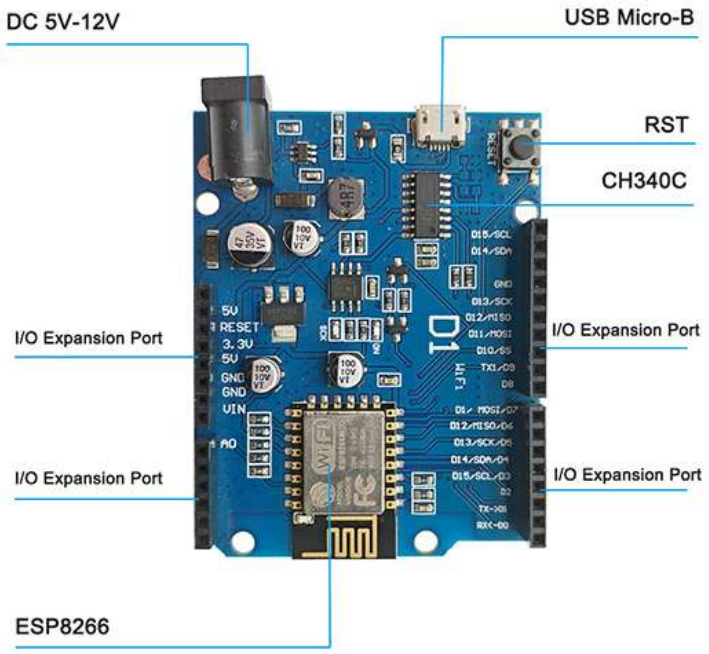

Figure 3-1 On-board resource layout

Label	Name	Description
DC 5V-12V	DC power input	Input of the on-board 5V/1A switching power supply; accepts a 5–12 V wide-range input

Label	Name	Description
USB Micro-B	USB port	Used for power supply, program download, and serial debugging
RST	Reset button	Press to restart the module
CH340C	USB-to-serial chip	Handles serial communication between USB and the ESP8266, and program download
I/O Expansion Port	I/O expansion headers	Expose all I/O pins; Arduino UNO-compatible header layout (4 groups)
ESP8266	Core module	Core Wi-Fi module; runs user programs

Table 3-1 On-board resources

3.2 I/O Pin Description

The board exposes 11 digital I/O pins and 1 ADC channel. The table below lists each pin's silkscreen label, function, and the corresponding chip-internal GPIO.

Pin	Description	Internal IC pin
D0 (RX)	Serial receive	GPIO3
D1 (TX)	Serial transmit	GPIO1
D2	I/O; no interrupt, PWM, I2C, or 1-wire support	GPIO16
D3 / SCL / D15	I/O; I2C SCL in the default configuration	GPIO5
D4 / SDA / D14	I/O; I2C SDA in the default configuration	GPIO4
D5 / SCK / D13	I/O; SPI clock (SCK)	GPIO14

Pin	Description	Internal IC pin
D6 / MISO / D12	I/O; SPI MISO	GPIO12
D7 / MOSI / D11	I/O; SPI MOSI	GPIO13
D8	I/O; pulled up; enters FLASH mode when low	GPIO0
D9 / TX1	I/O; pulled up	GPIO2
D10 / SS	I/O; pulled down; default SPI chip select (SS)	GPIO15
A0	Analog input, 0–3.3 V	ADC

Table 3-2 I/O pin description

- **Caution:** All I/O pins operate at 3.3 V. They tolerate 5 V momentarily, but do not apply 5 V levels for extended periods.
- **Interrupts and pin muxing:** All I/O pins except D2 support interrupts, PWM, I2C, and 1-wire.

3.2.1 Boot-Related Pins

D8 (GPIO0) and D9 (GPIO2) are internally pulled up, while D10 (GPIO15) is pulled down; these three pins determine the boot mode when the module powers on. When designing external circuits, avoid driving them to abnormal levels at power-up, or the program may fail to run.

3.3 Power Supply

- DC input: powered via the DC jack with a 5–12 V input, regulated by the on-board switching supply to 3.3 V.
- USB power: powered directly through the Micro-USB port, which also handles program download and debugging.
- Output capability: the on-board 5V rail supplies about 1 A. Power motors, relays, and other high-current peripherals from a separate supply.

3.4 On-Board LED

The board carries one user LED. In the Arduino environment it is recommended to control it via the LED_BUILTIN macro instead of memorizing the pin number; the macro maps to GPIO2 of the ESP8266, i.e. pin D9 on the board silkscreen.

4 Mechanical Dimensions

The outline dimensions of the board are shown below. It matches the standard Arduino UNO form factor, so common UNO acrylic enclosures and expansion shields fit directly.

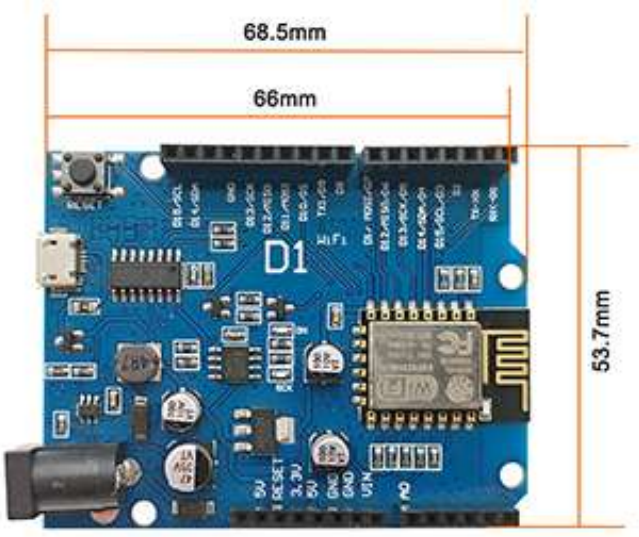

Figure 4-1 Outline dimensions

Item	Dimension
Board length (vertical)	53.7 mm
Board width (PCB body)	66 mm
Board width (with side I/O headers)	68.5 mm

Table 4-1 Mechanical dimensions

5 Setting Up the Development Environment

This chapter walks through the complete process of setting up the ESP8266 D1 environment from scratch, using the Arduino IDE as an example.

5.1 Prerequisites

- ESP8266 D1 Board × 1;
- A Micro-USB cable that supports data transfer × 1 (some cables only supply power and cannot download programs);
- A computer running Windows / macOS / Linux × 1.

5.2 Installing the Arduino IDE

Download and install the latest Arduino IDE from the official Arduino website (both 1.8.x and 2.x work). Allow the installer to install the related drivers.

5.3 Adding the ESP8266 Board Package

1. Open the Arduino IDE and go to File → Preferences.
2. Paste the following URL into "Additional Boards Manager URLs" and click OK:

```
https://arduino.esp8266.com/stable/package_esp8266com_index.json
```

3. Open Tools → Boards → Boards Manager and search for esp8266.
4. Locate "esp8266 by ESP8266 Community", install it, and wait for the download to finish.

5.4 Selecting the Board and Port

After installation, configure the following items in the Tools menu:

Setting	Recommended value
Board	LOLIN(WEMOS) D1 R1
Port	COM port assigned to the CH340 (Windows)
Upload Speed	115200
Flash Size	4M (4MB)
CPU Frequency	160 MHz (or 80 MHz)
Debug port	Disabled

Table 5-1 Recommended board and port settings

5.5 Installing the USB Serial Driver

The board uses a CH340C USB-serial chip. If your computer does not recognize the port, install the CH340 driver; after installation, unplug and replug the USB cable and a new serial device will appear in Device Manager.

6 Compiling and Flashing

6.1 Connecting the Board

Connect the board to your computer with a Micro-USB cable. Once connected, the board's power LED lights up and a serial device appears on the computer.

6.2 Compiling a Sketch

Click the Verify/Compile button (✓) on the left of the toolbar, and the IDE will check and compile the current sketch. When compiling finishes, the status bar at the bottom shows how much Flash and RAM the sketch uses.

6.3 Flashing the Program

With the correct board and port selected, click the Upload button (→) in the toolbar. The IDE compiles and then downloads the sketch; when the status bar shows "Done uploading", flashing is complete and the board restarts automatically to run the new program.

- **Tip:** If the upload fails, try holding the RST button while clicking Upload, or unplug and replug the USB cable and retry.

6.4 OTA (Over-the-Air) Upload

This board supports OTA (Over-The-Air) upload. After the first USB flash of a sketch with OTA support and connecting the device to Wi-Fi, you can select the network port in the Arduino IDE Port menu and flash wirelessly without a cable — especially convenient for devices already installed in the field.

7 Quick Start: Blink the On-Board LED

This chapter uses a simple on-board LED blink example to help you quickly verify that the development environment is configured correctly.

7.1 Example Code

```
// ESP8266 D1 UNO R3 -- On-board LED blink example
// The on-board LED is connected to GPIO2; in Arduino you can simply use LED_BUILTIN

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);     // Set the LED pin as an output
}

void loop() {
  digitalWrite(LED_BUILTIN, LOW);   // LED on (active low)
```

```
  delay(500);                       // Hold for 500 ms
  digitalWrite(LED_BUILTIN, HIGH);  // LED off
  delay(500);                       // Hold for 500 ms
}
```

- **Note:** On ESP8266 boards the on-board LED is usually active-low, so digitalWrite(LED_BUILTIN, LOW) turns it on and HIGH turns it off. For an external LED, wire it as pin → current-limiting resistor → LED → GND.

7.2 Step-by-Step Instructions

1. Set up the development environment as described in Chapter 5, and make sure the board and port are selected correctly.
2. Copy the example code above into the Arduino IDE editor and save it.
3. Click the Upload button and wait for the download to finish.
4. After a successful upload the board restarts automatically and the on-board LED blinks at about 1 Hz, which confirms that the environment is configured correctly.

7.3 Serial Debugging

To view runtime information, use the serial print feature. Example code:

```
void setup() {
  Serial.begin(115200);             // Initialize serial at 115200 baud
}


void loop() {
  Serial.println("Hello ESP8266!");  // Print a string over serial
  delay(1000);
}
```

After uploading, open the Serial Monitor at the top-right of the Arduino IDE, set the baud rate to 115200, and you will see the message printed continuously.

8 Precautions

1. Logic levels: all I/O pins operate at 3.3 V. They tolerate 5 V momentarily, but sustained 5 V levels may damage the module.

2. Power budget: the on-board 5V rail supplies about 1 A. Power motors, relays, and other high-current peripherals separately to avoid board resets.

3. Boot pins: GPIO0 (D8), GPIO2 (D9), and GPIO15 (D10) determine the boot mode at power-up; do not drive them to abnormal levels during startup.

4. Antenna area: do not cover the antenna area of the ESP8266 module with metal or large copper planes, or Wi-Fi signal strength will degrade.

5. ESD protection: take anti-static precautions before handling the board to avoid damaging the chips.

9 FAQ

Symptom	Possible cause and solution
Computer does not recognize the serial port	The CH340 driver is not installed — install it; or the USB cable only supports charging — use a data-capable cable.
Upload fails / stuck at "Connecting..." for a long time	Check that the correct board and port are selected; close other software occupying that serial port; unplug and replug the USB cable and retry.
On-board LED does not light	Make sure the sketch was uploaded successfully, and that the code uses LED_BUILTIN or the correct GPIO number.
Cannot connect to Wi-Fi	The ESP8266 supports only the 2.4 GHz band; make sure the router has the 2.4 GHz network enabled.
Board keeps resetting	Usually caused by insufficient power. Power the peripherals from a separate supply, or use a USB port with a higher current capacity.

Table 9-1 Troubleshooting quick reference

Revision History

Version	Date	Description	Author
V1.0	2026-07	Initial release	—