

# **ESP32-POE2**

## **User Manual**

**Document revision 2.7, October 2025**

**[www.olimex.com](http://www.olimex.com)**

# Table of Contents

1. Introduction to ESP32-POE2.....	3
ESP32-POE2 features.....	4
PoE standard.....	5
Differences between ESP32-POE2 and ESP32-POE-ISO.....	6
Order codes for ESP32-POE2 and accessories.....	7
2. Hardware.....	8
ESP32-POE2 layout.....	8
ESP32-POE2 pins and connectors.....	9
Power sense and battery level measurement.....	10
Buttons.....	11
SD card connector (1-bit SD interface).....	12
UEXT connector.....	13
EXT1 connector.....	15
Power supply.....	16
GPIOs.....	16
ESP32-POE2 schematics.....	17
ESP32-POE2 powering details.....	18
3. Dimensions.....	19
4. Software.....	20
5. FAQ (Frequently Asked Questions).....	22
6. Document revision history.....	24

# 1. Introduction to ESP32-POE2

[ESP32-POE2](#) is an upgraded version of the popular [ESP32-PoE](#) board with more power output capabilities. ESP32-POE2 is an IoT based on ESP32-WROVER-E WIFI/BLE/Ethernet development board with Power-Over-Ethernet feature. The [ESP32-POE2](#) allows 25W power negotiation and can provide 24V/12V/5V/3.3V to additional equipment connected to it.

The PoE in [ESP32-POE2](#) is handled by TPS2378 chip that is IEEE 802.3af-compliant, including pre-standard (legacy) PoE support.

The PoE powering requires at least 37V DC to operate successfully. The board can take power from the Ethernet cable and can be expanded with sensors and additional peripherals. Perfect solution for Internet-of-Things projects.

The board is considered Open Source Hardware, design files are released under CERN Open Hardware Licence Version 2 - Strongly Reciprocal. Design files can be found at our [GitHub](#).

Software is released under GPL V3 License and documentation is released under CC BY-SA 4.

**+ Important notice:** [ESP32-POE2](#) has **no galvanic isolation** on Ethernet's power supply, this might be a problem for ground loops. This also means that when you program the board via the micro USB connector the Ethernet cable should be disconnected (if you have power over the Ethernet enabled)! Any board with own external power supply attached to ESP32-POE2 can be dangerous. Consider using Olimex [USB-ISO](#) to protect your computer and board from accidental short circuit.

## ESP32-POE2 features

- ESP32-WROVER-E-N4R8 – WiFi and bluetooth module with 4MB flash, 8MB PSRAM
- CE-RED and LVD certification
- Original design by OLIMEX Ltd
- Low power design – 200uA consumption in deep sleep
- Ethernet 100Mb interface with IEEE 802.3 PoE support
- USB-C connector for ESP32 programming
- MicroSD card working in 1 bit mode (3 more GPIOs)
- LiPo battery charger with LiPo battery connector
- Battery level monitor pin on ADC
- External power supply detection pin on ADC
- Total output for external circuits – 25W max, max power distributed as follows:
  - 0.75A at 24V or 1.5A at 12V (selectable by jumper);
  - 1.5A at 5V;
  - 1A at 3.3V.
- [UEXT](#) connector
- EXT connector
- User button
- Reset button
- PCB dimensions: (59 x 90)mm ~ (2.32 x 3.54)in

## PoE standard

[ESP32-POE2](#) follows the original IEEE 802.3af PoE standard and provides up to 25 W of DC power (PoE requires minimum 44 V DC and 350 mA). Only 23 W is assured to be available at the powered device as some power dissipates in the cables.

The board is configured to work in PoE type 2 class 4 mode (PoE+ or IEEE 802.3at) – it draws 12.95 at minimum up to 25.50W maximum. Notice that network equipment usually reserves the maximum wattage for the port, meaning your switch will likely reserve 30W for each ESP32-POE2 connected to it.

The board requires networking equipment that complies with IEEE 802.3at. It is **STRONGLY** recommended to use **isolated PoE equipment**.

ESP32-POE2 uses LAN connector that is compatible with both mode A or mode B power delivery method.

## Differences between ESP32-POE2 and ESP32-POE-ISO

ESP32-POE-ISO has limited output voltage range. There are only 5V and 3.3V outputs available, while ESP32-POE2 has also 12V DC or 24V DC.

ESP32-POE2 operates in PoE type 2 class 4 mode (PoE+ or IEEE 802.3at), whereas ESP32-POE-ISO operates in PoE type 1 class 0 mode (PoE or IEEE 802.3af).

[ESP32-POE2](#) is **NOT** galvanically isolated which means that it's not safe to connect it to other devices which use non-isolated power supply. ESP32-POE-ISO is galvanically isolated.

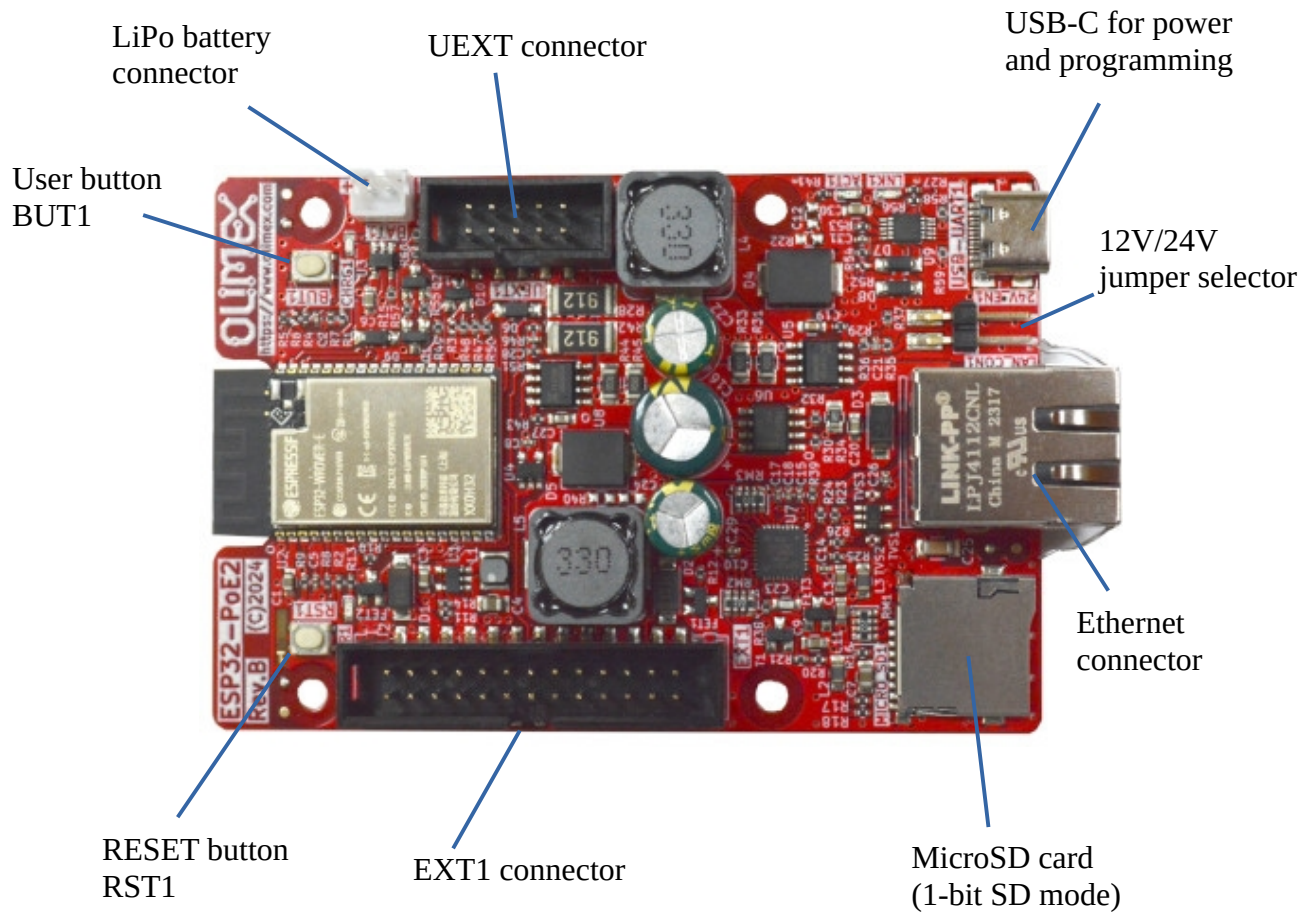
- + YOU SHOULD NOT CONNECT ESP32-POE2 to computer's USB port while it's powered by Ethernet POE!!! If you connect USB while ESP32-POE2 is powered by Ethernet you will damage the board or your computer or both. This also voids the warranty!
- + The board is susceptible to grounds loops. Make sure your PoE network equipment is isolated. Measure for voltage difference between different grounds before attaching devices to ESP32-POE2.
- + The most optimal setup for the board is being connected just to the PoE Ethernet cable and all attached peripherals are attached only to ESP32-POE2. No sources of power with different grounds should be attached at the same time. Additional boards that are attached to ESP32-POE2 and are also attached to other sources of power supply (being powered from other sources) can also cause problems!

## Order codes for ESP32-POE2 and accessories

<a href="#"><u>ESP32-POE2</u></a>	Commercial temperature grade 0-70C board with internal antenna
<a href="#"><u>USB-CABLE-A-TO-C-1M</u></a>	1 meter USB-A to USB-C cable for ESP32-POE2 powering and programming
<a href="#"><u>Ethernet-CABLE-1M</u></a>	1 meter Cat 5e Gigabit Ethernet cable with unshielded RJ45 connectors
<a href="#"><u>BOX-ESP32-POE2</u></a>	Plastic box for ESP32-POE2
<a href="#"><u>USB-ISO</u></a>	Galvanic USB isolator to protect your PC from electrical problems with ESP32-POE2
<a href="#"><u>BATTERY-LIPO1400mAh</u></a>	Lipo battery 3.7V 1400mAh – note these batteries can be shipped only by ground so we can deliver only to EU destinations.
<a href="#"><u>UEXT modules</u></a>	Different sensors, relays, LCDs, RTC, GSM, GPS, accessories which can be connected to UEXT connector

## 2. Hardware

### ESP32-POE2 layout

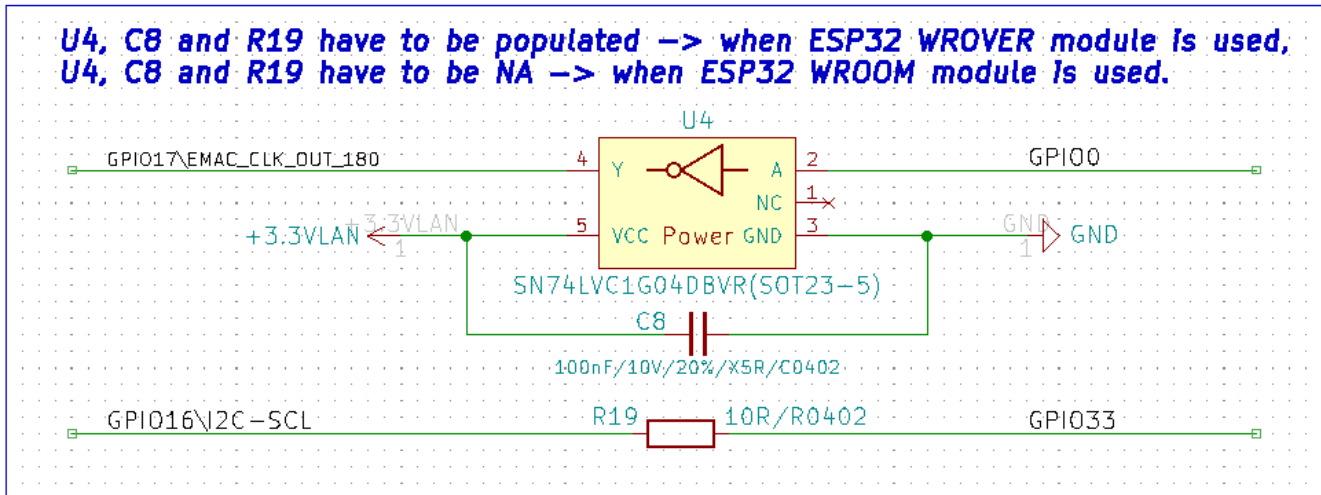




## ESP32-POE2 pins and connectors

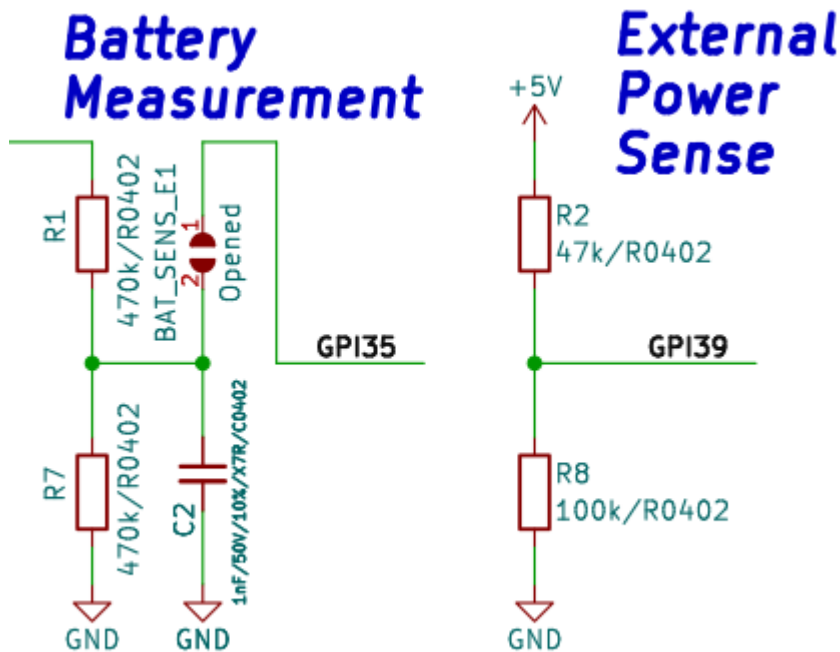
- + ESP32 chip and modules have very good pin multiplexing. You can define any free GPIO pin for almost any goal.
- + ESP32-POE2 uses WROVER chip, and it uses GPIO16 and GPIO17 for PSRAM. This is important when you are defining the pins for the Ethernet or using GPIO16 at the UEXT.

ESP32-POE2 with WROVER module remaps GPIO16 to GPIO33 (so, for example, at UEXT pin #5 you have GPIO33 instead of GPIO16) and GPIO17 to GPIO0 (this is the Ethernet clock source).



## Power sense and battery level measurement

- External power sense available on GPI39
- Battery measurement disabled by default (for an additional free GPIO), but can be enabled by SMT jumper change
- Close SMT jumper BAT\_SENS\_E1 (solder its pads together) to enable battery measurement on GPI35.



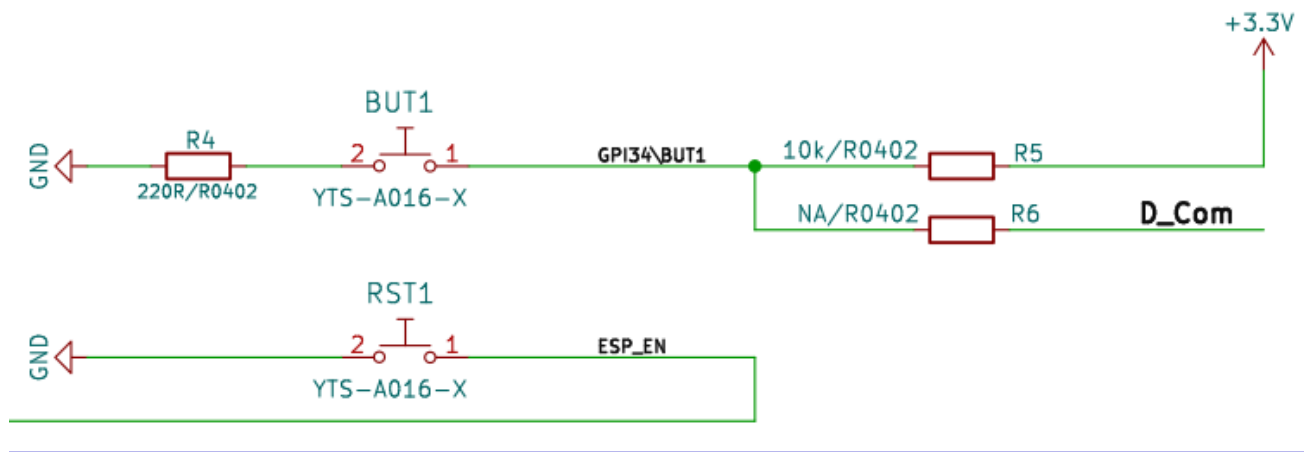
Notice about the battery measurement resistor divider coefficient is  $2 = (R1+R7)/R7$  and maximum battery charger voltage is around 4.2V, this leaves ADC values up to 2.1V. This means VREF needs to be set in ADC\_ATTEN\_DB\_11 range (150 mV ~ 3100 mV).

## Buttons

The ESP32-POE2 board has two buttons. One user button BUT1 and a reset button RST1.

By default button BUT1 is a user button since the board automatically enters bootloader mode. It is connected to GPI34 (GPIO34 can only be input, hence we've named it GPI34). Notice that BUT1 can be configured as on-demand BOOT button if you solder R6 pads together (or if you solder 0R resistor in 0402 package on same R6 pads). BOOT button can then be used to manually force boot mode.

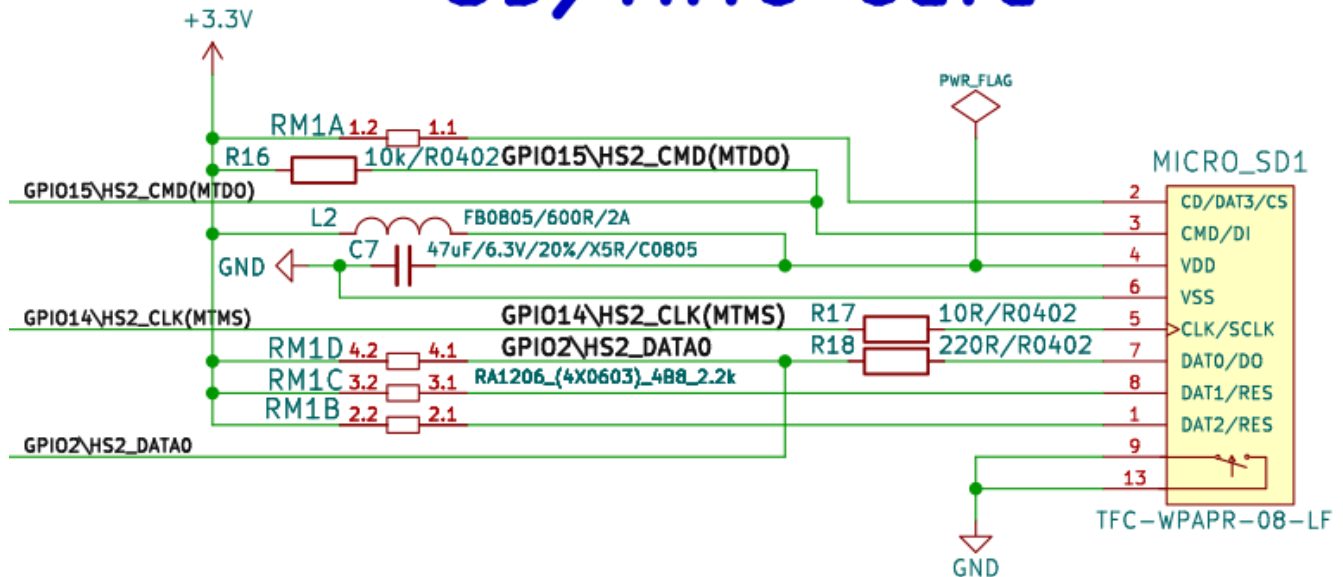
## Buttons



## SD card connector (1-bit SD interface)

Notice that the SD card pins are multiplexed with pins #7, #8, #9 of the UEXT (typically used for SPI). If you wish to use those pins at the UEXT you'd have to time share with the SD card using the different chip selects. It is important to notice the SD card is configured in 1-bit SD mode when considering the software.

# SD/MMC Card



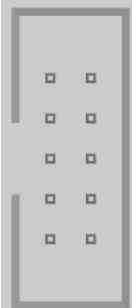
## UEXT connector

UEXT connector stands for Universal EXTension connector and contain +3.3V, GND, I2C, SPI, UART signals. It is usually used to attach external modules without soldering.

UEXT is 0.1" 2.54mm step boxed plastic connector. All signals are at 3.3V levels. GPIO16 is replaced by GPIO33 in boards with ESP32 WROVER module (all ESP32-POE2 boards use WROVER, so this is always true).

Pins #7, #8, #9 are multiplexed with the SD card, if you don't have or you don't need SD card it is fine to use, else you'd need to time-share the functionality between the UEXT and the SD card (using the different chip selects).

UEXT

RESISTOR <sup>1</sup>	MUX <sup>2</sup>	Signal	Pin#		Pin#	Signal	MUX <sup>2</sup>	RESISTOR <sup>1</sup>
-	-	3.3V OUT	1		2	GND	-	-
NO	NO	GPIO4	3		4	GPIO36	NO	10K
2.2K	NO	GPIO33 <sup>3</sup>	5		6	GPIO13	NO	10K
10K	SD CMD	GPIO15	7		8	GPIO2	SD DATA	2.2K
NO	SD CLK	GPIO14	9		10	GPIO5	NO	10K

<sup>1</sup>Some pins have pull-up resistors, that might affect the levels or readings.

<sup>2</sup>All pins are also routed and available at the EXT1 connector.

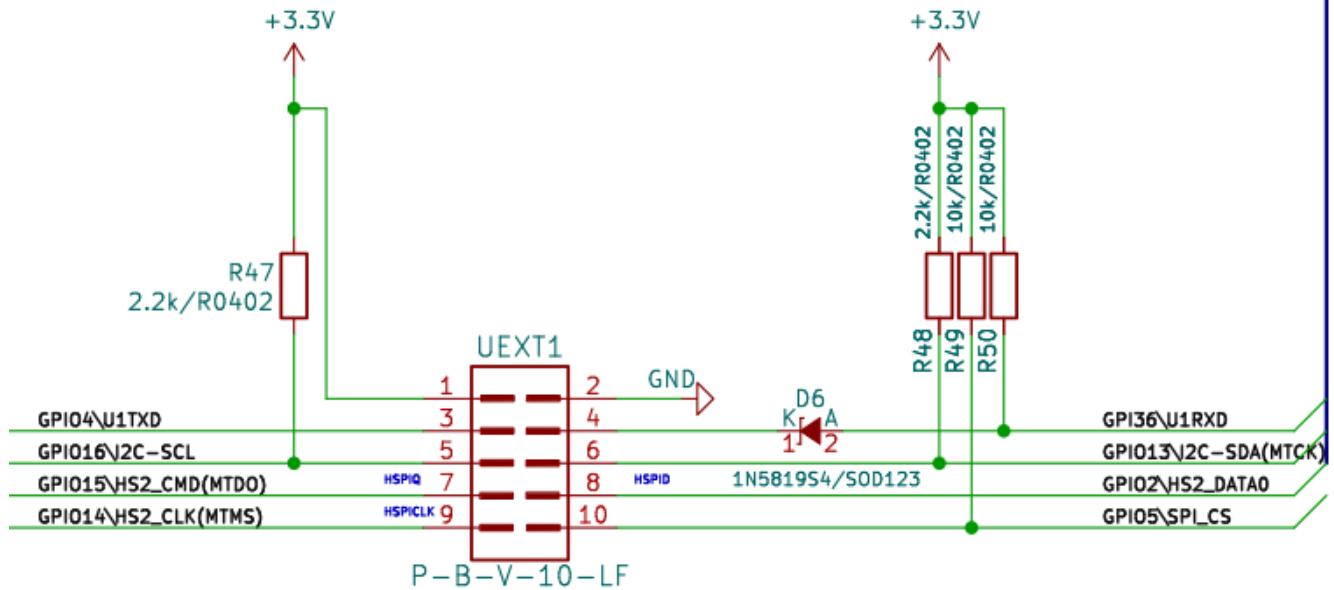
<sup>3</sup>By default ESP32-POE2 is manufactured with ESP32-WROVER module, for ESP32-WROOM module pin #5 would be routed to GPIO16.

These are suggested usage for the pins for compatibility with Olimex-made modules, but since ESP32 chip has good multiplexing they can be defined for other uses. Notice that you need to define the pins in your code (e.g. define them via software means) for the intended usage. For example, if you use GPIO33 for I2C SCL and GPIO13 for I2C SDA you need to define them in your code (in Arduino IDE it is usually done with "Wire.begin (13, 33);").

Olimex has developed number of [MODULES](#) with this connector. There are temperature, humidity, pressure, magnetic field, light sensors. Modules with LCDs, LED matrix, Relays, Bluetooth, Zigbee, WiFi, GSM, GPS, RFID, RTC, EKG, sensors and etc.

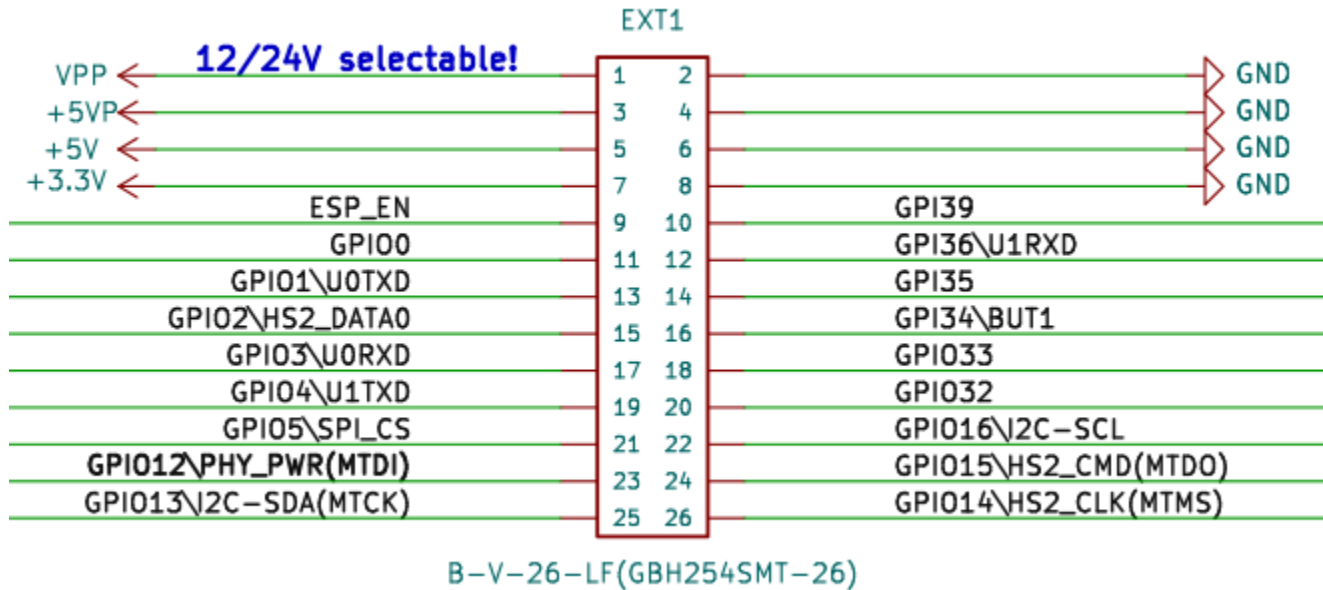
You can find excerpt from the schematic of the UEXT on the next page.

# UEXT



Important: GPIO16 is replaced by GPIO33 due to the board using WROVER chip (the schematic is made for both variants, but WROVER variant was never manufactured).

## EXT1 connector



Again notice about GPIO33 and GPIO16, when the WROVER module is used (by default), there is no GPIO16! And GPIO33 goes to both pin #18 and pin #22!

Same for GPIO0 – it will be used for Ethernet clock, if you decide it for other uses it will affect the Ethernet!

## Power supply

**VPP** selectable 12V/1.5A or 24V/0.75A output from PoE, depending on 24V\_E1 PTH jumper state

**+5VP** 5V/1.5A output from PoE

**+5V**

1. can be output or input;
2. when board is connected to USB or to Ethernet PoE this line can be used as output and power other circuits;
3. if you want to use as input i.e. to power the board from external 5V on this pin sure board is not connected to the USB!

**+3.3V** output which can source up to 0.5A @ 3.3V i.e. (1.65W)

## GPIOs

**ESP\_EN** resets ESP32 module;

**GPIO0** is used for Ethernet clock;

**GPIO16, GPIO17** are used for the PSRAM;

**GPIO1** is used for the serial line and programming;

**GPIO2, GPIO14, GPIO15** are used for the SD-card, if no SD card is present these are free to use, else you can time-share using the chip select;

**GPIO2, GPIO4, GPIO5, GPIO13, GPIO14, GPIO15, GPIO33, GPIO36** are shared on both UEXT and EXT1,2 headers so if you use them at one connector do not use at the other;

**GPIO33** is routed in place of GPIO16 at the header and the UEXT (since the board uses WROVER chip);

**GPIO39** is used to measure external power supply voltage;

**GPIO34** is connected to used button and have 10K pullup;

**GPIO35** is free to use but can be connected to measure the LiPo battery voltage if SENS\_BAT\_E1 solder jumper.

Notice that pins named GPI\* can only be inputs (unlike GPIO\*). This is limitation from the ESP32 chip itself.



## ESP32-POE2 schematics

[ESP32-POE2](#) latest schematic is on [GitHub](#)

## ESP32-POE2 powering details

[ESP32-POE2](#) can be powered by 4 sources:

- Ethernet PoE, notice this is class 4 PoE device requires at least 12.95W and can go up to 25.5W. **Notice that your network equipment will allocate the maximum wattage for the port, usually around 30W per device; this is important if you connect multiple ESP32-POE2 boards to the same piece of PSE (Power Sourcing Equipment)!**
- USB-C connector
- LiPo battery
- EXT1 pin 5 (+5V) note that this signal is connected to USB 5V signal so when you power with this pin you should not connect the board to the USB! If possible avoid powering from EXT1 to avoid accidental short-circuit!

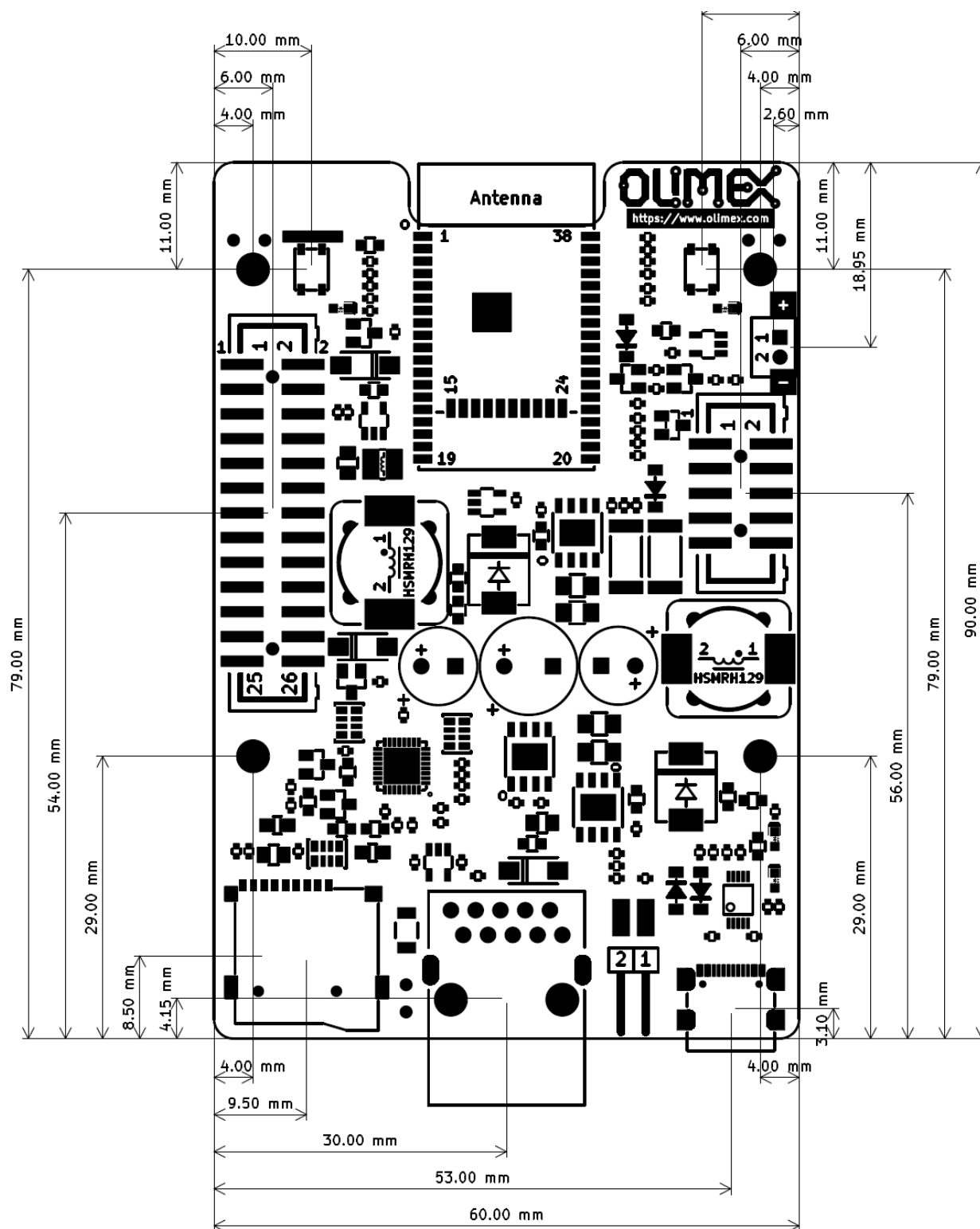
Power consumption of [ESP32-POE2](#) is between 50mA and 200mA depending on the operation mode.

A LiPo battery gets charged automatically from other sources of power with about 100mA.

When the LiPo battery is attached and external power supply is missing internal DCDC step-up converter and switching circuit automatically powers ESP32-POE2 from the battery. 1400mAh battery will provide about 8 hours of stand alone operation.

+ The LiPo battery connector is JST 2.0 mm connector and with Olimex's battery polarity. If you use batteries from other manufacturers please make PLUS and MINUS are connected properly else you will damage the board!!!

### 3. Dimensions



## 4. Software

[ESP32-POE2](#) uses same software as ESP32-POE which is very popular board and supported by

- [Espressif ESP-IDF](#)
- [MicroPython](#)
- [Arduino IDE](#)
- [Esphome](#)
- [PlatformIO](#)

**In your software (for example, Arduino IDE) make sure to enable the PSRAM in the board selector (when board is with ESP32-WROVER-E)!**

Software notices:

- Ethernet notice – For the Ethernet operation make sure to properly set the GPIO for the Ethernet clock and the pin for the power enable pin! GPIO12 is used as PHY power pin and GPIO0 is used as clock output pin (when board uses ESP32-WROVER-E module). Under Arduino defines should be like (must be defined before ETH.h):

```
#ifndef ETH_PHY_TYPE
#define ETH_PHY_TYPE    ETH_PHY_LAN8720
#define ETH_PHY_ADDR    0
#define ETH_PHY_MDC23
#define ETH_PHY_MDIO    18
#define ETH_PHY_POWER   12
#define ETH_CLK_MODE    ETH_CLOCK_GPIO0_OUT
#endif

#include <ETH.h>
```

- I2C (TWI) notice – since the board uses WROVER chip to define the I2C pins at the UEXT connector use GPIO13 for SDA and GPIO33 for SCL. For example, to define in Arduino IDE use:

```
Wire.begin(13, 33);
```

Remember to include <Wire.h> at the beginning:

```
#include <Wire.h>
```

## 5. FAQ (Frequently Asked Questions)

- **Help! Ethernet is not working always upon power-up. If I press the reset or power cycle the board a few times sometimes it starts working. Is my device faulty?**

It is probably because your software didn't utilize the PHY\_PWR pin (GPIO12) properly. This behavior happens when the PHY controller (LAN8720) gets powered before the ESP32 module can generate the PHY clock. What your software should do is hold the LAN8720 in reset for few milliseconds via the PHY\_PWR pin and release it from reset only after PHY clock is already generated by the ESP32 module. Simple software workarounds that can be tested are adding delay before Ethernet initialization or toggling GPIO12 to reset only the Ethernet chip until connection is restored.

- **Ethernet doesn't work. Is my board defective?**

Make sure you've selected board variant with PSRAM enabled (this board has WROVER module, not WROOM). Make sure your code defines GPIO0 as Ethernet clock source pin.

- **Where are the I2C, UART, SPI pins exposed?**

The ESP32 chip has very advanced multiplexing and you can define any free GPIO pin for I2C, UART, SPI operation as long as you are within the maximum supported (some penalties to SPI's maximum frequency apply, when not using the dedicated pins). Notice that some ESP32 pins can only be inputs. Double check if the pins you want to use are free. Defining pins for another function is a purely software effort.

- **I want to use the I2C pins at the UEXT but I have problems. I have defined pins 13 and 16 for the I2C. What is the problem?**

The I2C pins at the UEXT are 13 and 33. Pin 16 gets replaced by pin 33 due to the board using WROVER chip (instead of WROOM). Refer to [page 9](#).

- **I provide 24V to the Ethernet of ESP32-POE2's but it doesn't seem powered. What is the problem?**

TPS2375PW (Si3402) would NOT work with 24V DC. The recommended voltage is 48V DC and the minimum is around 37V DC. For more info refer to TPS2375PW's (Si3402-B's) datasheet.

- **I power the board from the battery connector. The LEDs remain off. Is it broken?**

Probably not. This is a low-power design. The LEDs would not turn on when operating on battery to save power. You need other ways to determine if it works or not. For example, something over the serial lines or over the Ethernet (with no PoE enabled else it would get powered from there).

- **What is the power delivery mode of these boards? Is it mode A or mode B?**

ESP32-POE2 uses LAN connector that is compatible with both mode A or mode B. Either one is fine.

- **What is the PoE class of ESP32-POE2?**

ESP32-POE2 is set for type 2 class class 4 PoE operation (12.95W-25.5W). Class 4 operation might require more current since it requires at least 12.95W up to 25.5W. **Notice that your network equipment will allocate the maximum wattage for the port, usually around 30W per device; this is important if you connect multiple ESP32-POE2 boards to the same piece of PSE (Power Sourcing Equipment)!** If you have trouble with powering multiple devices from the same switch then refer to the schematic and unsolder the resistor R34 to switch back to class 0 operation. (0.44W to 12.95W), but this will reduce the wattage available from the board, roughly half of the wattage only will be available.

- **R28 and R42 get very hot! What are these components? Are they safe? Can I remove them?**

These resistors are required by the IEEE standards. They are used to maintain power signature (MPS). Search online "maintain power signature poe" to get the general idea.

Lack of load causes availability problems when using ESP32 power saving modes – the board would consume less wattage than PoE equipment can detect, which leads to PoE equipment shutting the ESP32 board off.

If you are not using the low power modes of the ESP32 board and the heat from the resistor bothers you, you might try desoldering R28 and R42 and see if that effects your setup negatively. If the board gets shut down, solder them back. Notice that we don't recommend doing so.

## 6. Document revision history

Revision 2.7 October 2025

- Expanded FAQ section;
- Reworked UEXT connector table;
- Expanded I2C clarifications (GPIO16 → GPIO33 due to WROVER chip);
- General minor improvements.

---

Revision 2.6 September 2025

- Clarified multiplexing and cleared UEXT section

---

Revision 2.5 September 2025

- Added FAQ section

---

Revision 2.4 March 2025

- Minor improvements and fixes

---

Revision 2.3 August 2024

- Additional notes about software setup for Ethernet and board selection

---

Revision 2.2 May 2024

- Fixed mistakes in GPIO descriptions

---

Revision 2.2 May 2024

- Fixed mistakes in GPIO descriptions

---

Revision 2.1 May 2024

- Added dimensions;
- Spelling improvements.



---

Revision 2.0 May 2024

- Fixed wrong info about ESP32 module – the board is manufactured with WROVER module, not WROOM;
- Clarified warnings about isolation;
- Added some extra info about different GPIOs;
- Spelling improvements.

---

Revision 1.0 April 2024