WIZNET MAGAZINE

EOE mag

Edge Offload Ethernet

EDITION 01

THE CONNECTED EDGE



HYBRID CONNECTIVITY WITH W5500 + ESP32

How Espressif and WIZnet are shaping wired, wireless, and hybrid embedded systems.

EDGE OFFLOAD IN PRACTICE

W5500 takes the Ethernet role; ESP32 handles local intelligence and wireless.

INSIDE THIS ISSUE

- **One Pairing, Many Edges** - W5500 + ESP32 roles
- **The Connected Edge Atlas** - Six application clusters
- **Different Links, Real Places** - Wired, wireless & hybrid
- **Beyond the Usual IoT Box** - Mobility, aviation & professional control
- **The Board Is Not the Product** - Drivers, libraries & reusable platforms

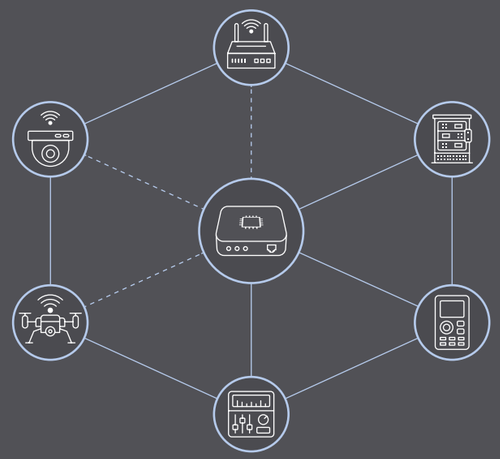
maker.wiznet.io

WIZnet

1

ONE PAIRING, MANY EDGES

In this edition, edge offload begins with clear role separation: W5500 takes the Ethernet role; ESP32 handles local intelligence and wireless.



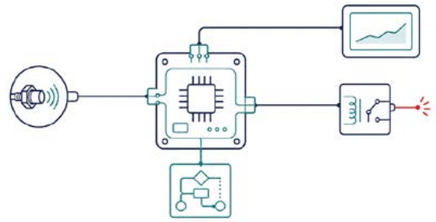
An embedded network now begins well before a link is selected. Espressif devices increasingly bring sensing interfaces, display and touch functions, control logic, and multiple radio choices into compact products. That concentration changes the design question.

Connectivity is not simply a software capability waiting to be enabled; every path occupies a physical position in a cabinet, room, gateway, receiver, or moving platform. WIZnet Ethernet devices can add a practical fixed-network role to that system without displacing the value of local wireless links. The result is a pairing that appears in many product forms and architectural patterns.

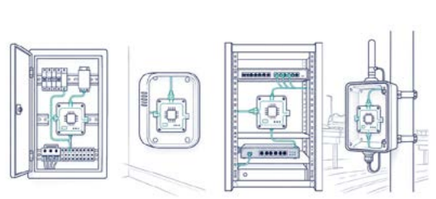
This edition maps that breadth: not as one reference design, but as a field of connected-edge possibilities.

1 Local intelligence is the starting point

The application remains close to the device because the device is doing more than joining a network. An Espressif MCU may read sensors, manage a display, interpret local inputs, maintain protocol state, and execute control decisions within the same product. Radio function is important, but it is only one part of that role. Local processing lets an endpoint remain an active participant in the system rather than becoming a thin attachment to a remote service.



2 Network decisions become physical

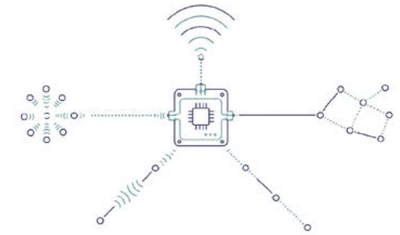


Once a network path reaches hardware, placement becomes an engineering decision. A controller inside an electrical cabinet faces different constraints from a room sensor, a gateway, an aviation receiver, or a vehicle prototype. Power availability, connector access, service procedures, enclosure limits, and the product's own reliability policy all shape the choice.

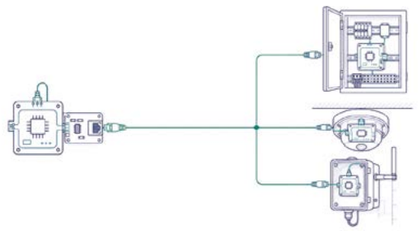
The network interface therefore influences more than data movement; it helps define where the device belongs and how it will be maintained.

3 Radio freedom still matters

Across Espressif-based systems, wireless options are not interchangeable, and none should be treated as a temporary substitute for a cable. Wi-Fi can connect to familiar infrastructure; Bluetooth LE (BLE) can support nearby devices and commissioning; Zigbee and Thread address low-power mesh roles; ESP-NOW enables direct device-to-device exchange; and an attached LoRa radio can extend a design into long-range, low-data-rate use cases. The productive question is not which medium wins. It is which link should carry each interaction, at each distance, under each operating policy.



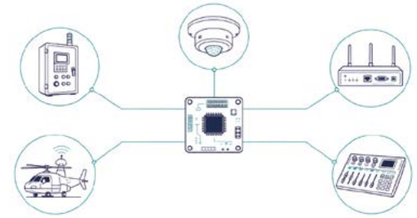
4 Fixed paths make systems placeable



A fixed Ethernet path gives designers another way to place traffic and responsibility. It may serve as a gateway uplink, a management connection for an installed endpoint, a route for fixed traffic, or one element of a fallback policy. It can also clarify an installation by giving a device an explicit network location. The software boundary still varies: some designs integrate Ethernet through the MCU's networking stack, while others use transport libraries or hardware-socket interfaces. Its application role must be read from the design, not assumed from the parts.

5 Applications are moving beyond familiar IoT categories

The pairing now appears wherever local computation, radio function, and fixed network access need to meet in a compact product form. That includes control systems coordinating field devices; Matter and radio gateways translating between local links and IP networks; installed sensors with serviceable network access; mobility and aviation receivers that combine radio data with on-board processing; professional control interfaces; and resilient network services designed around more than one available path. These categories do not describe a single architecture. They reveal how broadly the same component relationship can be reinterpreted.



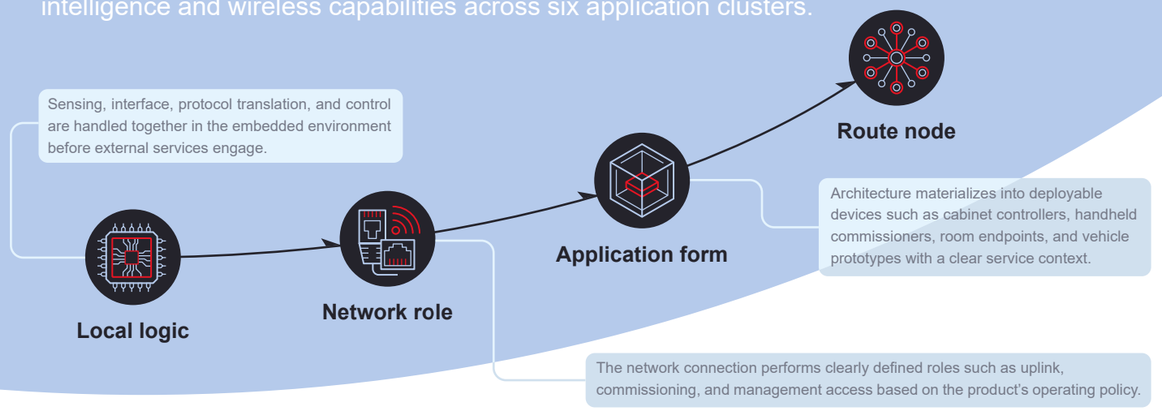
6 Why W5500 + ESP32?



This edition is a curated field guide, not a catalog and not a declaration that every connected device needs the same pairing. Its projects are representative signals: examples of engineers using Espressif and WIZnet parts to define control nodes, gateways, receivers, installed endpoints, libraries, and development platforms. Read together, they show where application boundaries are shifting and where components are taking on new roles. For design engineers and firmware developers, the value lies in architectural comparison. For makers, product teams, VARs, and ecosystem partners, it lies in recognizing directions that connect with a problem, product form, or development path.

The Connected **Edge Atlas** 2

One hybrid pairing combines W5500 Ethernet with ESP32 local intelligence and wireless capabilities across six application clusters.



An application atlas is more useful than a board-family index. Across these selected projects, the Espressif-WIZnet pairing starts with distinct demands: a control burden, a radio boundary, a sensing task, a receiver, or a reusable platform need. The route moves from what the device must interpret or decide locally to its network role, then into a form that can be installed, carried, serviced, or reused.

No single design captures the whole ecosystem or establishes a universal architecture. Read together, however, they reveal a widening set of placements for local computation, radio function, Ethernet access, and application-shaped hardware.

Six W5500 + ESP32 Application Clusters

Control & recovery

Control nodes combine local decisions with explicit access paths, while recovery provisions keep configuration and service options within reach.

- Ethernet Gate Controller PCB
- ESP32 Modbus-MQTT Bridge
- SMART K1

Radio & backhaul

Radio-facing systems assign different work to local wireless links, commissioning channels, translation layers, and fixed IP access.

- Tanmatsu Matter
- Art-Net to Zigbee Bridge
- lightyear_hub

Installed edge

Permanently placed endpoints turn sensing and metering into serviceable room, utility, and perimeter-level product forms.

- Sensy-One E1 Pro
- Batear
- Triple Water Meter

Mobility & mission systems

Receivers and vehicle prototypes join radio data, local control, and networked exchange inside constrained moving-system contexts.

- V2X2MAP
- Zonal ECU Prototype
- ADSBee

Professional interfaces

Purpose-built controls translate keys, gestures, signaling, and operator intent into network-aware media and communication functions.

- OSC-KEYPAD-ONYX
- OpenDeck
- SipServer

Resilience & developer paths

Libraries, routers, and official components make connectivity policy reusable beyond any single endpoint or finished application.

- EthWiFiManager
- newRouter
- espressif/w5500

The pattern is not repetition; it is architectural reuse under sharply different application pressures.

CONTROL

Ethernet Gate Controller PCB
UCC · Purpose-Shaped Control Board



Keywords

PCB · Dry Contacts · Control · ESP32 · W5500

Summary

Ethernet Gate Controller PCB concentrates a gate-control task into a dedicated board built around an ESP32-WROOM-32E, a W5500 Ethernet interface, and three dry-contact relay outputs. An earlier Raspberry Pi Pico, protoboard, and separate relay-board arrangement becomes application-shaped hardware. The ESP32 carries the HTTP API, relay logic, and update handling; W5500 provides the fixed Ethernet path over SPI.

Why This Matters

The signal is the consolidation itself: once the control task, outputs, power, and access path are understood, a module stack can give way to hardware organized around one physical responsibility.

RADIO

Tanmatsu Matter
UCC · Matter Commissioner



Keywords

Matter · ESP32-P4 · ESP32-C6 · Thread · Ethernet · UI

Summary

Tanmatsu Matter turns the handheld Tanmatsu into a screen-led Matter commissioner. The ESP32-P4 carries the application and user interface; the ESP32-C6 provides Wi-Fi, Bluetooth LE, and Thread radio functions. An optional W5500 module adds Ethernet on-network operation. It remains one interface choice within a broader design, and supported commissioning methods differ according to the selected transport.

Why This Matters

The project points toward embedded tools that coordinate display, input, multiple radio paths, and optional Ethernet in one field-facing form, without treating those interfaces as interchangeable across every commissioning workflow.

INSTALLED EDGE

Sensy-One E1 Pro Multi Sense
VAR · Installed Presence Endpoint



Keywords

mmWave · ESP32-C3 · W5500 · PoE · ESPHome · Home Assistant

Summary

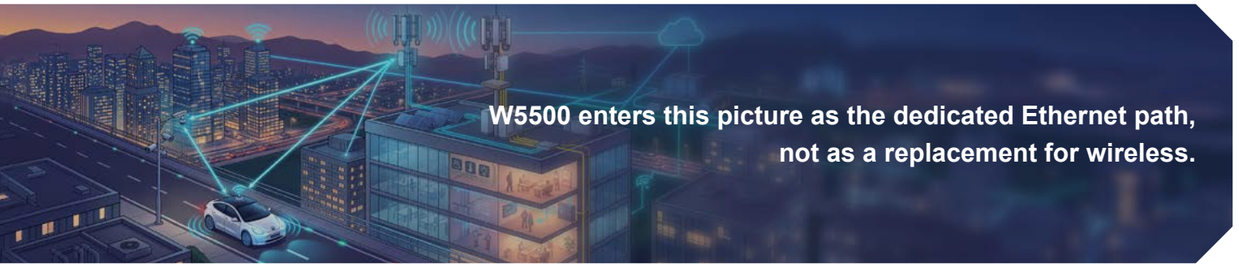
Sensy-One E1 Pro Multi Sense is an ESP32-C3-based room endpoint combining mmWave presence sensing with W5500 Ethernet. PoE can carry power with the network connection, while USB-C provides an alternative power path; separate Wi-Fi firmware is also available. ESPHome and Home Assistant place the device within a local automation environment shaped for fixed, room-level placement.

Why This Matters

It shows the pairing reaching a product form in which sensing, connection choice, power strategy, and automation integration are resolved around a specific installed room endpoint rather than a development bench.

3

DIFFERENT **LINKS,** REAL **PLACES**



W5500 enters this picture as the dedicated Ethernet path, not as a replacement for wireless.

At the connected edge, a link earns its place by the job it performs and its physical context. A room endpoint may need nearby commissioning. A control node may exchange messages without an access point. A gateway may listen to low-power sensors while forwarding selected data upstream.
These are not contradictions; they are why radio diversity matters.

W5500 enters this picture as the dedicated Ethernet path, not as a replacement for wireless. It can carry an uplink, expose management access, handle fixed traffic, or provide a service path when policy calls for it. The radio can remain central to sensing, discovery, local exchange, or field communication.

Installation makes the architecture tangible. Power must reach the device. Antennas, connectors, and sensors need workable positions.

Mounting and enclosure choices affect what remains accessible. Maintenance may happen in a cabinet, on a wall, above a ceiling, or at a remote fixed site. The story is system-level role assignment: each link, power path, and physical interface receives a purpose that survives beyond the bench.

Give Every Link the Right Job

Wi-Fi suits devices that join local infrastructure or present a configurable interface. BLE often handles nearby discovery, setup, or sensor collection.
Zigbee and Thread place low-power mesh devices within application fabrics, while ESP-NOW supports direct exchanges between Espressif nodes. LoRa extends sparse, low-data-rate communication across larger physical areas.
Ethernet can take the upstream, management, fixed-traffic, or service role. The useful architecture is not a stack assembled for maximum variety; it is a disciplined division of labor shaped by the application.

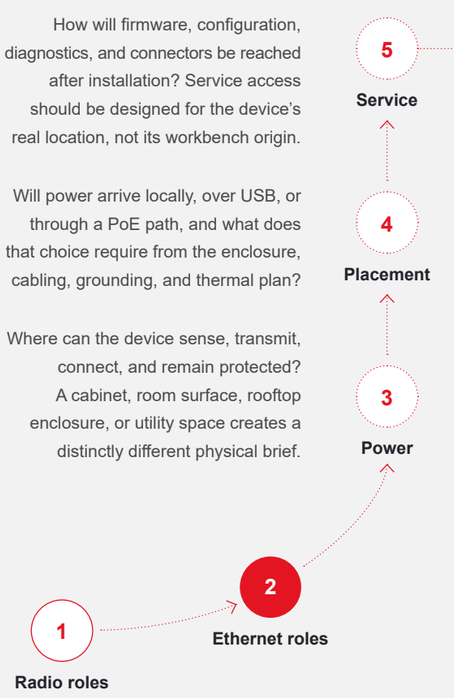
The EDGE finds a place

A networked design becomes an installed endpoint only when its electrical, mechanical, and service assumptions are resolved together. Placement turns connectivity from a capability into an operating condition, with consequences for ownership, access, and maintenance.

How will firmware, configuration, diagnostics, and connectors be reached after installation? Service access should be designed for the device's real location, not its workbench origin.

Will power arrive locally, over USB, or through a PoE path, and what does that choice require from the enclosure, cabling, grounding, and thermal plan?

Where can the device sense, transmit, connect, and remain protected? A cabinet, room surface, rooftop enclosure, or utility space creates a distinctly different physical brief.

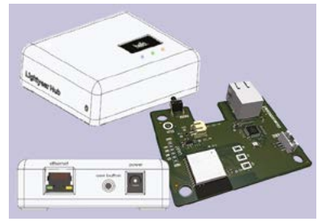


A connected edge is credible when every interface has purpose.

Three featured application signals

1 [Gateway] Lightyear Hub

UCC · Multi-Radio Sensor Gateway



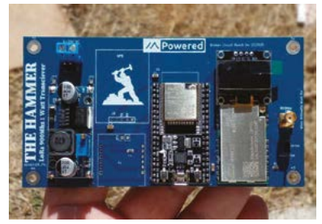
Role Sensor collection by radio; upstream exchange through Ethernet.

Summary Built around ESP32-C6, lightyear_hub collects data from BLE and Zigbee sensors, while its W5500 provides an Ethernet path toward an upstream service. Wi-Fi remains available, but the architecture separates sensor-side radio activity from the selected upstream route. A local web interface supports setup, status, control, logs, and custom BLE sensor definitions, giving the gateway an on-device operational surface for commissioning and ongoing local configuration work.

Why this? That separation gives each interface a responsibility: radios meet nearby devices, Ethernet serves the external path, and the ESP32-C6 coordinates both. The pattern can be adapted without pretending every sensor network needs one universal transport.

2 [Infrastructure] The Hammer

Development signal · fixed mesh infrastructure



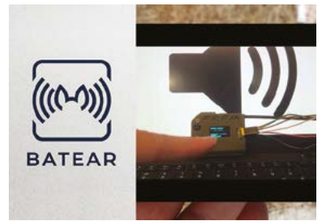
Role Positioning LoRa hardware for powered, fixed-site network placement.

Summary The Hammer combines an ESP32, an E22 LoRa radio, optional GPS, and a W5500 Ethernet module on a board intended for fixed LoRa installations. PoE-capable hardware points toward single-cable placement in an enclosure on a roof, tower, or other fixed site. The open repository includes board materials and firmware work, but its own notes distinguish general W5500 operation from unresolved Meshtastic Ethernet behavior in practice.

Why this? This is an infrastructure direction, not evidence of finished-product maturity: radio, positioning, power, and IP access are converging around a mounting location. Firmware support remains version-specific engineering work, not a blanket capability across all releases.

3 [Fixed sensing] Batear

UCC · Local Acoustic Sensing Node



Role Local acoustic processing; LoRa or Ethernet carries reporting.

Summary Batear uses an ESP32-S3 for local acoustic processing and offers distinct build roles from one codebase. A detector can send events over LoRa to a gateway, while a wired-detector build uses an ESP32-S3 Ethernet/PoE board to publish directly over W5500 Ethernet. This creates a distributed fixed-node concept in which sensing remains local and the reporting path can be chosen for the installation and service model.

Why this? The important idea is architectural choice, not a performance claim. A sensing node can retain the same local processing purpose while its outbound path changes to match distance, power availability, site access, and system organization.

Also on the map

Satellite1 PoE SHOE
DEVELOPMENT PREVIEW
Power plus networking.

SipServer
UCC
Ethernet/PoE supports embedded SIP signaling paths.

ESPHome Triple Water Meter
UCC
PoE supports metering.



4

BEYOND THE USUAL IoT BOX

1 · MOBILITY

V2X2MAP

UCC / DERIVED PROJECT
Mobility Receiver

These cases and development signals show W5500 + ESP32 hybrid connectivity expanding beyond familiar IoT into mobility, automotive, aviation, and professional control.

The familiar shorthand for embedded IoT, from sensor to dashboard to cloud, captures only a narrow slice of what these components now do. Mobility receivers, vehicle-scale prototypes, aviation systems, and professional control surfaces divide work more deliberately. One device may handle RF capture, another decoding; an Espressif device may coordinate application state or network management; a WIZnet interface may provide a fixed output path.

That division reflects operational context. A traffic receiver turns broadcast frames into map data. A zonal prototype separates local actuation from central control. An aviation platform distinguishes signal decoding from network services. A lighting interface translates physical input into a protocol used by professional software.

The projects here are community implementations, platforms, and prototypes, not evidence that one architecture is universal or production-standard. Their significance is categorical. Together, they show the Espressif-WIZnet relationship moving beyond familiar environmental sensors and home dashboards into systems shaped by mobility, mission context, operator workflow, and specialized data paths rather than generic device categories.

Case #1

V2X2MAP adapts OpenTrafficMap's receiver firmware for an ESP32-C5 development board, listening to 5.9 GHz ITS-G5 / IEEE 802.11p broadcasts and forwarding captured frames to a phone or bridge. Its Android application turns decoded traffic messages into a live map, logs, and optional MQTT output. The fixed-network precedent comes from upstream OpenTrafficMap, whose receiver architecture supports SPI Ethernet, including W5500, as an independent path for sending data from fixed receivers toward mapping and analysis services, rather than from the derived mobile package.

Why it matters

This case broadens mobility electronics from in-vehicle telemetry to external observation of traffic systems. Its value is not a deployment claim, but the separation of radio reception, local interpretation, and multiple paths toward map or analytics output.

Automotive Prototype

[Prototype]
Zonal ECU Prototype
Automotive Network Architecture



Case #2

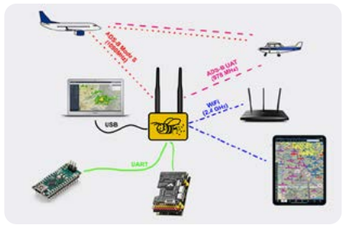
Zonal ECU Prototype models front and rear zonal nodes around ESP32 controllers, a central switch, and a central computing unit. W5500 interfaces place the zones on an Ethernet backbone, while local services retain motor-control and sensing responsibilities. The public implementation shows low-level Ethernet and UDP bring-up, with SOME/IP presented as an early direction in public code rather than a completed vehicle stack or production-oriented architecture.

Why it matters

The signal is architectural separation: local zones aggregate physical functions while a central layer coordinates behavior.
As a prototype, it supports examining boundaries and service organization, not claims of validation, AUTOSAR compliance, or vehicle readiness.

Aviation

[UCC / PLATFORM]
ADSBee
Aviation Receiver



Case #3

ADSBee is an open receiver platform that assigns signal decoding and network management to different processors.
The RP2040 uses programmable I/O to detect and decode ADS-B messages; the ESP32-S3 maintains network-facing aircraft data and manages Wi-Fi or Ethernet reporting.
A W5500-based interface is available as an external Ethernet option, extending the platform beyond USB and wireless output paths for downstream tools, services, and integration workflows.

Why it matters

This division keeps RF work, packet interpretation, and network services conceptually distinct. It shows how an aviation receiver can become an extensible embedded platform without implying certification, verified range, or any particular scale of deployment.

Professional Control

[UCC · Professional]
OSC-KEYPAD-ONYX
Lighting Interface



Case #4

OSC-KEYPAD-ONYX converts USB keypad events into network commands for ONYX lighting playback. The ESP32-S3 operates as a USB host, interprets HID key events, and maps them to OSC addresses.
Commands travel as UDP messages through a W5500 Ethernet interface. The result is a professional control node defined less by general-purpose connectivity than by a specific operator action, application protocol, and fixed network path for operators.

Why it matters

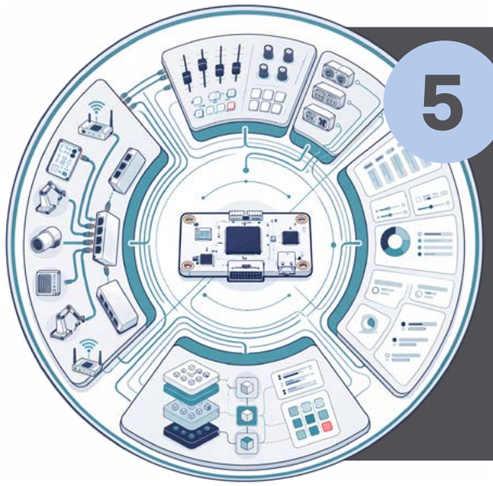
The project places the pairing inside a live-production workflow, where a control surface must speak the language of an established application.
Input handling, command mapping, and Ethernet transport remain distinct responsibilities within a purpose-shaped interface.



ALSO ON THE MAP

OpenDeck

Its configurable MIDI and OSC framework extends the category from one keypad to reusable control surfaces, sensor interfaces, and Ethernet-connected nodes. ESP32-S3 and W5500 targets place flexibility within this ecosystem.



5 The Board Is Not the Product

From edge offload to platform, W5500 + ESP32 designs become reusable through drivers, libraries, secure transport, and multi-path networking.

A project becomes more valuable when its role can be understood and repeated. The decisive move is not from prototype board to custom PCB.

It is from one instance to an architecture another team can interpret: what the device senses or controls, which link performs which job, how configuration is retained, and where responsibility sits.

Productization extends beyond the schematic. Interfaces must remain coherent as firmware changes. Installation assumptions need documentation. Configuration must survive the move from the bench into a serviceable device. Maintenance, diagnostics, update paths, enclosure access, and replacement strategy shape whether a design can live beyond its build.

The evidence in this edition sits at different layers. A UCC demonstrates a community implementation. A platform offers behavior that can travel across more than one product form. A prototype tests an architectural direction. A library packages a software decision for reuse. An official component defines a path within an ecosystem. These labels are not a maturity ladder, and they do not establish commercial lineage between unrelated projects.

Nor does one board choice define the field. Some designs use ESP-IDF Ethernet through esp_eth, esp_netif, and lwIP; others use Arduino Client-style transports or direct WIZnet APIs. The Overall Edition connects parallel signals rather than claiming a market map. Its value lies in showing where experiments begin to repeat, and where repeatability starts to resemble a platform.

Four productization questions

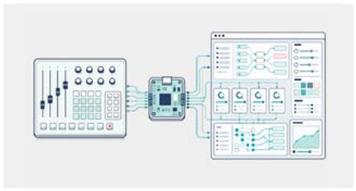
1. Can another engineer explain the application role without reconstructing the experiment?
2. Can the device be powered, placed, reached, and serviced after installation?
3. Can dependencies, configuration, updates, and diagnostics stay maintainable as software changes?
4. Can another team reproduce the architecture without inheriting assumptions or one-off fixes?

1 OPEN PLATFORM

OpenDeck

UCC / PLATFORM

MIDI and OSC Control Surface



The platform

OpenDeck treats a control surface as configurable behavior rather than a fixed panel. It maps buttons, encoders, analog inputs, outputs, and sensors into MIDI or OSC, with browser configuration online or offline.

Supported targets include ESP32-S3 boards paired with W5500 or W6100 Ethernet hardware, alongside other Espressif Ethernet targets. The interaction model can move across custom surfaces, Ethernet-connected OSC nodes, and installation-specific hardware.

Why this matters

A reusable platform preserves application intent while hardware choices change. OpenDeck shows how configuration, protocol behavior, and board support can become infrastructure rather than remain embedded in a control-surface build.

Repeatability begins when the architecture remains legible beyond its original maker.

Platform layers

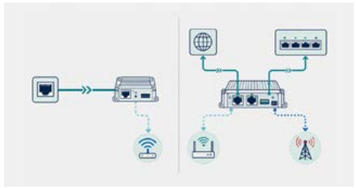
2 Resilience layer

Ethernet-First and Multi-Path Networking

UCC / LIBRARY + PROTOTYPE

Projects

ethwifimanager · newrouter



Summary

EthWiFiManager turns connectivity policy into an Arduino library: SPI Ethernet is preferred, with automatic Wi-Fi fallback when that link is unavailable. It uses ESP-IDF Ethernet drivers beneath its API. newRouter explores an ESP32-S3 architecture with two W5500 interfaces, a Wi-Fi access point, and cellular backup. One organizes link policy for maintenance and reuse; the other exposes a router prototype for multi-interface experiments.

Why this matters

Resilience is not one feature. A library can standardize link selection; a prototype can expose routing and interface questions without implying uninterrupted failover, production performance, or equivalence to a commercial router.

Eth WiFi Manager

newRouter

3 Secure and low-level paths

Above and Below the Socket

UCC / LIBRARY + RESEARCH

projects

ethernetclientsecure ·

esp32 × rust × w5500 macraw



Summary

EthernetClientSecure moves upward into secure services. In the documented Arduino-style path, SSLClient performs TLS on the ESP32 and passes encrypted data through an Ethernet Client transport to W5500 hardware sockets; the Ethernet controller does not perform TLS. Conversely, an ESP32–Rust experiment uses W5500 MACRAW mode to construct and transmit an ARP frame and observe it in Wireshark. They span secure traffic and packet-level investigation.

Why this matters

The range matters to developers: W5500 can sit beneath a secure client abstraction or expose raw frames for bring-up and diagnostics. These are distinct software architectures, not interchangeable security claims.

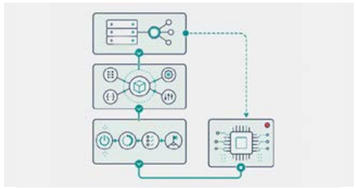
Ethernet Client Secure

ESP32 × Rust × W5500 MACRAW

4 WIZnet + Espressif Developer Paths

Drivers, Common Code, Initialization, and ioLibrary

OFFICIAL



espressif/w5500 - v2.0.0

ESP-IDF W5500 Ethernet driver integrating MAC and PHY functions over SPI through the Espressif networking path.

espressif/wiznet_common - v1.0.0

Shared SPI, MAC, and PHY operations for W5500 and W6100 drivers; not intended as a standalone component.

espressif/ethernet_init - v1.4.1

Configuration-led helper that initializes supported Ethernet drivers and can attach them to selected ESP-IDF network interfaces.

wiznet/esp_wiz_toe → wiznet/wsm_driver - v1.1.0

Current WIZnet port layer for W5500/W6300 and official ioLibrary integration, including direct hardware-socket APIs and examples.

Many applications, defined Ethernet roles, and reusable W5500 + ESP32 development paths.

THE EDGE KEEPS EXPANDING

Local intelligence. Purpose-built connectivity.
Every link has a role. Every edge has a purpose.
More applications. More possibilities.
One connected edge.

WIZnet

maker.wiznet.io