

KARL'S ULTIMATE DIY DYNO

Functional Specification Document (FSD)

Firmware Version: **3.72**

Build Date: **12 August 2026**

Hardware Platform: **ESP32-S3 Custom PCB**

Complete functional and technical specification for the KarlsDIYDyno V3.72 firmware, controller hardware interfaces, measurement system, data handling, user interface and installation.

Document	Functional Specification Document
Document	Functional Specification Document
Firmware	V3.72 – Dynamic Drum Edition
Build date	12.08.2026
Hardware	ESP32-S3 Custom PCB
Status	Release Document
Author	Karl Bø / KarlsDIYDyno

Document. Control

Revision	Date	Description	Author
3.72 Rev A	12.08.2026	Complete functional specification based on Firmware V3.72. Includes controller connection and installation diagram.	Karl Bø / AI

Document Basis

This document describes the implemented KarlsDIYDyNO V3.72 firmware and the ESP32-S3 custom controller as represented by the supplied V3.72 source code and the published V3.69 FSD used as the structural baseline. The firmware source is the implementation reference for configurable values, processing behaviour, communications and storage.

The supplied connection diagram is included as the user-facing wiring reference. Where a value is configurable in firmware, the default value and the configurable range are identified where supported by the source.

Important: This document describes the controller and firmware. Mechanical dyno construction, brake sizing, drum containment, engine restraint and other workshop safety engineering remain the responsibility of the builder.

Table of Contents

- 1. Project Overview
- 2. Hardware Architecture
- 3. Electrical Connections and Wiring
- 4. Sensor Systems
- 5. Firmware Architecture
- 6. Signal Acquisition and Filtering
- 7. Dynamometer Operating Modes
- 8. Measurement and Calculation Engine
- 9. Gear Ratio Management
- 10. Dynamic Drum Signal Processing
- 11. Run Recording and Data Storage
- 12. Web Interface and Communications
- 13. Configuration and Persistent Settings
- 14. Calibration
- 15. GPIO Trigger System
- 16. Wi-Fi and Network Operation
- 17. Firmware Update System
- 18. Sensor Status and Fault Handling
- 19. User Installation Guide

- 20. Operating Sequence
- 21. Engineering Limits and Safety Functions
- 22. Functional Requirements Summary
- Appendix A – Controller Pin Reference
- Appendix B – Configuration Parameter Reference
- Appendix C – WebSocket Command Reference
- Appendix D – Run Data Format
- Appendix E – Connection Diagram
- Appendix F – Installation Checklist

1. Project Overview

1.1 Purpose

KarlsDIYDyna is a compact, Wi-Fi-enabled engine dynamometer platform intended for motorcycles, scooters, mopeds, ATVs, go-karts, small racing engines and other rotating machinery where engine speed, torque and power measurement are required.

Firmware V3.72 runs on an ESP32-S3 controller and performs sensor acquisition, filtering, dynamometer calculations, run recording, configuration, storage and browser communication locally. A standard web browser provides the user interface; dedicated PC measurement software is not required.

1.2 Design Objectives

- Low-cost, serviceable measurement hardware.
- Browser-based operation through Wi-Fi.
- Interrupt-driven RPM acquisition.
- Local real-time signal processing on the ESP32-S3.
- Support for both brake and inertia dynamometer operation.
- Persistent storage of settings and recorded runs.
- Configurable sensor characteristics and dynamometer parameters.
- Practical installation using dedicated PCB interfaces.

1.3 Supported Dynamometer Modes

Mode	Primary measurement principle	Load cell	Typical use
Brake	Reaction torque from measured brake-arm force	Required	Steady-state tuning and controlled loading
Inertia	Angular acceleration of the dyno drum	Not required for power calculation	Acceleration runs and inertia testing

1.4 Firmware Identity

Parameter	Value
Firmware version	3.72
Firmware title	Dynamic Drum Edition
Build date	12.08.2026
MCU	ESP32-S3
Primary storage	LittleFS + ESP32 Preferences
Web server	ESPAsyncWebServer
Live transport	WebSocket
RTOS	FreeRTOS

2. Hardware Architecture

2.1 System Overview

The controller integrates the ESP32-S3 with dedicated interfaces for engine RPM, drum RPM, a configurable external trigger, a NAU7802 load-cell ADC, an MCP9600 thermocouple interface and a BMP280 environmental sensor.

Subsystem	Device/interface	Function
Controller	ESP32-S3	Processing, Wi-Fi, web server, interrupts and control
Engine RPM	Hall sensor / GPIO 4	Engine rotational speed
Drum RPM	Hall sensor / GPIO 5	Drum rotational speed
Load cell	NAU7802	Brake reaction force
EGT	MCP9600 at 0x67	K-type exhaust temperature
Atmosphere	BMP280 at 0x76 or 0x77	Ambient pressure and temperature
External trigger	Configurable GPIO, default 17	Run trigger state machine
Storage	ESP32 flash / LittleFS	Web files and recorded runs
Configuration	ESP32 Preferences	Persistent settings

2.2 Controller Pin Allocation

Function	GPIO / interface	Notes
Engine Hall	GPIO 4	INPUT_PULLUP, falling-edge interrupt
Drum Hall	GPIO 5	INPUT_PULLUP, falling-edge interrupt
I ² C SDA	GPIO 1	Sensor bus
I ² C SCL	GPIO 2	Sensor bus
External trigger	GPIO 17 default	Configurable; INPUT_PULLUP, falling-edge interrupt
Reserved	GPIO 26–33	Not available for configurable trigger
USB-C	USB interfaces	Power / data-programming connections

2.3 Electrical Logic

The ESP32-S3 uses 3.3 V GPIO logic. External Hall sensor installations must therefore be arranged so that the signal presented to the controller is suitable for the controller input. The supplied connection diagram recommends an optocoupler interface where electrical noise or signal-level compatibility is a concern.

Warning: Never connect an external signal above the ESP32-S3 GPIO voltage rating directly to a controller input. Verify the electrical interface before applying power.

2.4 I²C Sensor Bus

The firmware initializes I²C on GPIO 1 (SDA) and GPIO 2 (SCL) at 100 kHz. The BMP280 is detected at either 0x76 or 0x77. The MCP9600 EGT interface uses address 0x67.

3. Electrical Connections and Wiring

3.1 User Connection Overview

The following diagram is the primary user-facing connection reference for the ESP32-S3 controller. It shows the engine RPM Hall input, drum RPM Hall input, external trigger, four-wire load cell, K-type EGT thermocouple and USB connections.

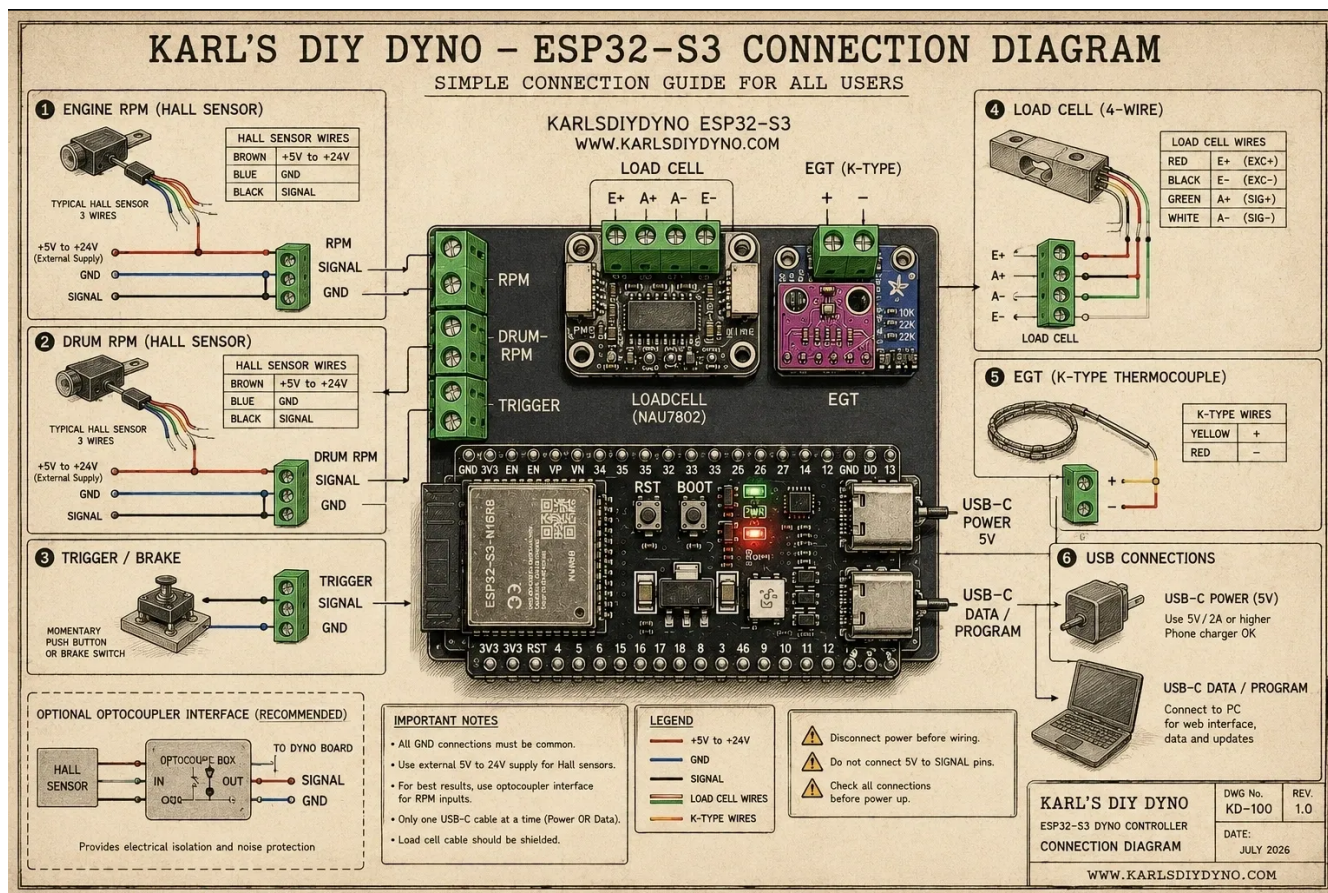


Figure 3-1. KarlsDIYDyNO ESP32-S3 controller connection diagram supplied with the project.

3.2 Engine RPM Hall Sensor

The engine RPM Hall sensor is connected to the dedicated RPM input. The firmware uses GPIO 4 with an interrupt on the falling edge. The engine Hall input has a configurable debounce period; the V3.72 default is 1400 μ s.

Connection	Function
Sensor supply	External Hall sensor supply, as required by sensor
Signal	Controller Engine RPM input / GPIO 4
Ground	Controller signal ground

3.3 Drum RPM Hall Sensor

The drum RPM Hall sensor is connected to the dedicated DRUM RPM input on GPIO 5. Drum pulses per revolution are configurable from 0.5 to 20.0 pulses/revolution. V3.72 uses this value when calculating drum rotational speed and when determining dynamic drum filtering parameters.

3.4 External Trigger

The external trigger uses a configurable GPIO. GPIO 17 is the default stored value. The firmware configures the selected input as INPUT_PULLUP and reacts to a falling edge. The trigger can be used as a physical run control.

3.5 Load Cell

The four-wire load cell connects to the NAU7802 interface using E+, E–, A+, and A–. The firmware converts the measured load to force using 9.81 m/s² and then calculates brake torque from the configured lever-arm length.

3.6 K-Type EGT

The MCP9600 interface is configured for a K-type thermocouple. The firmware reads both hot-junction temperature and ambient/reference temperature and applies a five-sample moving average through the EGT tracker.

3.7 USB

One USB-C connection is used for power and the other for data/programming as shown in the supplied diagram. USB data access is used for firmware development and service operations.

4. Sensor Systems

4.1 Engine RPM Measurement

Engine RPM is measured from the time interval between Hall sensor pulses. The firmware maintains a circular interval buffer containing 24 measurements. Intervals shorter than 80 μ s are ignored. At least four valid intervals are required before a raw RPM value is calculated.

The basic conversion is: $\text{RPM} = 60,000,000 / (\text{pulse interval in } \mu\text{s} \times \text{pulses per revolution})$. The configured engine pulses-per-revolution value is constrained by the user interface to 0.5–10.0.

4.2 Drum RPM Measurement

Drum RPM uses a separate 24-entry interval buffer. Intervals shorter than 60 μ s are ignored. The processing guard conditions differ between inertia and brake modes to allow more relaxed low-speed tracking in inertia mode.

Mode	No-pulse timeout	Minimum valid intervals
Inertia	3.0 s	2
Brake	1.0 s	4

4.3 Load Cell

The NAU7802 is configured for gain 128 and 320 samples/second. The firmware performs an automatic tare during startup when valid readings are available. The displayed load is constrained to 0–200 kg-equivalent before conversion to force.

4.4 EGT

The MCP9600 is configured for K-type thermocouple operation, ADC resolution 14-bit and filter coefficient 3. EGT is sampled approximately every 66 ms and passed through a five-sample moving average. Peak EGT, final EGT and ambient/reference temperature are retained for completed runs.

4.5 Atmospheric Measurement

The BMP280 is read approximately every 500 ms. The firmware calculates air density using the ideal-gas relationship for dry air, based on measured pressure and temperature, and constrains the resulting density value to 1.0–1.3 kg/m³.

5. Firmware Architecture

5.1 Processing Model

Firmware V3.72 separates time-critical signal acquisition from higher-level processing. Hall sensors are captured through hardware interrupts using the ESP32 timer. Processing is performed in FreeRTOS tasks.

5.2 Interrupts

- Engine Hall input: falling-edge interrupt on GPIO 4.
- Drum Hall input: falling-edge interrupt on GPIO 5.
- External trigger: falling-edge interrupt on the configured trigger GPIO.
- Interrupt handlers record pulse timing and place intervals into protected circular buffers.

5.3 FreeRTOS Tasks

Task	Function
SensorTask	Sensor acquisition, filtering, calculations, recording and WebSocket telemetry
CalibTask	Load-cell calibration procedure
Main loop	Wi-Fi reconnection, captive DNS processing and WebSocket client cleanup

5.4 Timing

Function	Nominal period
Sensor processing	50 ms
WebSocket live update	50 ms
EGT	66 ms
Gear-ratio update	100 ms
BMP280	500 ms

5.5 Persistent Storage

ESP32 Preferences stores configuration values such as calibration, drum geometry, operating mode, Wi-Fi credentials and trigger configuration. LittleFS stores the browser resources and recorded run files.

6. Signal Acquisition and Filtering

6.1 Engine Signal Processing

- 24 interval samples are examined.
- Intervals below 80 μ s are rejected.
- At least four valid intervals are required.
- A median filter is applied using the current median-buffer size.
- An exponential filter is applied to the median result.
- An additional ultra-smoothing stage uses 0.09 / 0.91 weighting.
- Above 5000 RPM, the engine RPM filter alpha is increased to 0.28 for faster response.

6.2 Drum Signal Processing

V3.72 introduces dynamic drum signal processing. The effective debounce time, median filter size and RPM filter alpha can be calculated from the configured drum pulse count and maximum safe drum RPM.

6.3 Dynamic Drum Debounce

The firmware first calculates the minimum pulse interval associated with the configured maximum safe drum RPM and pulse count. The dynamic debounce is then derived as approximately 35% of this minimum interval and constrained to a practical range of 150–2500 μ s, unless the user has supplied an explicit override.

6.4 Dynamic Median Size

Drum pulses/rev	Dynamic median size
12 or more	9
6 to <12	7
3 to <6	5
Below 3	3

The selected median size is maintained as an odd value between 3 and 9. A user override may be stored instead.

6.5 Dynamic RPM Filter Alpha

When no user override is active, the drum RPM filter alpha is calculated from pulse frequency using the implemented frequency-dependent expression and constrained to 0.05–0.50. This allows the filtering behaviour to adapt to the available pulse rate.

6.6 Manual Overrides

The settings interface can explicitly override drum debounce, median size and RPM filter alpha. The RESET_DYNAMIC_DRUM command clears those overrides and returns the processing to dynamic calculation.

7. Dynamometer Operating Modes

7.1 Brake Dynamometer Mode

Brake mode derives brake torque from measured load-cell force acting through the configured lever arm. Engine torque is obtained by applying the current gear ratio and drivetrain-efficiency compensation. Horsepower is then calculated from engine torque and engine RPM.

Stage	Calculation
Force	$\text{Load} \times 9.81 \text{ N/kg}$
Brake torque	$\text{Force} \times \text{lever arm}$
Engine torque	$\text{Brake torque} \div \text{gear ratio} \div \text{drivetrain efficiency}$
Horsepower	$\text{Engine torque} \times \text{RPM} \div 7121$

7.2 RPM Source in Brake Mode

The firmware supports spark/engine RPM, brake/drum-derived RPM and drum source selections. When RPM_SOURCE is set to brake, the displayed engine RPM is derived from drum RPM multiplied by the configured brake-to-engine ratio. Otherwise the measured engine RPM is used.

7.3 Inertia Dynamometer Mode

Inertia mode calculates drum torque from drum moment of inertia and angular acceleration. No load-cell force is required for the primary power calculation.

7.4 Drum Inertia

The firmware supports three inertia descriptions: solid, hollow and custom. The solid and hollow models use drum mass and diameter; the hollow model additionally uses wall thickness. A custom inertia value can be entered directly.

7.5 Acceleration

Angular acceleration is derived from changes in effective drum RPM over time. A five-sample RPM history is used, and acceleration is smoothed with an alpha of 0.12.

8. Measurement and Calculation Engine

8.1 Brake Torque

In brake mode, brake torque is limited to the implemented 0–2000 N·m calculation range. The load cell is converted to force, then multiplied by the lever-arm length.

8.2 Engine Torque

The current gear ratio is used to translate brake torque to engine torque. The resulting value is divided by the configured drivetrain efficiency. The calculation is constrained to the firmware's implemented 0–2000 N·m range.

8.3 Horsepower

The firmware uses the standard torque/RPM relationship implemented as $HP = \text{torque} \times \text{RPM} / 7121$. Brake-mode horsepower is only calculated when engine RPM is at or above the configured minimum brake threshold of 35 RPM.

8.4 Inertia Torque

During positive acceleration, drum torque is calculated as $\text{moment of inertia} \times \text{angular acceleration}$. Engine torque is then obtained by multiplying by the current gear ratio.

8.5 Drag Torque

During a qualifying coast-down condition, the firmware estimates drag torque from drum inertia and the magnitude of negative angular acceleration. Drag torque is constrained to 0–100 N·m and smoothed for display.

8.6 Air-Density Correction

In inertia mode, horsepower, engine torque and drum torque receive an air-density correction based on standard air density divided by measured air density. The correction factor is constrained to 0.85–1.15.

8.7 Output Smoothing

Horsepower and engine torque pass through moving-average smoothing buffers before presentation and recording. Additional display limits prevent unrealistically large transient values.

9. Gear Ratio Management

9.1 Automatic Gear Ratio

The GearRatioTracker maintains a history of measured engine-to-drum relationships and calculates a confidence value based on ratio stability. In brake mode, automatic tracking is accepted above 0.6 confidence. In inertia mode, the acceptance threshold is 0.4.

9.2 Manual Gear Ratio

The manual gear ratio can be configured from 0.5 to 100.0. Manual mode is used whenever automatic ratio tracking is disabled. When RPM_SOURCE is set to drum, the firmware always uses the manual ratio.

9.3 Brake-to-Engine Ratio

The brake-to-engine ratio is a separate configured ratio used as a fallback in brake mode and when drum-derived engine RPM is selected. Its configurable range is 0.1–50.0.

9.4 Ratio Reset

The RESET_RATIO command clears the gear-ratio tracker. This allows a new ratio measurement sequence to be established.

10. Dynamic Drum Signal Processing

10.1 Purpose

The Dynamic Drum Edition is intended to make drum RPM processing automatically adapt to drum sensor configuration and the selected safety RPM limit. The user can still force known values when a particular installation requires them.

10.2 Inputs to Dynamic Calculation

- Configured drum pulses per revolution.
- Configured maximum safe drum RPM.
- Optional user debounce override.
- Optional user median-size override.
- Optional user RPM-filter override.

10.3 Reset Behaviour

RESET_DYNAMIC_DRUM stores -1 for the three user override parameters and recalculates the dynamic drum values immediately.

10.4 Safety Limit

The drum RPM result is constrained to the configured MAX_SAFE_DRUM_RPM. The user interface accepts a maximum safe drum RPM from 500 to 25,000 RPM.

10.5 Why the System Uses Both Median and Exponential Filtering

The median stage suppresses isolated timing disturbances while the exponential stage provides continuous smoothing. A final ultra-smoothing stage further reduces visible jitter. The dynamic system changes these characteristics according to pulse availability rather than requiring the same filter values for every drum installation.

11. Run Recording and Data Storage

11.1 Recording Modes

- Manual start and stop.
- Automatic start and stop between configured RPM limits.
- External GPIO trigger sequence.
- Automatic safety timeout after 45 seconds of recording.

11.2 Recorded Point Structure

The firmware records a run point approximately every 100 ms. Each point contains the selected recording RPM, smoothed horsepower and smoothed torque. The active in-memory run is limited to 1500 points.

11.3 Run Completion

At completion, the firmware stores peak horsepower, peak horsepower RPM, peak torque, peak torque RPM, peak EGT, final EGT and ambient EGT/reference information.

11.4 File Format

Completed runs are stored as CSV data in LittleFS. The current V3.72 run-data interface reads three columns: RPM, HP and Torque.

Current V3.72 run data interface: RPM, HP, Torque.

11.5 Run Index

A /runs.idx index stores timestamp, filename, customer, vehicle, peak HP, peak torque, peak EGT and unit system for each saved run.

11.6 Run Capacity

The firmware maintains a maximum of ten saved runs. When the configured capacity is reached, the oldest entry is removed when a new run is stored.

12. Web Interface and Communications

12.1 Embedded Web Server

The firmware serves the browser interface directly from LittleFS using ESPAsyncWebServer. The root page is the dashboard, while dedicated settings, trend and update pages are available when their files exist.

12.2 WebSocket

Live measurement and control communication is implemented with an asynchronous WebSocket. The browser receives processed telemetry while sending commands for settings, recording, calibration and run management.

12.3 Live Telemetry

Field	Meaning
er	Engine/display RPM
dr	Drum RPM
hp	Smoothed horsepower
et	Engine torque
dt	Drum torque
l	Load in kg-equivalent
gr	Current gear ratio
rc	Gear-ratio confidence
ad	Air density
at	Ambient temperature
pr	Atmospheric pressure
eg	Filtered EGT when included
drag	Drag torque
rec	Recording state

12.4 Settings Broadcast

The firmware broadcasts version, build date, maximum display ranges, calibration factor, lever arm, pulse counts, RPM filtering, dyno mode, RPM source, gear-ratio mode, drum geometry, drivetrain efficiency, safe drum RPM, debounce values, median size and trigger GPIO.

12.5 Sensor Status

The sensor status endpoint reports availability of the NAU7802, MCP9600 and BMP280 interfaces.

13. Configuration and Persistent Settings

13.1 Configuration Philosophy

Settings are stored in non-volatile ESP32 Preferences so they survive power loss and firmware restarts. Configuration values are validated before being accepted.

Parameter	Range / choices	Default
Maximum RPM	5000–50000	16000
Maximum HP	5–1000	25
Maximum torque	5–1000	25
Brake arm	5–100 cm	25 cm
Engine pulses/rev	0.5–10	2
Brake-to-engine ratio	0.1–50	10
Auto-run hysteresis	50–500 RPM	180
Dyno mode	brake / inertia	brake
Drum mass	1–300 kg	10 kg
Drum diameter	10–300 cm	30 cm
Drum pulses/rev	0.5–20	10
Inertia type	solid / hollow / custom	solid
Wall thickness	0.5–20 cm	2 cm
Custom inertia	0.01–100	0.5
Drivetrain efficiency	70–99 %	93 %
Maximum safe drum RPM	500–25000	3000
Manual gear ratio	0.5–100	10
Engine debounce	100–5000 µs	1400 µs

13.2 RPM Source

RPM_SOURCE accepts spark, brake or drum. Spark uses the engine Hall signal as the primary engine-speed reference. Brake and drum selections allow drum-derived operation where the installation is configured accordingly.

13.3 Units

The browser can request metric or imperial display units. The selected unit system is also stored with saved run metadata.

14. Calibration

14.1 Load-Cell Tare

The TARE command waits briefly and then averages 16 available NAU7802 readings to establish the zero offset. A successful tare updates the active zero offset.

14.2 Load-Cell Calibration

Calibration is initiated with a known weight. The firmware waits two seconds, then collects readings for five seconds. The first ten qualifying samples are excluded from the averaging window, after which the calibration factor is calculated from the known weight and averaged raw signal.

14.3 Calibration Storage

The resulting calibration factor is stored in ESP32 Preferences and included in the settings broadcast.

14.4 Calibration Conditions

- Use a known, stable calibration load.
- Keep the load cell mechanically stationary during acquisition.
- Ensure the load path represents the same mechanical direction used during normal measurement.
- Repeat the calibration if the mechanical installation, load cell or amplifier configuration is changed.

15. GPIO Trigger System

15.1 Trigger State Machine

The external trigger is deliberately implemented as a three-stage state machine:

State	Trigger action	Firmware response
0	First press	Open run dialog
1	Second press	Start run
2	Third press	Stop and save run

15.2 Electrical Behaviour

The trigger input uses INPUT_PULLUP and a falling-edge interrupt. A 500 ms software debounce is applied.

15.3 Configurable GPIO

The trigger pin can be changed through the WebSocket command SET_GPIO_PIN. The firmware rejects pins that are reserved or conflict with the engine Hall, drum Hall or I²C pins.

15.4 Trigger Testing

TEST_GPIO_TRIGGER allows the web interface to invoke the same trigger-processing path without a physical switch.

16. Wi-Fi and Network Operation

16.1 Access Point Mode

When no valid stored station credentials are available, the firmware starts its own Wi-Fi access point. The source defines the default AP SSID and password as 'karlsdyno'.

16.2 Station Mode

When stored station credentials are present, the controller attempts to join the configured Wi-Fi network for up to 20 seconds during startup.

16.3 Fallback

If station connection fails, the firmware falls back to AP mode. This allows the dyno to remain accessible even when the workshop network is unavailable.

16.4 Captive DNS

In AP-only mode, the firmware starts a DNS server and redirects requests to the root page through a captive request handler.

16.5 Wi-Fi Reliability

Wi-Fi sleep is disabled. If station mode is active and the connection is lost, the main loop periodically requests reconnection.

17. Firmware Update System

17.1 Firmware OTA

The web server exposes an /update endpoint for firmware upload. The firmware image is written through the ESP32 Update facility and the controller restarts after a successful update.

17.2 File-System OTA

A separate /update_fs endpoint is provided for updating the LittleFS content used by the browser interface.

17.3 Purpose

- Firmware can be updated without a physical programming connection when the network interface is operational.
- Dashboard and browser resources can be updated independently of the firmware binary.
- Update errors are reported through the HTTP response and serial diagnostics.

18. Sensor Status and Fault Handling

18.1 Sensor Availability

At startup, the firmware attempts to initialize the BMP280, MCP9600 and NAU7802. Their availability is exposed to the browser through sensor-status JSON.

18.2 Missing Sensors

The software checks sensor availability before attempting sensor-specific operations. For example, the load-cell path remains inactive when the NAU7802 is not initialized.

18.3 RPM Timeout

Engine and drum RPM are forced to zero after their respective pulse timeout conditions. This prevents stale RPM values from remaining indefinitely when a rotating component stops or a sensor signal disappears.

18.4 Measurement Limits

Calculated torque, horsepower, RPM and drag values are constrained at several stages. These constraints are software safeguards against unreasonable calculated values; they do not replace mechanical safety systems.

Important: Software limits must never be treated as a substitute for a mechanically rated drum, brake, shaft, bearings, guards, emergency stop and engine restraint system.

19. User Installation Guide

19.1 Controller Installation

- Mount the controller securely in a protected location.
- Provide stable USB-C power appropriate for the controller.
- Keep sensor wiring away from ignition and high-current wiring.
- Provide strain relief on all external cables.
- Do not expose the PCB to conductive debris, fuel or uncontrolled moisture.

19.2 Engine Hall Sensor

Mount the Hall sensor rigidly and align the target so that the signal transition is repeatable. Keep the cable away from ignition leads where practical.

19.3 Drum Hall Sensor

Mount the target magnets or trigger features evenly around the drum. Configure the exact pulses-per-revolution value in the settings.

19.4 Load Cell Mechanical Installation

The load cell should experience the intended axial tension or compression only. Side loading, bending and twisting can reduce accuracy or damage the sensor.

19.5 EGT Probe

Install the K-type probe in a representative exhaust location appropriate for the engine. Use proper thermocouple wiring when an extension is required.

19.6 BMP280

The BMP280 is mounted on the controller PCB. The sensor requires access to the surrounding air for useful ambient measurements.

20. Operating Sequence

20.1 Startup

Step	Firmware action
1	Initialize serial diagnostics and verify ESP32 timer
2	Load stored dynamic-filter overrides
3	Attempt stored Wi-Fi station connection
4	Fall back to AP mode when required
5	Initialize I ² C and sensors
6	Mount LittleFS
7	Perform load-cell auto-tare
8	Load persistent configuration
9	Calculate dynamic drum parameters
10	Start web server, interrupts and FreeRTOS tasks

20.2 Manual Run

- Select or enter run/customer information.
- Verify sensor status.
- Tare/calibrate the load cell if required.
- Start recording with MANUAL_START.
- Monitor RPM, torque, power, EGT and other live values.
- Stop with MANUAL_STOP or the external trigger.
- The run is finalized, saved and added to the run list.

20.3 Automatic Run

AUTO_RUN defines a start and end RPM. The firmware applies configured hysteresis before accepting the start crossing. Recording ends when the selected recording RPM reaches the end RPM.

20.4 Safety Timeout

Any recording exceeding 45 seconds is automatically terminated and saved. This prevents an unattended recording state from continuing indefinitely.

21. Engineering Limits and Safety Functions

21.1 Implemented Software Limits

Function	Implemented limit
Maximum configurable engine RPM	50,000 RPM
Default engine RPM	16,000 RPM
Maximum safe drum RPM setting	25,000 RPM
Default safe drum RPM	3,000 RPM
Brake torque calculation ceiling	2,000 N·m
Engine torque calculation ceiling	2,000 N·m
Drag torque ceiling	100 N·m
Load-cell display range	0–200 kg-equivalent
Automatic recording timeout	45 s

21.2 Mechanical Safety

The firmware cannot detect structural failure, loose fixtures, bearing failure, tyre failure, shaft failure or fragmentation. The dyno builder must independently engineer containment, emergency stopping, guarding and mechanical ratings.

21.3 Electrical Safety

- Disconnect power before changing wiring.
- Never apply uncontrolled external voltage to an ESP32-S3 GPIO.
- Verify Hall sensor signal levels before connecting the controller.
- Use isolation/optocoupling where electrical noise or voltage compatibility warrants it.
- Keep thermocouple and load-cell wiring away from high-current and ignition wiring.

21.4 Software Reset Functions

RESET_DEFAULT clears Preferences and reboots into AP mode. RESET_ALL_SETTINGS clears Preferences, formats LittleFS and reboots. These functions should be used deliberately because RESET_ALL_SETTINGS removes stored run files and configuration.

22. Functional Requirements Summary

ID	Requirement	Status
FR-001	The controller shall measure engine RPM using GPIO 4.	Implemented
FR-002	The controller shall measure drum RPM using GPIO 5.	Implemented
FR-003	The controller shall support brake and inertia dynamometer modes.	Implemented
FR-004	The controller shall acquire load-cell data through the NAU7802.	Implemented
FR-005	The controller shall acquire K-type EGT through the MCP9600.	Implemented
FR-006	The controller shall acquire atmospheric pressure and temperature through the BMP280.	Implemented
FR-007	The controller shall provide browser-based operation over Wi-Fi.	Implemented
FR-008	The controller shall provide live telemetry through WebSocket.	Implemented
FR-009	The controller shall store run data in LittleFS.	Implemented
FR-010	The controller shall support manual and automatic recording.	Implemented
FR-011	The controller shall support an external configurable GPIO trigger.	Implemented
FR-012	The controller shall support persistent configuration storage.	Implemented
FR-013	The controller shall support load-cell tare and calibration.	Implemented
FR-014	The controller shall support firmware OTA updates.	Implemented
FR-015	The controller shall support filesystem OTA updates.	Implemented
FR-016	The controller shall dynamically calculate drum filtering parameters when overrides are disabled.	Implemented
FR-017	The controller shall limit drum RPM to the configured safety RPM.	Implemented
FR-018	The controller shall retain up to ten saved runs.	Implemented

Appendix A. Controller Pin Reference

Pin / Interface	Function	Direction / mode	Notes
GPIO 1	I ² C SDA	I ² C	BMP280 / MCP9600 / NAU7802 bus
GPIO 2	I ² C SCL	I ² C	100 kHz bus
GPIO 4	Engine Hall	Input pull-up / interrupt	Falling edge
GPIO 5	Drum Hall	Input pull-up / interrupt	Falling edge
GPIO 17	External trigger	Input pull-up / interrupt	Default; configurable
GPIO 26	Reserved	—	Not available for trigger
GPIO 27	Reserved	—	Not available for trigger
GPIO 28	Reserved	—	Not available for trigger
GPIO 29	Reserved	—	Not available for trigger
GPIO 30	Reserved	—	Not available for trigger
GPIO 31	Reserved	—	Not available for trigger
GPIO 32	Reserved	—	Not available for trigger
GPIO 33	Reserved	—	Not available for trigger

Trigger configuration rule: The firmware rejects a trigger GPIO that conflicts with the Hall inputs, I²C pins or reserved GPIO 26–33.

Appendix B. Configuration Parameter Reference

Parameter	Default	Valid range / values	Stored
maxRPM	16000	5000–50000	Yes
maxHP	25	5–1000	Yes
maxTorque	25	5–1000	Yes
armCm	25 cm	5–100 cm	Yes
pulsesPerRev	2	0.5–10	Yes
brakeToEngineRatio	10	0.1–50	Yes
autoHyst	180	50–500	Yes
dynoMode	brake	brake / inertia	Yes
drumMass	10 kg	1–300 kg	Yes
drumCm	30 cm	10–300 cm	Yes
pulsesPerRevDrum	10	0.5–20	Yes
drumInertiaType	solid	solid / hollow / custom	Yes
drumWallCm	2 cm	0.5–20 cm	Yes
drumInertiaCustom	0.5	0.01–100	Yes
drivetrainEff	93 %	70–99 %	Yes
maxSafeDrumRpm	3000	500–25000	Yes
rpmSource	spark	spark / brake / drum	Yes
gearRatioMode	auto	auto / manual	Yes
gearRatioManual	10	0.5–100	Yes
engDebounce	1400 µs	100–5000 µs	Yes
drumDebounce	dynamic	50–5000 µs	Override only
medianSize	dynamic	3,5,7,9	Override only
rpmFilter	dynamic	0.01–0.9	Override only
gpioTriggerPin	17	0–39 excluding reserved/conflicts	Yes

Appendix C. WebSocket Command Reference

Command	Function
TARE	Zero the load-cell offset.
RESET_RATIO	Reset automatic gear-ratio tracking.
RESET_DYNAMIC_DRUM	Return drum filtering to dynamic calculation.
SET_CUSTOMER:	Set run metadata.
SET_RUN_TIMESTAMP:	Set run timestamp.
CALIBRATE:	Start load-cell calibration using known mass.
SET_SINGLE:=	Set and persist a validated configuration parameter.
WIFI_CLIENT::	Store station Wi-Fi credentials and reboot.
RESET_DEFAULT	Clear Preferences and reboot to AP mode.
RESET_ALL_SETTINGS	Clear Preferences, format LittleFS and reboot.
AUTO_RUN::	Enable automatic recording between RPM limits.
GET_SETTINGS	Broadcast current settings.
GET_SENSOR_STATUS	Return sensor availability.
GET_RUNS	Return saved-run list.
GET_RUN_DATA:	Return saved run data.
DELETE_RUN:<index>	Delete a saved run.
SET_UNITS_IMPERIAL	Select imperial display mode.
SET_UNITS_METRIC	Select metric display mode.
PAGE_SETTINGS	Tell firmware that settings page is active.
PAGE_MAIN	Return to main page processing.
CANCEL_AUTO_RUN	Cancel automatic/manual recording state.
MANUAL_START	Start manual recording.
MANUAL_STOP	Stop and save recording.
SET_GPIO_PIN:<pin>	Change external trigger GPIO.
GET_GPIO_STATUS	Return trigger GPIO status.
TEST_GPIO_TRIGGER	Exercise trigger state machine in software.

Appendix D. Run Data Format

D.1 Stored CSV

The V3.72 run-data reader expects the saved run CSV to contain three engineering columns.

Column	Meaning	Units
RPM	Recording/display RPM	RPM
HP	Smoothed horsepower	HP
Torque	Smoothed engine torque	N·m

D.2 Run Index

Field	Meaning
Timestamp	Run timestamp
Filename	LittleFS run file
Customer	Customer/run owner
Vehicle	Unit/vehicle description
Peak HP	Peak recorded horsepower
Peak Torque	Peak recorded torque
Peak EGT	Peak EGT
Unit system	Metric or imperial

D.3 Downsampling

When a run contains more than the storage target, the firmware downsampling routine reduces the saved data to a maximum of approximately 900 CSV rows. The in-memory acquisition buffer can contain up to 1500 points.

Appendix E. Connection Diagram

E.1 Full-Page Wiring Reference

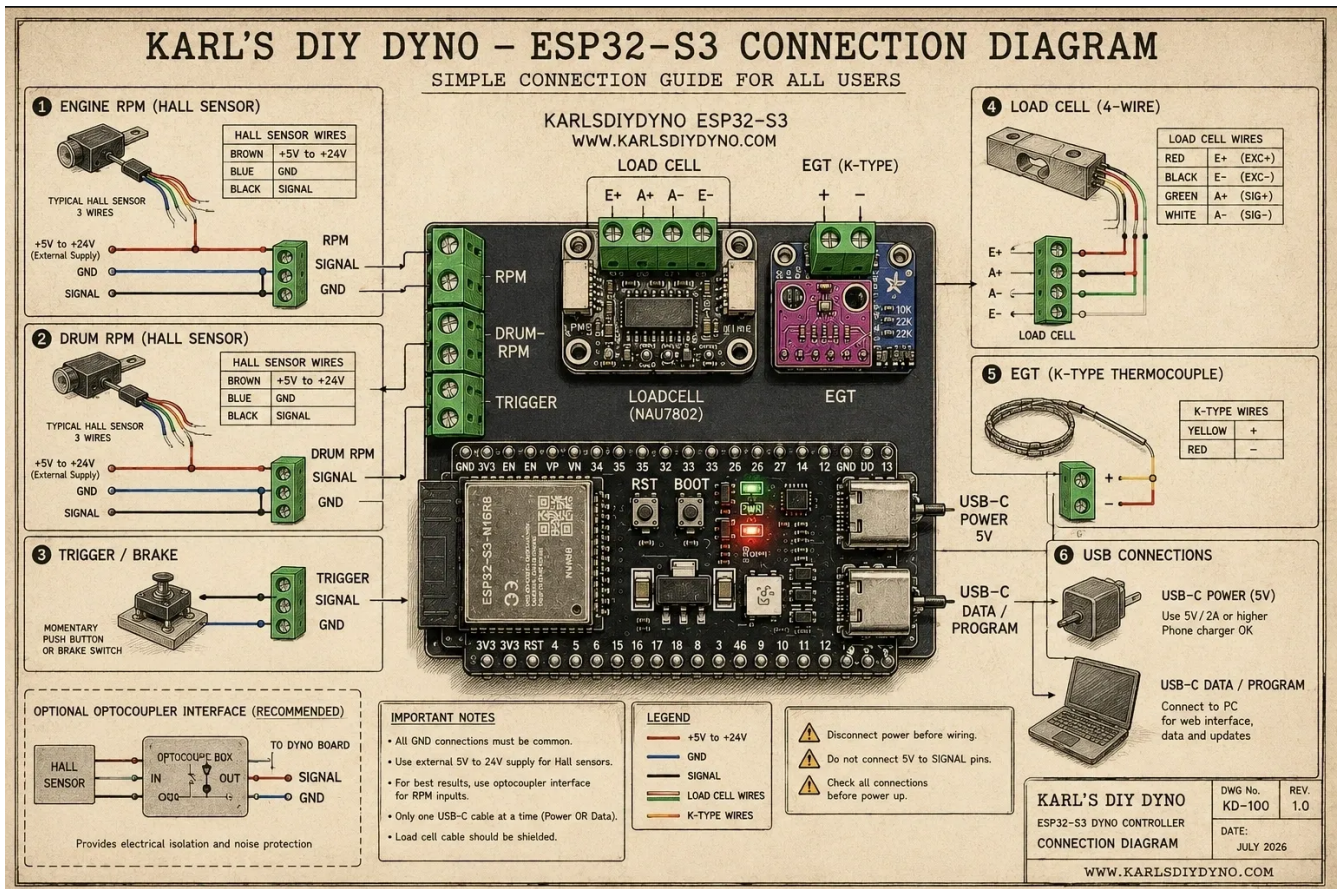


Figure E-1. User connection diagram for the KarlsDIYDyno ESP32-S3 controller.

Appendix F. Installation Checklist

- ESP32-S3 module firmly seated on the controller PCB.
- Engine Hall sensor connected to the Engine RPM input.
- Drum Hall sensor connected to the Drum RPM input.
- Drum pulses-per-revolution correctly configured.
- External trigger connected only to a valid configured GPIO.
- Load cell connected E+, E–, A+, A– with polarity verified.
- K-type thermocouple connected with correct polarity.
- BMP280 unobstructed and exposed to representative ambient air.
- Sensor grounds correctly referenced.
- Hall signals verified to be ESP32-S3 compatible.
- Optocoupler/signal conditioning fitted where required.
- All sensor cables secured against vibration.
- Signal cables separated from ignition and high-current wiring.
- Load cell mechanically aligned without side loading.
- Drum mechanical containment and emergency stopping verified independently.
- Maximum safe drum RPM configured before testing.
- Load-cell tare completed.
- Load-cell calibration completed with a known mass.
- Sensor status page checked before first run.
- First run performed at conservative speed while verifying all measurements.

Final note: The KarlsDIYDyno controller is a measurement and control system. The complete dyno installation must be engineered so that mechanical, electrical and thermal hazards are controlled independently of the software.

Document End

KARL'S ULTIMATE DIY DYNO – Functional Specification Document

Firmware Version 3.72 • Dynamic Drum Edition • Build 12 August 2026

This document is intended as the technical and installation reference for the V3.72 ESP32-S3 controller and firmware.

www.karlsdiydyno.com