Gearotons

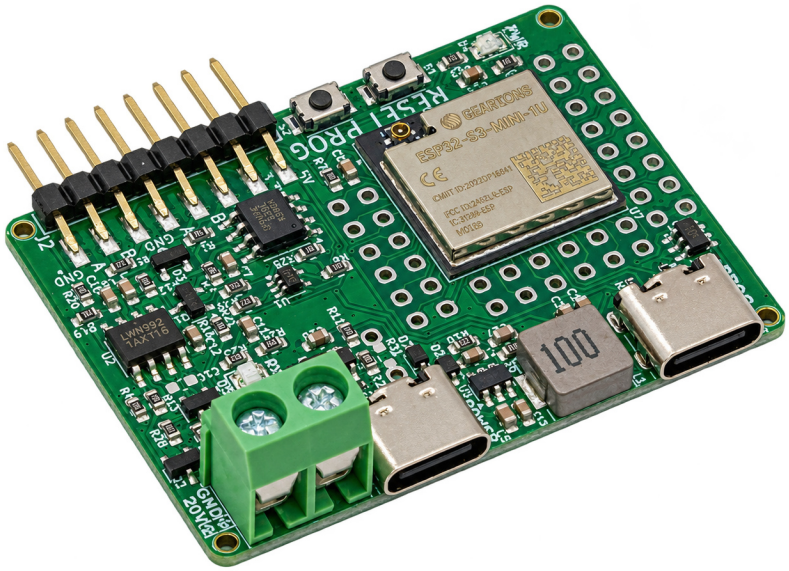
PROGRAMMABLE MOTION CONTROL

RS485 Controller

ESP32-S3 development board for Gearotons servomotors and other RS485 devices

Affordable and Simple All-in-One Motion Control

From Education to Innovation



External antenna included. Cables, power supply and motors are separate system components.

A compact controller for shared-bus motion

Run motor-control logic directly on the ESP32-S3-MINI-1U module. The board combines native USB programming, a half-duplex RS485 interface, a USB-C PD power input and accessible GPIO pads.

Embedded control: dual-core LX7 processor, up to 240 MHz; 2.4 GHz Wi-Fi and Bluetooth 5 LE capabilities depend on application firmware.

Simple wiring: two parallel A/B/GND groups, automatic RS485 direction circuitry, reset and programming buttons.

Motor-power integration: standard assembly requests 20 V USB-C PD, with a direct motor-power output at the screw terminal.

Visible bus activity: bright LEDs indicate transmission (red TX) and reception (green RX).

SPECIFICATIONS

Electrical & power

Power inputs, onboard supply rails and RS485 interface specifications.

Parameter	Specification / conditions
Processor / module	ESP32-S3-MINI-1U; dual-core Xtensa LX7, up to 240 MHz.
Wireless hardware	2.4 GHz Wi-Fi 802.11b/g/n and Bluetooth 5 LE; external antenna included. Wireless control requires suitable firmware.
Programming supply, J5	5 V nominal USB input and native ESP32-S3 USB data. J5 alone is sufficient to power and program the module; motor power is separate.
PD power input, J3	Requests 20 V using CH221K and R27 = 200 kΩ. Actual voltage depends on the source and successful PD negotiation.
Motor-power output, J1	Pin 1 = J3 VBUS rail (labelled +20 V); pin 2 = GND. Direct pass-through, not a separately regulated 20 V output.
Logic / RS485 interface	3.3 V logic; SP3485EN half-duplex differential transceiver. Controller, USB and motor grounds are common; no galvanic isolation.
RS485 example setting	230400 baud, 8 data bits, no parity, 1 stop bit. TX = GPIO4; RX = GPIO5. Board maximum baud rate is not specified.
Bus loading / protection	R5 = 220 Ω across A/B. R1 = 560 Ω pulls A high; R3 = 560 Ω pulls B low. SM712 transient suppressor on the bus.
Auxiliary power pins, J2	3.3 V rail and J5-associated 5 V rail. Available external current is not specified. The 5 V rail is not generated from the PD input.

USB-C PD request and actual motor voltage

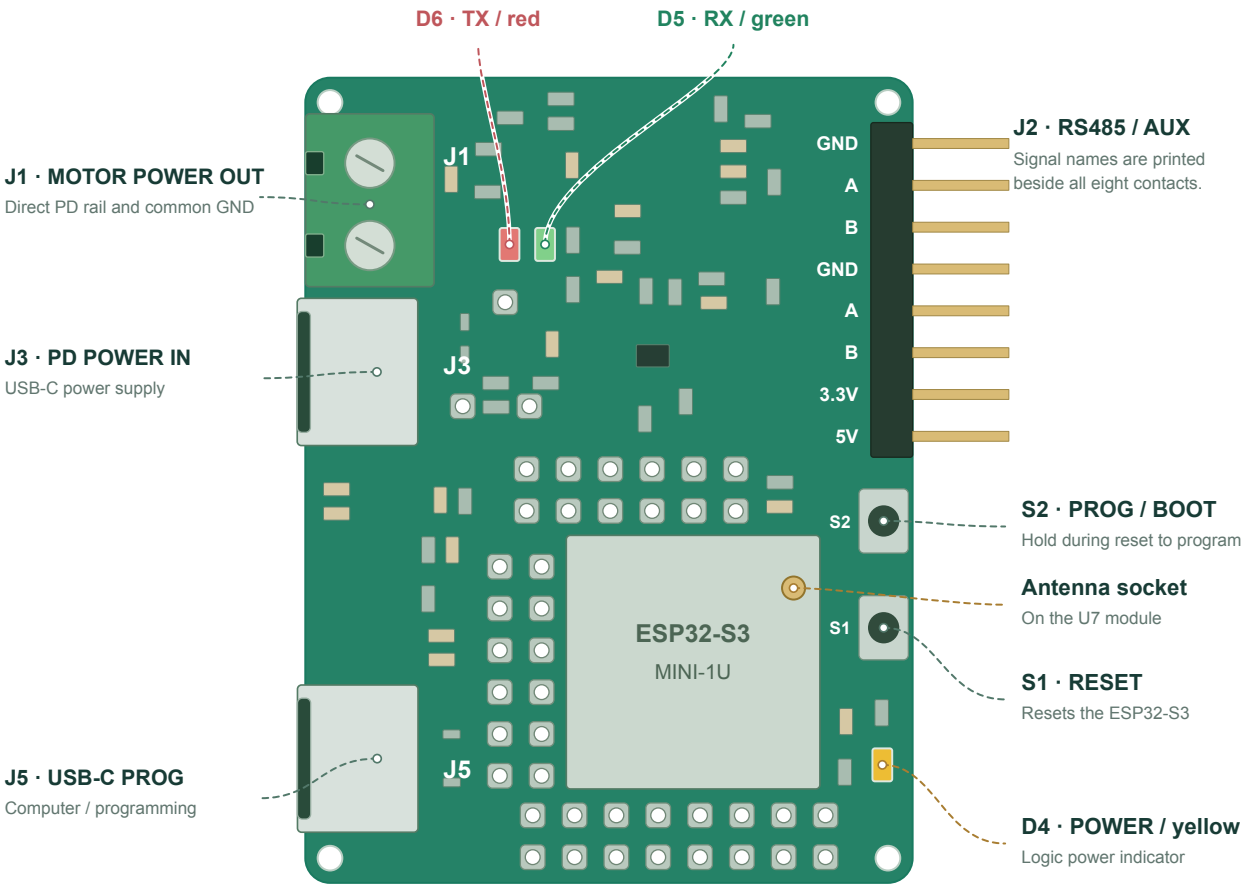
The CH221K requests 20 V from the USB-C power supply. If the source cannot supply 20 V and instead delivers 15 V or 12 V, Gearotons M17 motors can still operate. Below 12 V, the motors will likely not work correctly. Measure J1 before connecting motors and ensure the supply and cable can support the motor load. J5 alone does not provide motor power at J1.

Unspecified product ratings: board current consumption, continuous motor-power pass-through current, maximum cable length, environmental limits and product-level ESD/EMC ratings. Component ratings do not define board-level limits.

INTERFACES · PHYSICAL LOCATOR

Power, programming & status

Locate connectors, switches and indicator LEDs before connecting or programming the controller.



U7 · ESP32-S3-MINI-1U

COMPONENT SIDE (B.Cu) · USB sockets at left · J2 header at right

Component-side view (B.Cu). J1 supplies motor power; J2 provides the RS485 interface and auxiliary rails. Attach the included antenna to the U7 coax socket before using the radio. Connector pinouts and controls are listed on the next page.

Separate the two USB-C functions

J3: USB-C PD motor-power input. **J5:** native USB programming and 5 V logic power. The J3 USB data pins are unconnected. USB-C connector shape does not indicate which function it provides.

INTERFACES · ELECTRICAL REFERENCE

Connector pinout & controls

The two RS485 groups on J2 are electrically parallel; all grounds share one reference.

J2 pin	Signal	Use
1	GND	Common ground
2	A	RS485 A
3	B	RS485 B
4	GND	Same net as pin 1
5	A	Same bus as pin 2
6	B	Same bus as pin 3
7	3.3 V	Logic supply rail; external load not rated
8	5 V	Programming USB supply rail; external load not rated

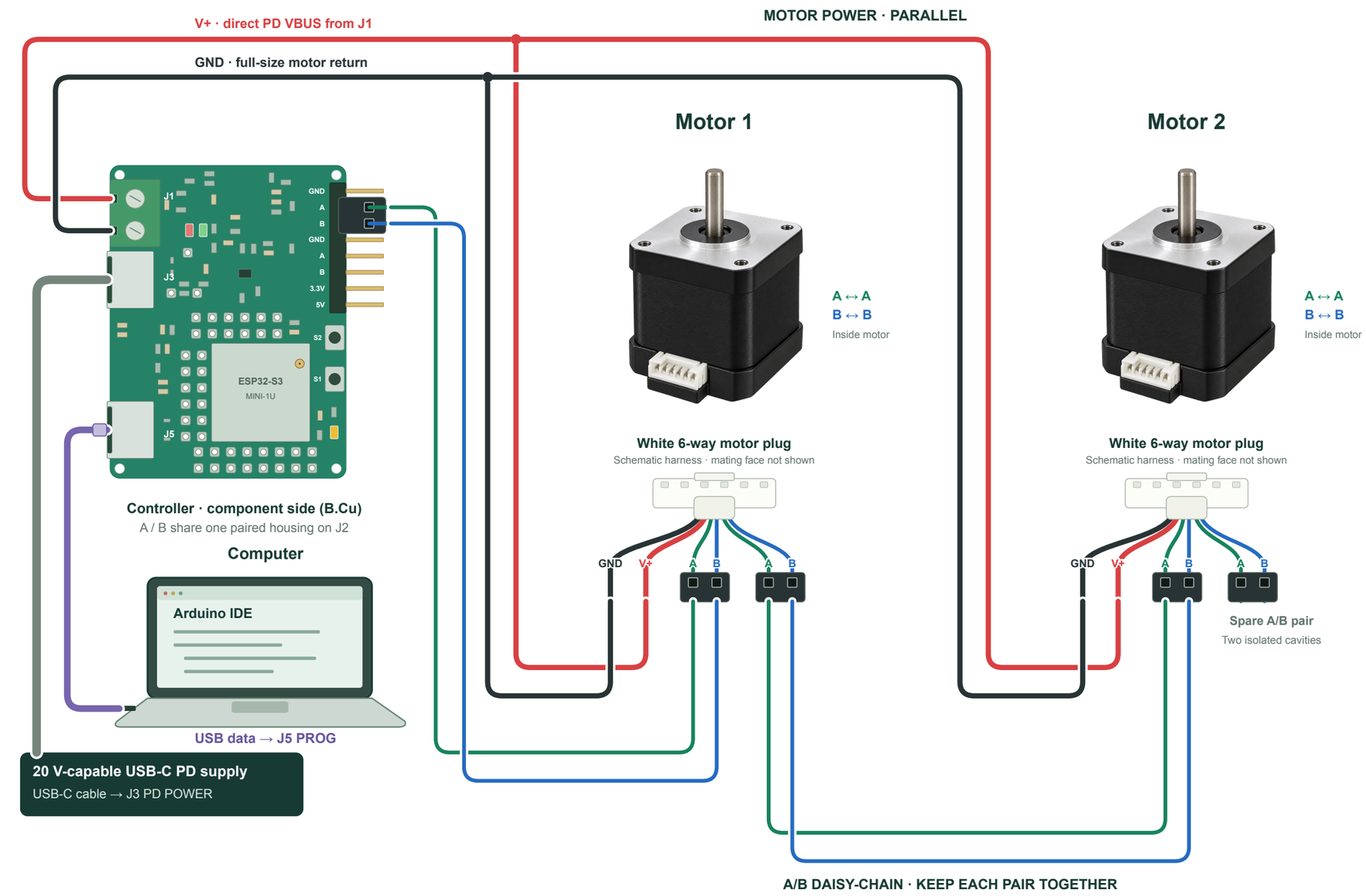
Power, programming, switches and indicators

Reference	Function
J1 · POWER OUT	Two-way screw terminal. Pin 1 is the positive PD rail; pin 2 is GND. Verify polarity on the board.
J3 · PD POWER IN	USB-C power input and PD negotiation. This socket does not program the ESP32-S3.
J5 · USB-C PROG	Native USB D–/D+ connect to GPIO19/GPIO20. Use a data-capable USB cable.
S1 · RESET	Pulls the module EN signal low to reset the controller.
S2 · PROG	Pulls GPIO0 low; hold during reset to enter ROM download mode.
D4 · yellow LED	Logic-power indication.
D5 · green LED	Receive activity; indication depends on bus traffic.
D6 · red LED	Transmit activity; indication depends on bus traffic.

Keep motor supply voltage off J2. Its 3.3 V and 5 V pins are auxiliary logic rails, not motor-power terminals. Refer to the physical locator and actual PCB labels for orientation.

Two-motor connection diagram

System connections: computer USB to J5 PROG and USB-C PD power supply to J3.



Power in parallel. A/B continues through Motor 1 to Motor 2.

J1 and J2 GND are common on the PCB; both motors have full-size power-return wires.
20 V is nominal after PD negotiation. Wire with power off; verify voltage and polarity.
A/B lines show electrical pairing; select compatible mating parts. Each black shell has two isolated cavities.

Computer: J5 USB-C PROG. Motor-power supply: J3 PD POWER IN. Connector mating faces are not shown; match actual labels.

SYSTEM WIRING

Connect two servomotors

The illustrated USB-C PD arrangement requests 20 V; the motor rail follows the actual supply voltage.

Use the following net-by-net connections for both motors. Connect the data bus along one route and distribute motor supply/return wiring for the combined motor current.

Net	Controller connection	Motor 1 and Motor 2
V+	J1 pin 1 (actual J3 voltage)	Each motor V+ / positive supply
GND	J1 pin 2 and J2 pin 1 or 4	Each motor GND / supply return
A	J2 pin 2 or 5	A to A, onward along the bus
B	J2 pin 3 or 6	B to B, onward along the bus

1. Check power while disconnected. With motors disconnected, connect the USB-C PD source to J3 and measure J1. Check the actual voltage and polarity, then turn the source off before wiring the motors.

2. Wire the common ground and motor supply. Connect both motor supply returns and controller GND. Motor returns must be sized for motor current; the data-ground lead must not be the only motor-current return.

3. Connect the differential bus. Connect A to A and B to B. Route A/B as a twisted pair along one bus, keeping branches short. Give each motor a distinct alias before addressed motion commands.

4. Power and test deliberately. Secure both motors before discovery or movement examples. The power supply, cable and distribution wiring must support the combined load.

The M17 motor reference specifies 12–24 V operation. A 15 V or 12 V supply can run the motors even when 20 V is unavailable; below 12 V they will likely not work correctly. R5 already loads A/B with 220 Ω. Select additional end termination for the complete bus instead of adding resistors at every motor. No fixed motor-count guarantee is specified.

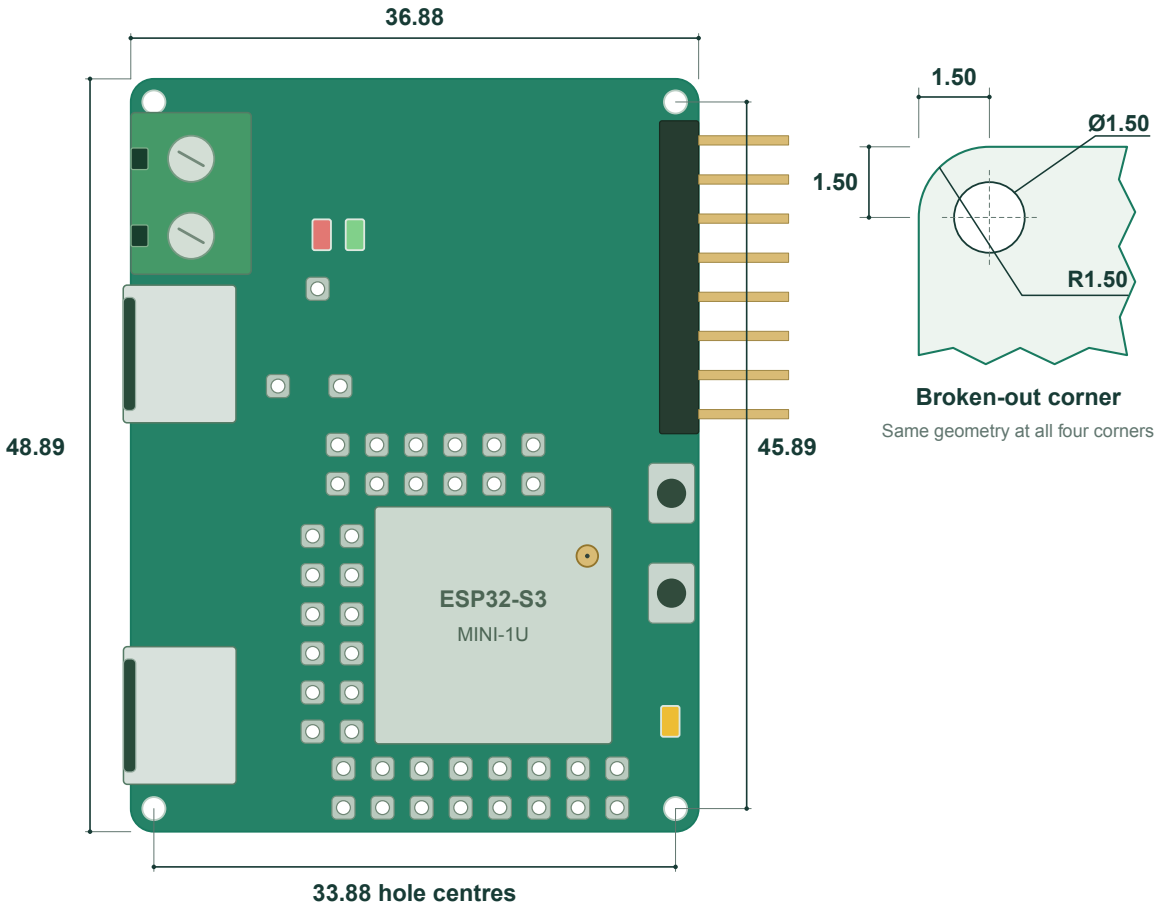
Using a separate motor power supply

Keep J5 USB-C PROG connected to power the controller. Connect a suitable external supply positive directly to each motor V+, and its negative to each motor GND. Also connect the external supply negative to controller GND (J2 pin 1 or 4, or J1 pin 2). This common-ground connection is required for RS485 communication. Leave J3 and J1 pin 1 unused in this arrangement.

MECHANICAL INTEGRATION

Detailed PCB dimensions

PCB revision 0.5. Nominal dimensions in millimetres; connector overhang and antenna excluded.



mm · 4 × Ø1.50 mounting holes · centres 1.50 from adjacent edges · corners R1.50

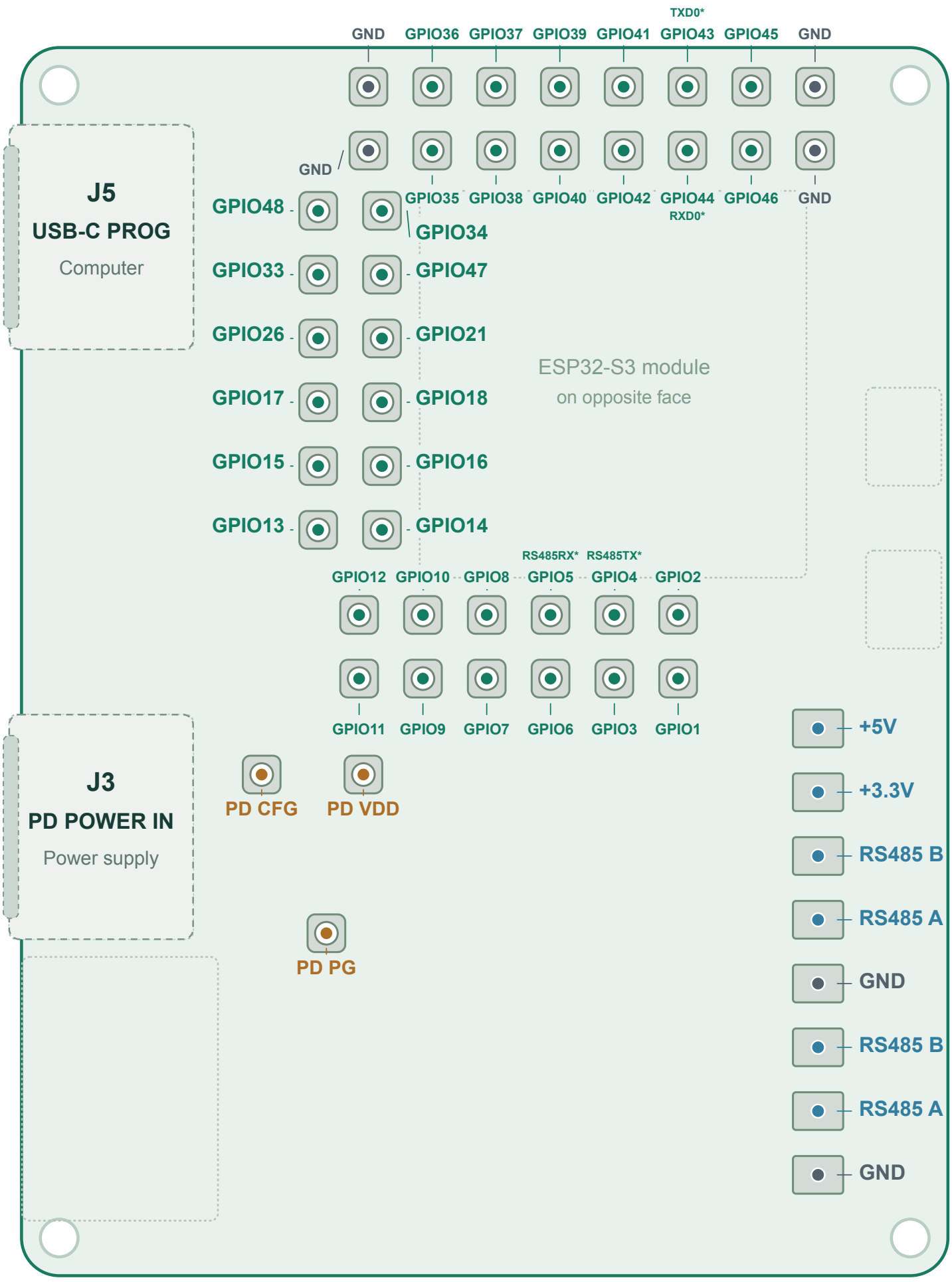
Nominal PCB outline, corner radius, mounting-hole diameter, hole-centre spacing and edge insets are shown above. Manufacturing tolerances are not specified; allow appropriate enclosure clearance.

Mounting and assembled envelope

PCB thickness is 1.6 mm nominal. Keep clearance around underside components and connector solder joints. Overall assembled height and antenna/cable envelope are not specified. The Ø1.50 mm holes are not M2 or M3 clearance holes.

Breakout headers — solder required

Rear / label-side view (F.Cu). Breakout headers are not fitted; solder headers or wires to the required through-hole pads.



REAR / LABEL SIDE (F.Cu) · looking toward the PCB

USB-C sockets and module are on the opposite face; dashed outlines locate them.

PD CFG / PD VDD / PD PG: HASL-finished plated through-hole pads for USB-C power delivery.

Signal legend: Green: GPIO. Gray: GND. Amber: PD-control points. Blue: RS485 and auxiliary contacts. Label colours indicate function; pads have a HASL finish. Dashed outlines locate components fitted on the opposite face.

*** PCB label aliases:** GPIO43 = TXD0 (UART0 transmit); GPIO44 = RXD0 (UART0 receive). The final character in TXD0/RXD0 is zero. These are the same physical pins, with functions selected by firmware. GPIO4 = RS485TX; GPIO5 = RS485RX. These logic signals are wired to the onboard RS485 circuitry; they are not the differential A/B bus and are not independent spare GPIO.

Shared functions: GPIO0/3/45/46 are module strapping signals; GPIO0 is connected to PROG. Use 3.3 V logic and avoid disturbing boot, USB or RS485.

PD CFG, PD VDD and PD PG: HASL-finished plated through-hole pads connected to the CH221K power-delivery chip, not GPIO. See the CH221K datasheet for pin functions and voltage configuration. Breakout headers are not fitted; J2 is separate.

Module reference: Official Espressif ESP32-S3-MINI-1 / MINI-1U datasheet (PDF).

BRING-UP

Getting started

Prepare the hardware and software, establish communication, then enable motion deliberately.

1. **Prepare the hardware.** Attach the included external antenna. With power off, check the motor polarity and A/B/GND connections. Secure the mechanism before running any movement example.

2. **Use the programming port.** For programming only, connect J5 USB-C PROG to a computer with a data-capable cable. This powers the ESP32-S3; no motor supply or J3 connection is needed.

3. **Install Arduino support.** Install the Arduino ESP32 board package and “Servomotor by Gearotons” from Library Manager. Select an ESP32-S3 target; configure native USB CDC for the USB serial console as appropriate for the board package.

4. **Enter download mode if needed.** Hold S2 PROG, press and release S1 RESET, then release PROG. Select the detected USB device and upload the application.

5. **Verify motor power and identity.** For motion, use J3/J1 or a suitable separate motor supply. With a separate supply, connect its negative to controller GND as well as motor GND. Check voltage and polarity before connecting motors; use the intended alias and secure the mechanism for one full rotation.

Arduino example for the controller

Install Servomotor by Gearotons in the Arduino IDE, then upload the basic motor-control program. The example uses GPIO4 for RS485 TX and GPIO5 for RS485 RX.

The example commands motor motion

The program enables the motor drive, requests one shaft rotation in one second, then disables the drive. Match the motor alias and serial connection to the intended motor, and secure the mechanism before running it.

GETTING STARTED · ARDUINO IDE

Install the Arduino library

Install Servomotor by Gearotons directly from the Arduino IDE.

Install the library through the Arduino Library Manager:

1. Open Arduino IDE
2. Go to Sketch → Include Library → Manage Libraries
3. Search for "Servomotor"
4. Find the library called "Servomotor by Gearotons" and click Install



The screenshot uses Tools → Manage Libraries. Install Servomotor by Gearotons and select an ESP32-S3 board target for this controller; the ESP32-C3 selection shown is an example.

GETTING STARTED · ARDUINO EXAMPLE

Arduino: basic motor control

Basic motor control using the Servomotor by Gearotons Arduino library.

Install “Servomotor by Gearotons” using Arduino Library Manager. This example uses GPIO4 TX and GPIO5 RX for the onboard RS485 interface.

```
// Minimal Arduino example: Trapezoid move using built-in unit conversions
// Goal: spin the motor exactly 1 rotation in 1 second, then stop.
// Sequence:
//  enable MOSFETs -> trapezoidMove(1.0 rotations, 1.0 seconds) -> wait 1.1s -> disable MOSFETs.
//
// Notes:
// - This uses the library's unit conversion (no raw counts/timesteps).
// - Configure Serial1 pins for your board (ESP32 example pins below).
// - Motor is created AFTER Serial1.begin(...) so hardware UART pins are set first.

#include <Servomotor.h>

#define ALIAS 'X'                   // Device alias
#define BAUD 230400                 // RS485 UART baud rate
#define DISPLACEMENT_ROTATIONS 1.0f // 1 rotation
#define DURATION_SECONDS 1.0f       // 1 second
#define TOLERANCE_PERCENT 10        // +10% wait margin because the motor's clock is not
                                    //  perfectly accurate
#define WAIT_MS ((unsigned long)(DURATION_SECONDS * 1000.0f * (100 + TOLERANCE_PERCENT) / 100))

// Example RS485 pins for ESP32 DevKit (change as needed for your board)
#if defined(ESP32)
#define RS485_TXD 4             // TX pin to RS485 transceiver
#define RS485_RXD 5             // RX pin from RS485 transceiver
#endif

void setup() {
  Serial.begin(115200); // Console serial for debugging

  // Create the motor; serial port opens on first instantiation.
#if defined(ESP32)
  Servomotor motor(ALIAS, Serial1, RS485_RXD, RS485_TXD);
#else
  Servomotor motor(ALIAS, Serial1);
#endif

  // Use units: rotations for position, seconds for time
  motor.setPositionUnit(PositionUnit::SHAFT_ROTATIONS);
  motor.setTimeUnit(TimeUnit::SECONDS);

  motor.enableMosfets();
  motor.trapezoidMove(DISPLACEMENT_ROTATIONS, DURATION_SECONDS);
  delay(WAIT_MS);
  motor.disableMosfets();
}

void loop() {
}
```

Set ALIAS to the intended motor alias before uploading. The program commands one shaft rotation in one second, then disables the motor drive.

DESIGN REFERENCE

Controller schematic & resources

Circuit schematic, manufacturer documentation and software resources for PCB revision 0.5.

Full controller schematic

The following page shows the complete PCB revision 0.5 controller schematic, including power regulation, USB-C PD negotiation, native USB, RS485, controls and indicator circuitry.

Official module documentation

Espressif ESP32-S3-MINI-1 / ESP32-S3-MINI-1U datasheet
Manufacturer reference for the module pin definitions, boot configuration, radio, electrical specifications and module mechanical details.

Power-delivery chip documentation

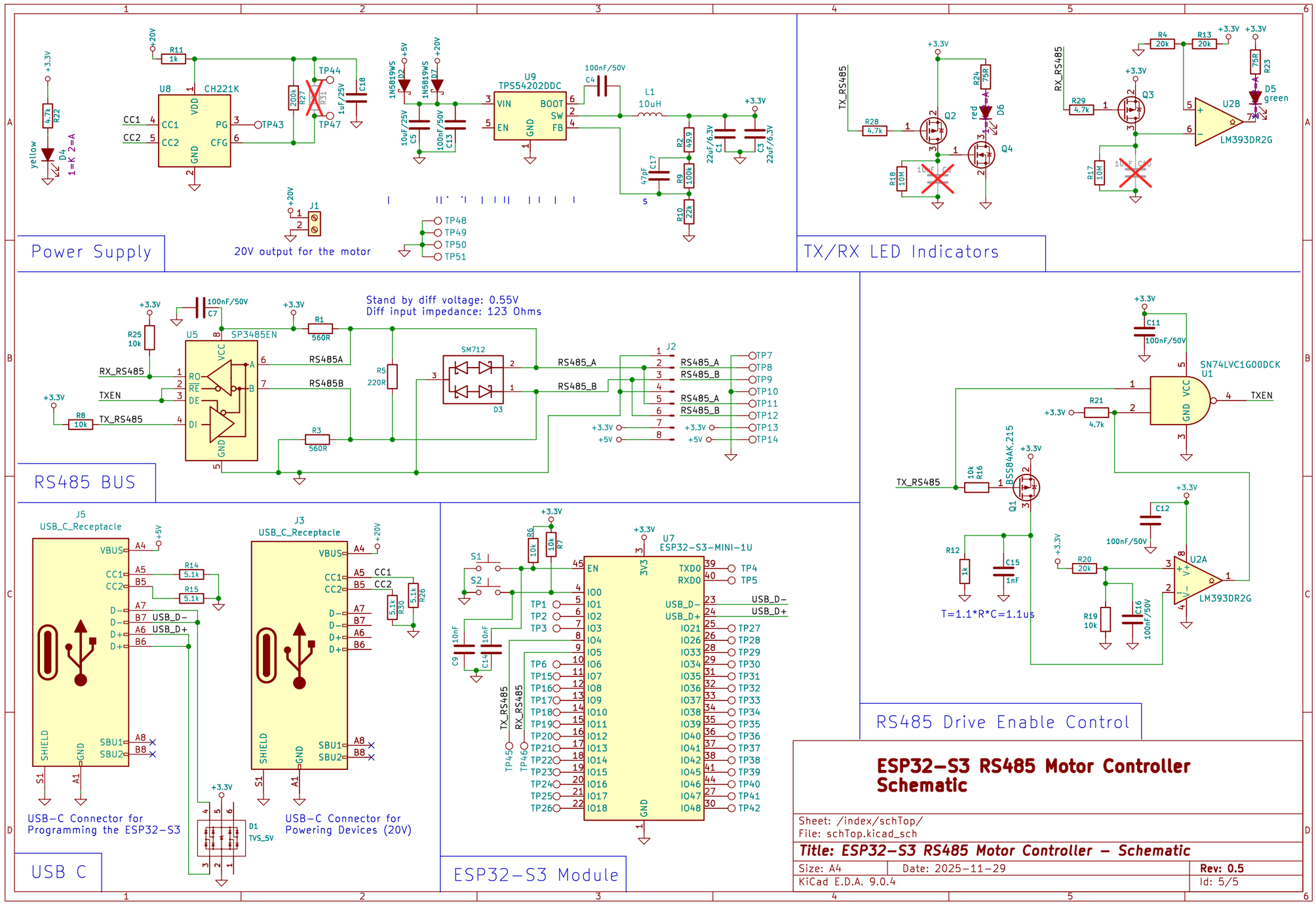
WCH CH224-family datasheet — CH221K sections
Refer to the CH221K pin descriptions and voltage-configuration table for PD CFG, PD VDD and PD PG. The standard 200 kΩ configuration requests 20 V from the USB-C power supply.

Software, tutorials and design files

Controller source: github.com/tomrodinger/RS485_Controller
Motor setup and protocol: tutorial.gearotons.com
Arduino library: github.com/tomrodinger/Servomotor_Arduino_Library
Feedback: tutorial.gearotons.com/feedback

Full controller schematic

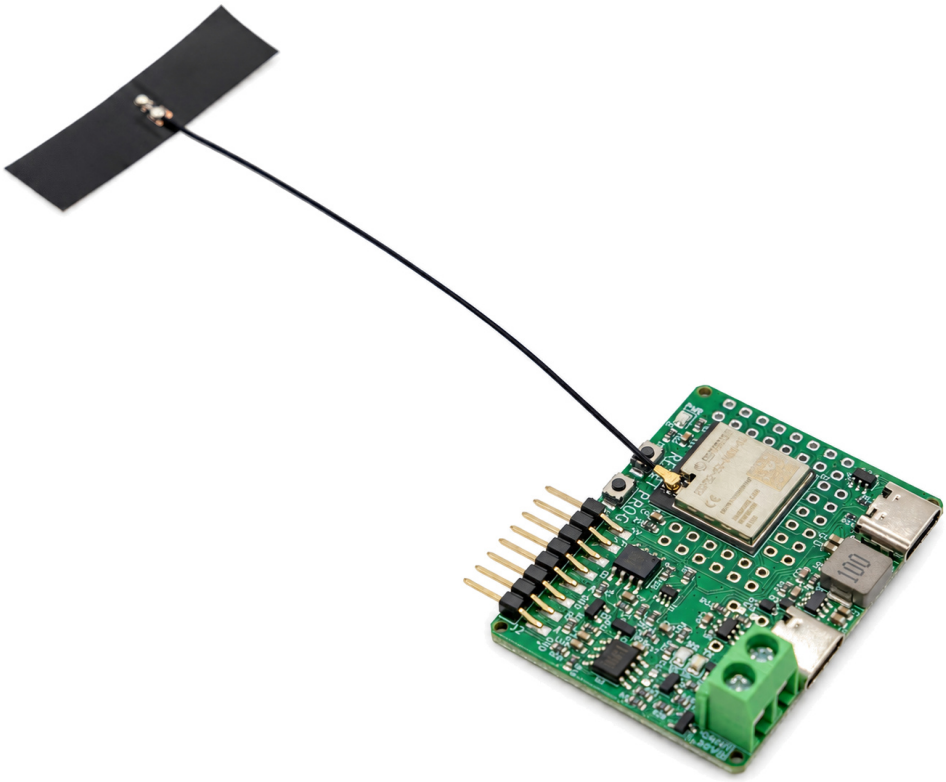
RS485 Controller · PCB revision 0.5



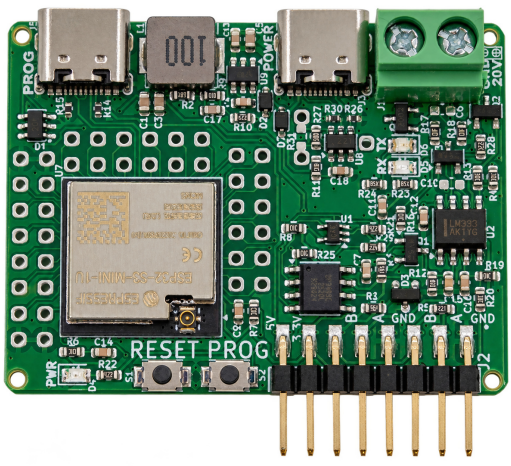
PRODUCT PHOTOGRAPHY

More views of the controller

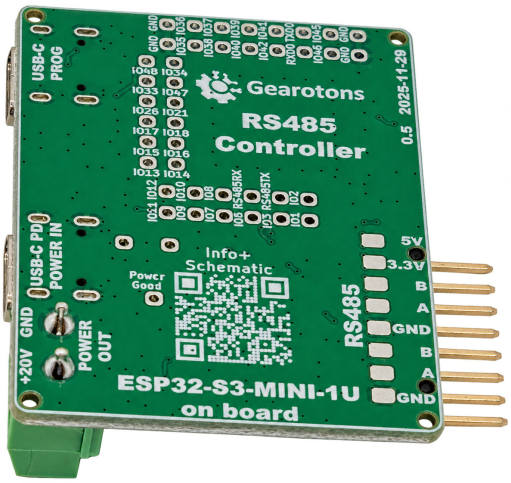
Included external antenna, USB-C and power connectors, and rear PCB labels.



Controller with the supplied external antenna connected.



USB-C sockets and motor-power terminal



Rear / label side

Mass: 11.7 g including the supplied external antenna.