ESP32 Arduino Autopilot Build (for wheel or tiller) – 2023

Item	Cost	Link
ESP32 devkit	**$9**	**https://www.ebay.com/itm/284415393405**
12v to 5v buck converter chip	**$1**	**https://www.ebay.com/itm/201986611110 or similar**
9-axis IMU compass (LSM303/L3G)	**$15**	**https://www.ebay.com/itm/202091223251**
IR Remote & Receiver (optional)	$5	https://www.ebay.com/itm/224263591887
Push button (optional)	$.50	Local shop
BTS7960 (IBT-2) Motor Controller	**$13**	**https://www.ebay.com/itm/112571698699**
Thermal pad for controller (optional)	$3	https://www.ebay.com/itm/313578863749
Motorbike belt drive kit with belt and 2 gears (1:4 ratio) (for wheel)	$45	https://www.ebay.com/itm/174775590807 (note: you can also 3D print or purchase your own gears. The small gear on my installation is 39mm and the large gear can range from 256mm to 306mm)
HTD 5mm x 15mm timing belt (optional based on installation needs)	$15	https://www.ebay.com/itm/405588785448?var=675888411548 (1250 mm for belt drive kit; 1350mm for 12” homemade large pulley)
Windshield wiper motor (wheel pilot)	$35	https://www.ebay.com/itm/163899761480 (May alternatively use power steering motor for more power)
Wiper motor box w/ clutch	?	Motor is secured inside a wood box using hose clamps. This box sits on top of a wood base that is secured to the steering pedestal with large hose clamps. The base and box are connected with a hinge and a homemade bolt latch. The hinge allows good belt tension underway plus ability to disengage motor linkage quickly. Note 2/1/26: experimenting with a 3D printed motor box and gears so you don’t need the motorbike belt drive kit. I’ll post it online when it’s tuned.
Linear Actuator (alternative to wiper motor for tiller pilot)	$86	https://www.ebay.com/itm/304592763080?var=603617299724 (300mm stroke, 300Newton, 45 mm/s)
Dupont cable jumper wires assortment	$6	https://www.ebay.com/itm/324089639653
Tiller linkage for actuator	?	Tiller bracket made from strip of 1.5” aluminum, 1/4” bolt, and two U-brackets. Cockpit seat socket made from 1/4” bolt with rubber bushing for top, and a 2” wood block glued under the cockpit locker hatch, with a 3/8” hole drilled and filled with epoxy and an embedded nut (stand a greased bolt up threaded through the nut while the epoxy is curing).
LCD (2-row) w/ I2C module (optional)	$7	https://www.ebay.com/itm/234062594780
Hose clamps	$6	For mounting motor on steering pedestal
Rudder position indicator (optional but useful for end-of-travel sensing)	$18	https://www.ebay.com/itm/164145915227 Note: it needs a 1k resistor on a third wire to make work. See diagram.
Wire	--	(USB cables for IMU & motor controller; 14/16 AWG for install)
Total (all options for wheel)	**$164**	**Excludes cost of scrap wood for box, various hardware (bolts, screws, U-brackets, hose clamps), enclosure (be creative!), and cables.**
Total (barebones with just ESP32, IBT-2, and IMU, no drive motor, rudder sensor, or linkage)	**$37**	**Assumes you have your own motor and steering linkage. All control is via onboard wifi HTML interface.**

Note: total cost could be cheaper if you are patient for parts to arrive from overseas.

To build:

Motor Linkage (for wheel):

- Attach small gear from belt drive kit to the windshield wiper motor. I did this by using red threadlocker to fix a nut onto the bolt on the end of the motor output, then fitting the gear snugly around the nut. Fill the center gap with JB Weld and hold level while it cures. Drill a small hole through the gear and bolt shaft and tap in a small piece of steel wire as a set pin.
- Connect the incoming 12v power from the controller, + to green and – to the loose wire. On my motor, fast speed was green and slow speed was yellow. Note: I used a 2-wire trailer connection from an auto parts store as a waterproof plug.
- Affix large gear from belt drive kit to the backside of your steering wheel using U-brackets.
- Note: If you want a higher gear ratio/more power, you can make your own large wheel gear using a round of wood with an inside-out timing belt glued to the outside. You could also 3D print a larger gear (working on this now as of January 2026).

LSM303 IMU (compass):

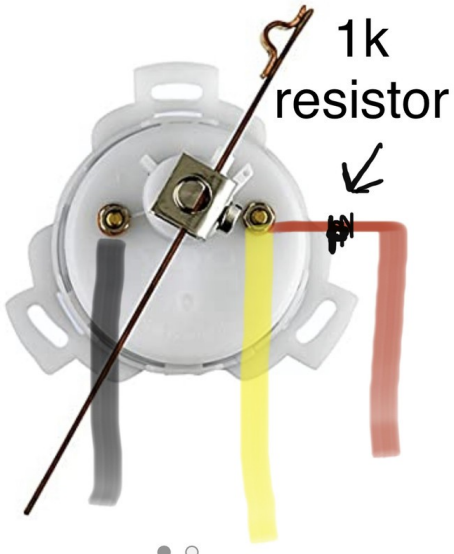
- Option 1: Remote stashable compass:
  - Cut both ends off of a USB cable no longer than 2m. Solder the wires from one end to VIN, GND, SDA, and SCL.
  - On the other end of the USB cable, attach female header pins for connecting to the ESP32. The lazy route is to clip, splice, and heat shrink Dupont wires onto the USB wire ends.
  - Glue the IMU to something you can firmly stick somewhere in the centerline area of the boat away from ferrous metals or electric fields. I used a tic-tac box.
- Option 2: direct attachment to ESP32
  - Don’t use a USB cable at all. Solder dupont wires to the IMU, glue the IMU to a plastic strip glued to the underside of the ESP, and plug in the IMU to the ESP. This whole arrangement fits inside a small prescription bottle.

IBT-2 motor controller:

- Decide where you want the controller relative to the motor/actuator and the ESP32 unit. The IBT-2 should not be right next to the compass or the electrical fields will interfere.
- Find a USB cable of appropriate length and cut off the heads. Expose the 4 wires on either end.
- Splice three female Dupont cables to the red wire in the USB cable. Splice one female Dupont wire to each of the other 3 USB cable wires.
- Connect the three red wires to VIN, L-EN, and R-EN. Connect black wire to GND and other two wires to LPWM and RPWM.
- Connect incoming 12V wire of at least 14AWG to the BATT +/- inputs
- Connect Motor +/- to motor or linear actuator. Polarity is reversible and doesn’t really matter.

Rudder Position Sensor wiring:

If you get the el cheapo rudder position indicator linked in the parts list, you will need to wire it like so (brown is the sensing wire)



ESP32 and components

- Run a 12v line from your boat breaker panel to the 12v>5v buck converter chip. Solder the +/- input to the input side of the buck converter. Cut the large head off a microUSB cable and expose the 4 wires inside. Connect the red and black wires to the +/- of the output pins on the buck converter. To waterproof this, I put a small piece of water hose over the connection and filled it with silicone. The microUSB male plugs into the ESP32.

- To power multiple components (e.g., IBT-2, IMU, LCD, rudder sensor) from the ESP32, cut a small rectangle out of a hobby breadboard and glue it to the underside of the ESP. This will be your power bus. Connect the VIN and GND pins to the +/- sides of the bus using Dupont wires. This will supply 5v to all components. You could also use the 3.3V pin on the ESP32 if your specific components need the lower voltage.

- Connect your components to the ESP board as follows:
  - IMU compass
    - VIN = + bus
    - GND = - bus
    - SDA = pin 21
    - SCL = pin 22
  - IBT-2 motor controller
    - VIN = + bus
    - GND = -bus
    - L-EN = + bus (splice with VIN)
    - R-EN = + bus (splice with VIN)
    - LPWM = 32
    - RPWM = 33
    - Connect incoming 12V +/- and outgoing motor +/- where indicated
  - Rudder sensor (optional but recommended if you use a wiper motor or an actuator without end of travel stop switches. The actuator linked above has EoT switches.)
    - VIN = + bus
    - GND = - bus
    - SENS = pin 34
  - LCD screen (optional)
    - VIN = + bus
    - GND = - bus
    - SDA = pin 21 (splice with compass IMU or designate additional SDA/SCL pins in code)
    - SCL = pin 22 (splice with compass IMU or designate additional SDA/SCL pins in code)
  - On/Off button (optional)
    - + = pin 5
    - - = - bus
  - IR receiver (optional)
    - VIN = + bus
    - GND = - bus
    - SENS (center) = pin 27

Uploading the code to the ESP32

- Download and install the Arduino IDE software on your computer.
- Install the ESP32 boards and libraries into the Arduino IDE. Here is a good tutorial: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/
- Make sure the following libraries are installed in your Arduino folder. They should all be able to be added from within the Library Manager in the IDE. If you can't locate one, it can be searched on the web and downloaded as a ZIP from its repository for manual installation through the *Sketch>Include Library>from .zip* menu option in the IDE.

    - <LiquidCrystal_I2C.h> // for the 16x2 LCD screen
    - <Wire.h>
    - <Bounce2.h> // use for red button toggle pilot on/off
    - <L3G.h> // IMU compass
    - <BTS7960.h> // IBT-2 motor controller library
    - <IRremote.h> // infrared remote control
    - <LSM303.h> // IMU compass
    - <WiFi.h>
    - <AsyncTCP.h> // NOTE: you need to use the library "Async TCP" (note the space) from ESP32Async.
    - <ESPAsyncWebServer.h>
    - <ElegantOTA.h>
    - <ArduinoJSON.h>
    - <esp_now.h>
    - <esp_wifi.h>

- Connect the ESP32 to your computer using the USB cable and select the right ESP32 board (or close enough)
- Upload the autopilot sketch and hold the "Boot" pin on the ESP32 until it's done. It will take a few minutes.
- After the first time uploading, any future changes you make to the code may be loaded via the wifi connection by exporting a binary and going to 192.168.4.1/update.

Code Build Notes – Library Edits Required:

- -- In cores\esp32\IPAddress.h, replace: const IPAddress INADDR_NONE(0,0,0,0); with: const IPAddress IP_ADDR_NONE(0,0,0,0);
- Find ElegantOTA.h and make sure the variable #define ELEGANTOTA_USE_ASYNC_WEBSERVER is set to 1
- Even with all the right libraries and a successful compile/upload, the UI webpage will not load without one additional tweak. The HTML file was bugging out because the ESPAsyncWebServer library uses % to mark placholder text (like for settings and heading information), but the css stylesheet for the rudder gauge also uses % for its normal meaning. If you want to edit the code for your own purposes rather than just upload my binary, you will need to follow the instructions on this page for changing all the % placeholders to $ within the WebResponseImpl.h file in your version of the ESPAsyncWebServer library: https://stackoverflow.com/questions/74649351/espasyncwebserver-request-send-p-problem.

Compass Calibration
To get stable compass readings and tilt compensation, you will need to run the Calibrate.INO sketch, located within the LSM303 library examples. After you've assembled the ESP32 and compass IMU, upload the calibration sketch to the device and use the Serial Monitor to record the XYZ min/max values for your IMU. (Note: the calibrate sketch and the Serial Monitor both need to be adjusted to 115200 baud rate instead of 9600 for the serial monitor to read properly). Once you've run the calibration sketch a few times and recorded the values, input them into the pilot code starting around line 300 of the main Autopilot.INO tab.

Operation*:*

1. Turn on power to the ESP32 and after a moment, connect to the SSID of the autopilot (current password is blank). Direct your browser to 192.168.4.1 to reach the pilot interface. Feel free to rename the SSID and password, located near the top of the main .ino file.

2. It's an autopilot! Turn it on and tell it to go somewhere.

3. The PID variables are tunable from within the HTML interface, but they reset every time you turn it on. To permanently change the default PID settings, rudder deadband, or magnetic variation offset, you need to change them in the main .ino file and reupload the arduino sketch to the ESP32.

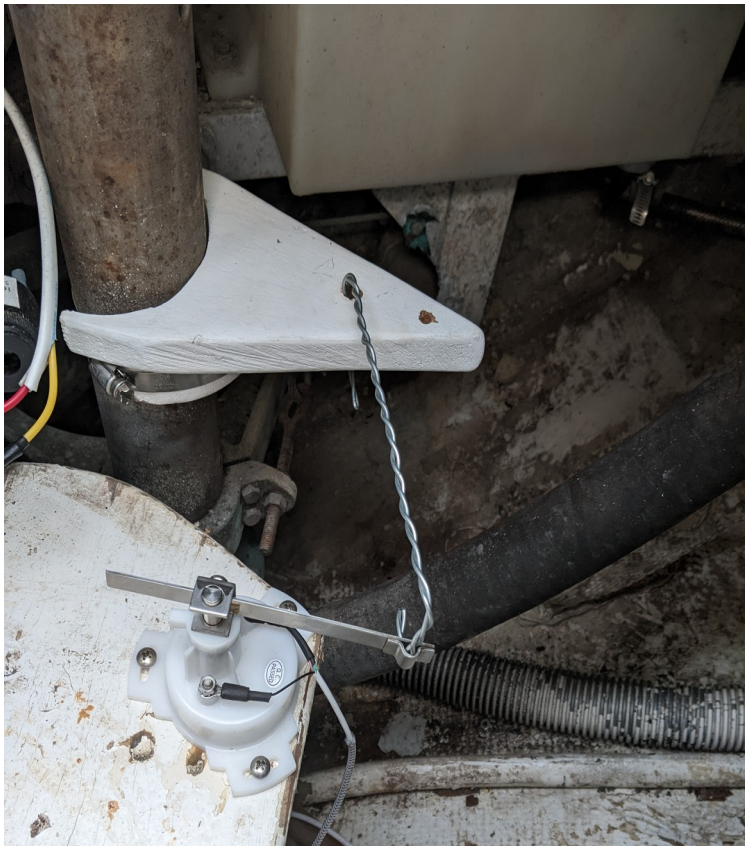
Photos of my hardware install for wheel and tiller (note: the tension pulley was replaced by the hinged box concept and holds belt tension better under operation. The base of the hinged box is attached to the pedestal with a T-shaped aluminum bracket with hose clamps.):

Prototypes of the motor linkage box. This is the hardest part of the whole build and takes some tinker engineering.

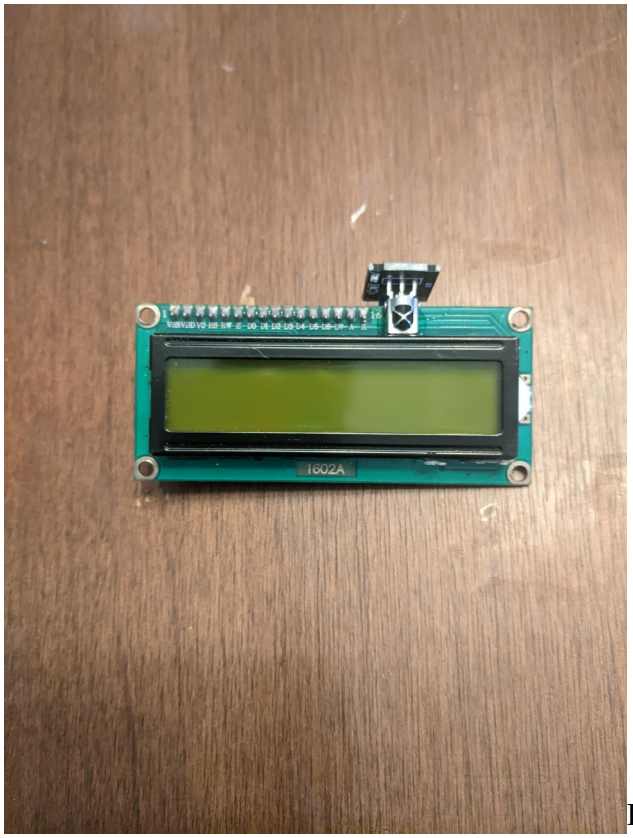
“Final” build with hinge box

The ESP32 “enclosure” that sits inside the boat.

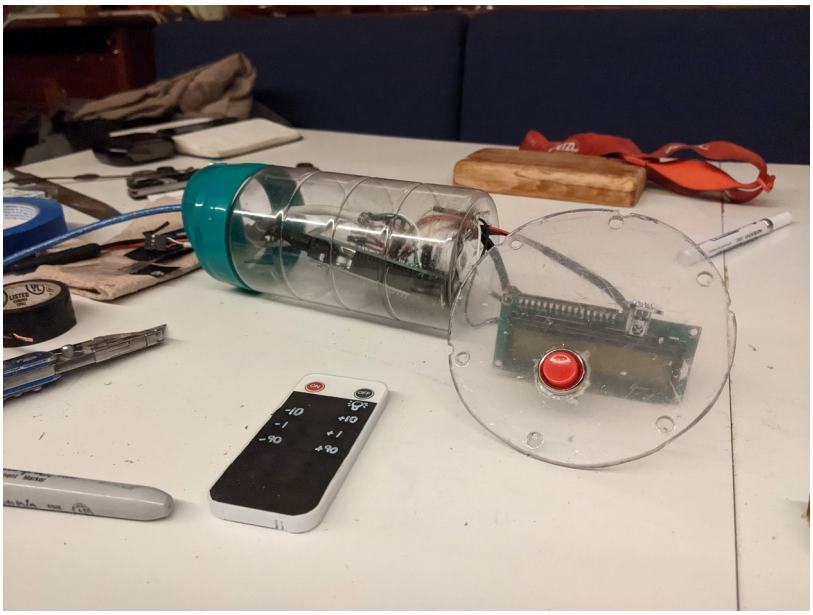
Rudder angle sensor and “vane” clamped to rudder post.



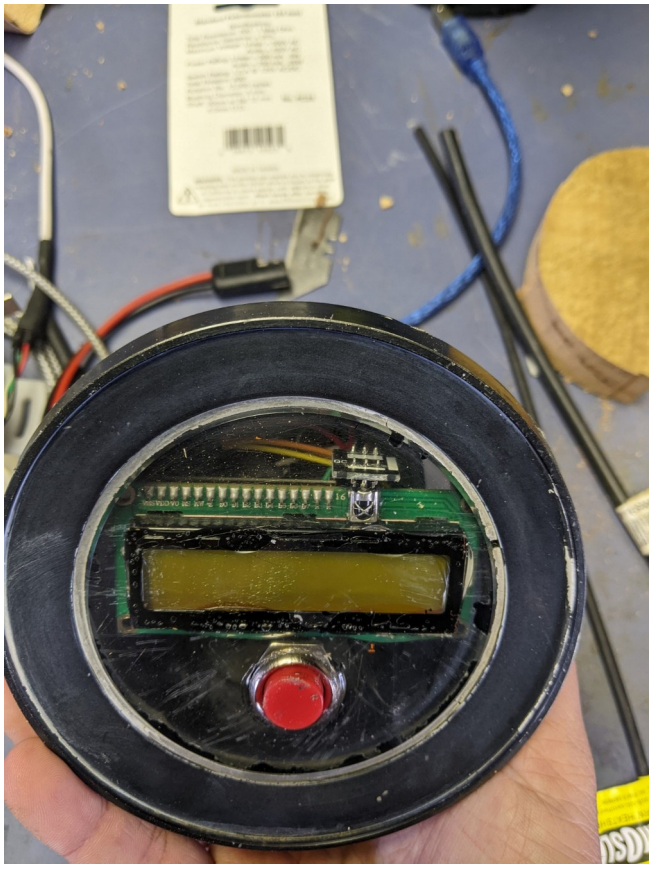
2026: Experimenting with 3D printing. This version has the box sliding up and down on a bracket, with a sliding lock at the bottom of the box to hold it in the down position during operation. To release, pull the handle and slide the box up a couple of inches.

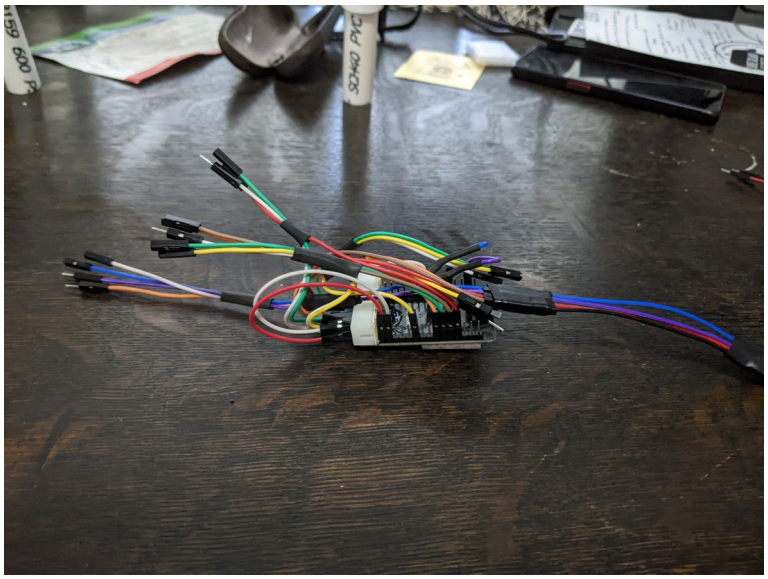


LCD with IR remote sensor and On/Off button



Early build of display with IR remote in foreground. Different remotes may use different HEX values and would take some work to decode.





The mess of dupont cables on the ESP32. Note the white bus bar built from a hobby breadboard.

Tiller Pilot Install (compass glued to back of ESP32. Mini battery only used for testing purposes, but the final install is tied to the 12v boat battery)