

SAMOVAR

Technical documentation

Firmware 7.00

Current version: <https://www.samovar-tool.ru/doc/en/device/>
Built 02.10.2026

Contents

Service section. Read this first :)

Where to start

- Building and flashing with PlatformIO

- ESP32 mounting

Minimal configuration

- Buzzer

- Atmospheric pressure sensors

- Temperature sensors

- Voltage regulators used

 - Regulator overview

 - How to connect a regulator to Samovar

 - Heating element

 - Using a boost heating element

- Contactor. RCD. Emergency shutdown

- Cooling

 - Using a diaphragm pump

 - Using a centrifugal pump

 - Without a pump

- Reflux level sensor

- Driver. Stepper motor

- Fraction distribution

- Emergency button. Additional emergency sensors

Recommended configuration

- LCD display, encoder, button

- How to connect a relay module correctly

Extended configuration

- Meter

- I2CStepper

- pH sensor and how to connect it

Additional features

- Bluetooth port monitor

Service Section. Read This First:)

Service information for those who help to extend and correct the build guide.

This information is intended for those who take part in creating this guide.

Address for sending materials and for communication about this Guide: device@samovar-tool.ru

Please send text in .doc or .txt format and images in .jpg. Send the text separately from the photos. You can combine everything into one archive file.

The guide was created on the basis of the general help. If you find a broken link, fix it or replace it with a link to the current section.

- If a section is "in the wrong place", that is not a problem - it is trivially easy to move it in the tree. So let's not focus on it for now.

What is proposed:

The structure of the guide may be refined as the project develops.

Each section should contain as much information as possible about a specific piece of hardware:

A photo, a link to where to buy it, the original name (so you can find it when the link dies), how to connect it, peculiarities, possible replacements, modifications, etc.

A link is strongly desirable to the post discussing the device on the forum - it is easy for you to find your own posts, while somebody else would have to read through the whole forum, and more than once... .

That is, one section per specific device; if a section gets bulky, it is easy to make a subsection. (For example, like the one about [regulators.](#))

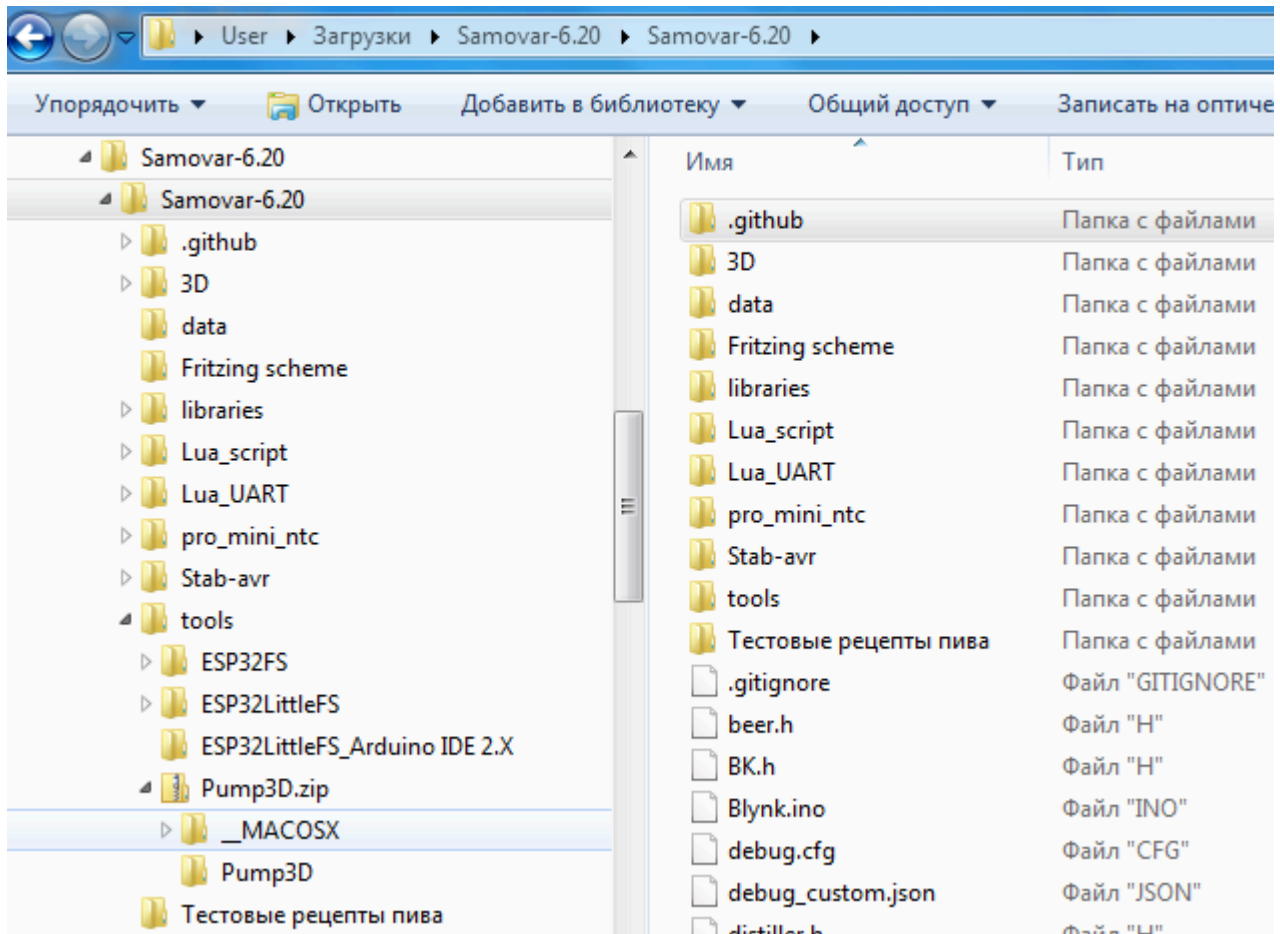
No additional information about a specific device goes in other sections. Mentions - as many as you like . Just make a link to that device's section and that's all. Otherwise it is very easy to get confused while compiling.

There may be exceptions - for example, the pressure sensor in the boiler. The description of the implementation with the MPX5010D connected specifically to Samovar starts in one place, and continues in the section on the Meter by dranek.

Authorship. If anyone wishes, a link to the author of the material will be placed at the end of the section.

Where to Start

First, download the [release](#) with the latest version of Samovar, unpack it on your computer and look through its contents.



What is in the folders:

Note

Start with the **README.md** file in the project root: it lists the supported boards, pin restrictions and the installation procedure.

- **.github** - a software folder.
- **3D** - models for 3D printing. Fraction collection tables, funnels, etc.
- **data_raw** — the web interface sources. The **data** folder is the ready-made image of the device file system; it is rebuilt entirely by the **python3 tools/build_web_assets.py** command, so it is not edited by hand. The **program_*.txt** files contain the default programs.
- **Fritzing scheme** - assembly diagrams.
- **libraries** - a software folder (libraries).
- **Lua_script** - a description of the Lua scripting language and various very useful scripts.
- **Lua_UART** - how to make a remote standalone COM port monitor.

- [pro_mini_ntc](#) - how to make an add-on for Samovar (the Meter); it can also work as a standalone device, as a measuring device.
- [Stab-avr](#) - a version of a power stabilizer built from ready-made components that Samovar can control.
- **tools** — the configurator (samovar_configurator.py), the web interface build scripts and the automatic project checks.
- doc - descriptions of the firmware subsystems (Markdown files).
- **partitions** and **ESP32-S3** — flash memory partition tables for ESP32 and ESP32-S3; PlatformIO uses them.
- Stab-avr, Lua_UART, pro_mini_ntc - separate Arduino sketches; they are not part of the Samovar build.
- Test beer recipes - several ready-made beer recipes to load into Samovar.

What is the minimum configuration Samovar can be built in?

For the first start, an ESP32 board and USB power are enough. The firmware can be built either [in the Arduino IDE](#) or in PlatformIO. The hardware settings are in **Samovar_ini.h**; user overrides that must be preserved on updates are in **user_config_override.h**. Ready-made binary files are not included in the release.

Detailed instructions: [building in the Arduino IDE](#) and [building and flashing with PlatformIO](#).

Note:

Sensors and actuators whose presence Samovar does not check, because there is no feedback:

- *Contactor / voltage regulator power-on (relay No. 1)*
- *Additional heater relay / voltage regulator boost (relay No. 4)*
- *Buzzer*
- *Water opening valve (relay No. 3)*
- *Reflux level sensor*
- *Peristaltic pump stepper motor*
- *Fraction collection table servo*
- *Cooling pump*
- *Atmospheric pressure sensor*
- *Boiler pressure sensor.*
- *Stirrer (relay No. 2)*

Sensors and actuators whose presence Samovar does check, if:

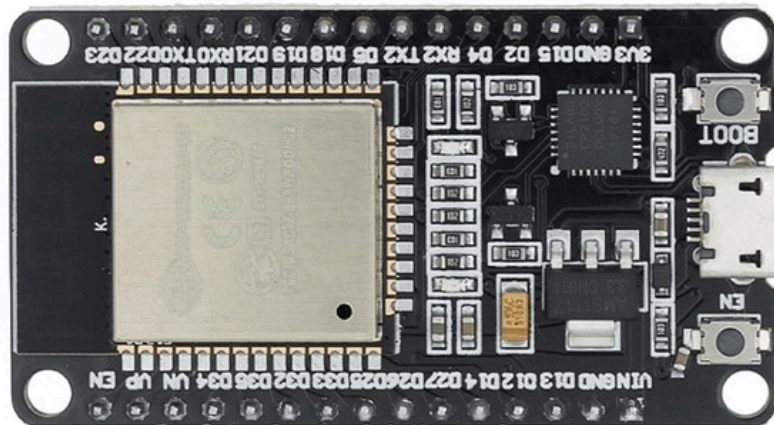
- *Temperature sensors - if the sensor was present at the start of the process. If it is missing, messages will come regularly.*
- *Water flow sensor – if its use is enabled.*
- *Heating regulator – if the regulator use check is enabled.*

Then [set up the Wi-Fi connection](#) and open the Samovar web interface. On the classic ESP32 DevKit, the "Turn heating on/off" («Включить/Выключить нагрев») command toggles GPIO2, which on many boards is connected to the built-in blue LED. On ESP32-S3, whether such an LED exists depends on the specific board.

An informative article about [ESP32](#)

Explanation.

ESP32 is a chip that is mounted on a board. The board additionally has a voltage converter, a Wi-Fi module, Bluetooth and possibly much more. Here, the term ESP32 means a board with the chip installed.



ESP-32 30P

There are many board variants.

The main board of the project is the **ESP32 DevKit V1 30 pin**, PlatformIO environment **Samovar**.

LILYGO T-Relay 4 (the same **Samovar** environment; the board is selected in the configurator) and

ESP32-S3 DevKitC-1 (environment **Samovar_s3**) are also supported. LILYGO and ESP32-S3 have a different pinout (see `Samovar_pin.h`); the Samovar 1.3.2 printed circuit board for the 30-pin DevKit does not fit them.

ESP32 is a chip

DevKit V1 - the version

30 pin - the number of contacts (legs, pins) on the board

CH340C is a USB bus data transfer adapter chip

Type-C - the USB connector form factor

Where to buy.

[AliExpress](#) (Search for: ESP32 smart board, USB flash CH340C WiFi + Bluetooth, ultra-low power consumption («Умная плата ESP32, флэш-накопитель USB CH340C WiFi + Bluetooth, сверхнизкое энергопотребление»))

[Ozon](#) (Search for: ESP32 WROOM-32 development board WiFi Bluetooth-compatible development board for IOT («ESP32 WROOM-32 Плата для разработки WiFi Bluetooth-совместимая плата для разработки IOT»))

If you are unsure whether you have chosen the right ESP32 model, ask on the forum.

Now you need to connect Samovar to the equipment.

First, decide how the [ESP32 will be mounted](#).

Then decide on the layout options of the devices. It will be either a single unit or several separate units connected to each other.

And finally, start the build from the [minimum configuration](#), gradually adding the necessary components until the most ambitious goals are achieved.

Questions and answers. FAQ

*** Can the ESP32 DevKit V1 38 pin board with 38 pins be used?**

*Yes, it can, but it cannot be mounted on the printed circuit board designed for **ESP32 30 pin**.*

Installing PlatformIO and Flashing Samovar for the First Time

Samovar supports two full installation methods: **Arduino IDE** and the **PlatformIO-based configurator**. This page describes PlatformIO. For Arduino IDE, use the section ["Preparing the ESP32 for Upload"](#).

The configurator lets you choose the board and equipment, save settings, flash the ESP32, upload LittleFS separately, and open the serial monitor. PlatformIO is a program that builds the source code and writes it to the board.

Windows 10/11 (64-bit only): download the [latest Samovar release](#), unpack the archive to a simple path, for example **C:\Samovar**, connect the ESP32 with a USB cable that carries data, and double-click **flash_windows.bat**. The batch file checks Python and PlatformIO, installs them if necessary and opens the setup window.

macOS and Linux: open Terminal in the unpacked project folder and run:

```
chmod +x flash_macos_linux.sh
./flash_macos_linux.sh
```

The script checks Python with Tkinter and PlatformIO, installs them if necessary and opens the same configurator.

If macOS reports Operation not permitted or getcwd: cannot access parent directories: move the project folder from Downloads to your home folder and run the script there. Or allow Terminal access to the Downloads folder in "System Settings → Privacy & Security → Files and Folders", then fully restart Terminal.

On macOS, automatic installation of Python and Tkinter requires Homebrew. On Linux, apt, dnf and pacman are supported; the system may ask for the sudo password - the administrator password of the computer.

Working in the configurator: choose ESP32 DevKit, LILYGO or ESP32-S3, set the equipment and, if needed, the initial Wi-Fi credentials. The other parameters are arranged on the tabs "General" («Основные»), "Temperatures" («Температуры»), "Regulator" («Регулятор»), "BK" («БК»), "NBK" («НБК»), "Sensors" («Датчики»), "Pumps" («Насосы»), "Equipment" («Оборудование»), "Stepper motor" («Шаговый двигатель»), "Rectification" («Ректификация»), "Network" («Сеть») - these are the same directives that used to be edited in `Samovar_ini.h`. Press **"Save settings"** («Сохранить настройки»), then **"Flash"** («Прошить»). The settings are saved to the file **user_config_override.h** next to the project and are not overwritten when the sources are updated. The build and flashing log is at the bottom of the window.

For a new board, press **"Upload LittleFS"** («Загрузить LittleFS») separately. LittleFS is a memory area holding the web interface and user files. Uploading it again may replace programs, Lua files and logs, so the configurator asks for confirmation. Before uploading again, save the user files you need through the Samovar file editor.

The **"Serial monitor"** («Монитор порта») button opens a separate diagnostics window (speed 115200). If flashing stays on Connecting... for a long time, hold BOOT, briefly press RESET/EN and release BOOT after writing begins.

Update over Wi-Fi. Subsequent firmware updates can be done without a cable. Press **"Refresh"** («Обновить») next to the "Port or address" («Порт или адрес») field: besides COM ports, the list includes Samovar devices found on the network and the address the device reported after the **"Get IP"** («Получить IP») button in the serial monitor. You can also enter the address manually (for example 192.168.1.37 or samovar.local). Select the device instead of a COM port and press the same "Flash" or "Upload LittleFS". Conditions: the firmware on the device is built with **"Allow update over Wi-Fi"** («Разрешить обновление по Wi-Fi») enabled (the "Network" tab), the computer and Samovar are on the same network, and the firewall allows incoming connections for python.exe. Full flash erase, the serial monitor and **"Reboot ESP"** («Перезагрузить ESP») work only over USB.

The **"File editor"** («Редактор файлов») button opens the device files (programs, Lua scripts, web interface pages) right in the configurator, and **"Open in browser"** («Открыть в браузере») opens the

Samovar web interface.

Manual method via PlatformIO:

pio run -e Samovar -t upload - ESP32 DevKit or LILYGO;

pio run -e Samovar_s3 -t upload - ESP32-S3;

pio run -e <environment> -t uploadfs - upload LittleFS;

pio run -e <environment> -t monitor - open the serial monitor.

Ready-made environments with a different regulator: **Samovar_rmvk** (RMVK), **Samovar_sem** (SEM_AVR), **Samovar_no_power** (no regulator, with UART); **Samovar_lua** - with Lua, **Samovar_alarm_button** - with the emergency button. User parameters for a manual build are set in **user_config_override.h** following the example in **user_config_override.example.h**.

After the first flashing, go on to [connecting Samovar to Wi-Fi](#). If the network data is already set in the configurator and the board memory does not yet hold another network, the firmware tries to connect to it automatically.

ESP32 mounting

Many peripheral devices are connected to the ESP32. So the ESP32 has to be placed and the connections organized.

Important. The pinout and the printed circuit board version 1.3.2 given below apply to the ESP32 DevKit V1 30 pin. For LILYGO T-Relay 4 (the environment **Samovar**, the board is selected in the configurator) and ESP32-S3 (the environment **Samovar_s3**) use the pinout from [Samovar_pin.h](#); the 1.3.2 board does not physically fit them.

Назначение в проекте ← Разъем на плате	Обозн.	№ вывода	ESP-WROOM-32 DevKitC v1	№ вывода	Обозн.	Разъем на плате → Назначение в проекте
--	--------	----------	-------------------------	----------	--------	--

Выход 3,3 В	3V3	1
Общий, масса	GND	2
Мешалка ← (Реле №2) ← IN2	D15	3
Контактор (PM2) ← (Реле №1) ← IN1	D2	4
Управление охлаждением ← PUMP	D4	5
Регулятор напряжения ← UART RX	D16	6
Регулятор напряжения ← UART TX	D17	7
Датчики температуры ← DS18B20	D5	8
Энкодер ← S1 ← ENC1	D18	9
Энкодер ← S2 ← ENC1	D19	10
Дисп+ давл ← SDL_LCD+BME680	D21	11
Системный, UART_0: RX	GPIO3	12
Системный, UART_0: TX	GPIO1	13
Дисп+ давл ← SDA_LCD+BME680	D22	14
Кнопка энкодера ← KEY(ENC1)	D23	15



30	V5	Вход. Питание 5В
29	GND	Общий, масса
28	D13	IN4 (Реле №4) → Разгон
27	D12	BUZ1 → Бипер,звук
26	D14	IN3 (Реле №3) → Охлаждение (вкл/выкл)
25	D27	Level → Датчик уровня флегмы
24	D26	STEP - Драйвер ШД
23	D25	SERVO1 → Сервопривод
22	D33	EN – Драйвер ШД
21	D32	DIR – Драйвер ШД
20	D35	ALARM_BTN → Аварийная кнопка
19	D34	нет → LUA_PIN
18	D39	BTN – Универсальная кнопка
17	D36	WATERSENSOR → Датчик потока ОЖ
16	EN	Системный Сброс [LOW]

ESP32 pin limitations. The GPIO34–39 pins are input-only and have no internal pull-ups: for the emergency button (D35), the universal button (D39) and the flow sensor (D36) an external 10 kΩ resistor to 3.3 V is mandatory, otherwise there will be false triggering and an emergency stop right after power-up. These resistors are already provided on the printed circuit board. Run the wires to the flow sensor separately from the heating element power cables. Pin D15 (relay No. 2, stirrer) is pulled up in hardware at reset — the relay on it switches on briefly at every boot; use an active-low module, the relay polarity is set in the Samovar settings (relay 1–4 fields). The I2C bus pull-ups (display, pressure sensor, I2CStepper) are 4.7–10 kΩ.

Using the printed circuit board.

A special printed circuit board was developed for using the ESP32_30pin with Samovar. The following is installed on it:

- the ESP32 itself,
- a 12 to 5 volt power converter,
- a stepper motor driver,
- a sound buzzer,

- all capacitors and resistors must be installed. Recommended values:

Capacitors:

C1, C2 - electrolytic capacitors $500\ \mu\text{F} \times 25\ \text{V}$ (or 1000×25)

C3, C4 - electrolytic capacitors $500\ \mu\text{F} \times 16\ \text{V}$ (or 1000×16)

Note

The recommendations give values with a large voltage margin. With a voltage margin, even a not very good electrolytic capacitor will last longer and work more reliably.

But if you want to and are sure of your components, you can:

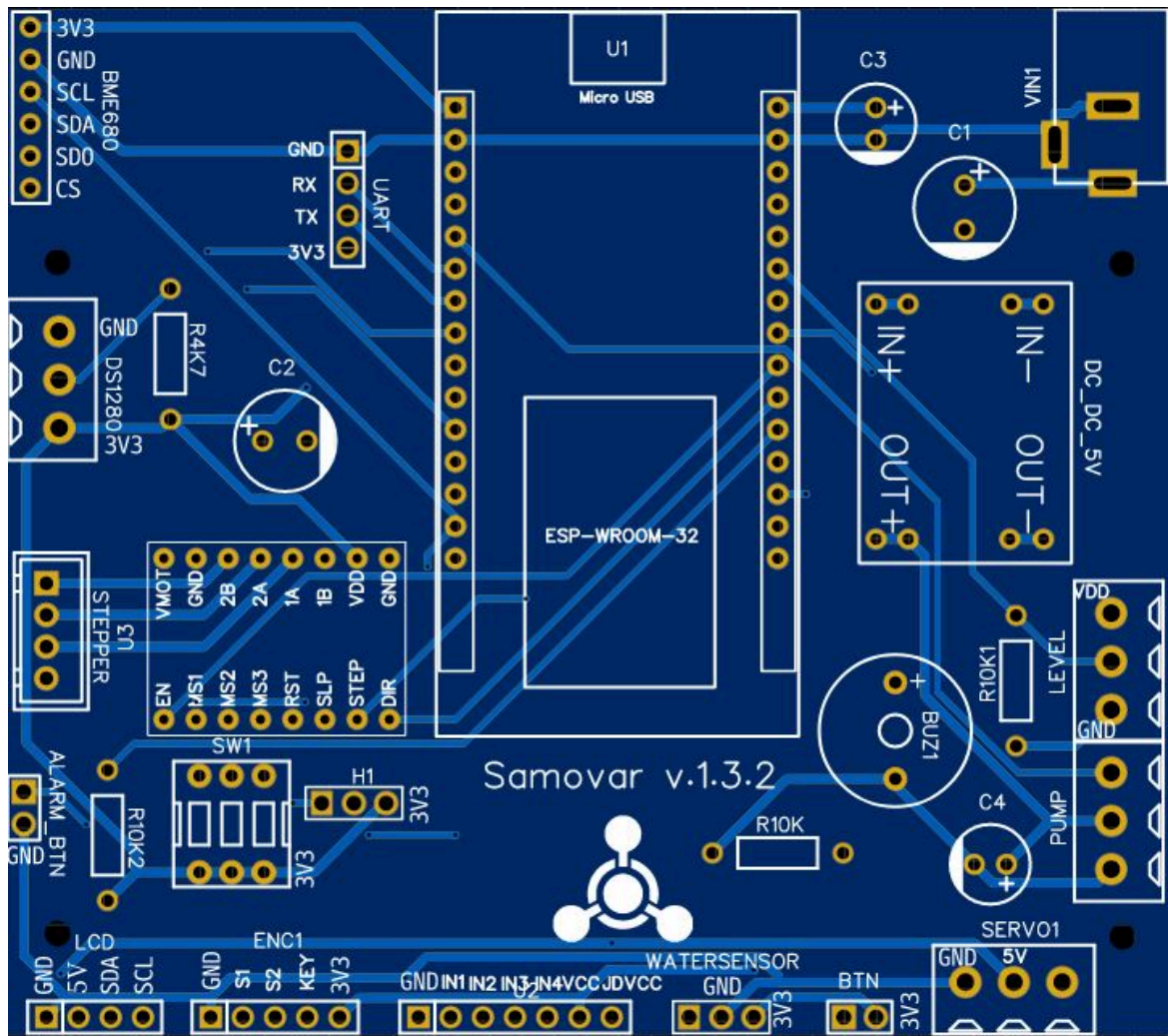
rate C1, C2 for 16 volts

C3, C4 - for 6.3 volts.

Resistors: R1-R3 = $10\ \text{k}\Omega$, R4 = $4.7\ \text{k}\Omega$. All 0.125 watt.

- connection connectors. (*note: some of the connectors can be left out and the wires soldered directly.*)

It is recommended to install all the elements on the board right away: the buzzer, the atmospheric pressure sensor, resistors and capacitors, the pin header for the stepper driver, switches and connectors. Even if you do not plan to use some of the equipment at the moment. Later this will make it much easier to expand Samovar's capabilities.



Mounting locations and elements:

- [ESP-WROOM-32](#) mounting location for the [ESP32](#). It is recommended to install it on a pin socket (female) connector
- [BME680](#) A place to connect an atmospheric pressure sensor (BMP180, BMP085, BMP280, BME280, BME680).
- [DS18B20](#) connection of temperature sensors. On the right (on the board) is a 4.7 kΩ power pull-up resistor for the sensors.
- [STEPPER](#) - connection of the NEMA17 stepper motor of the peristaltic take-off pump. On the right is the place for the stepper driver.
- [SW1 and H1](#) - stepper step divider and additional control of the stepper driver
- [ALRM_BT](#) Connection of the emergency button and emergency sensors.
- [LCD](#) - connection of the LCD2004 display (no other display is supported)
- [ENC1](#) - connector for the encoder with a button.
- [IN1IN2IN3IN4](#) connection of the relay module.
- [WATERSENSOR](#) - cooling flow sensor connector
- [BTN](#) - multifunction button

- [SERVO1](#) - connection of the servo drive of the fraction selection table
- [PUMP](#) - cooling pump control
- [LEVEL](#) - connection of the reflux level sensor
- [BUZ1](#) - 3 volt active buzzer
- DC_DC_5V - 12 to 5 volt voltage converter.
- VIN1 - 12 volt connector
- C1, C2 - electrolytic capacitors 500 μ F $\times$ 25 V (or 1000 $\times$ 25)
- C3, C4 - electrolytic capacitors 500 μ F $\times$ 16 V (or 1000 $\times$ 16)
- All resistors are 10 k Ω , except one 4.7 k Ω resistor

Where to get the board?

- Ask in the support thread. Perhaps someone who has already built one has a spare and can share it.
- Order it yourself.

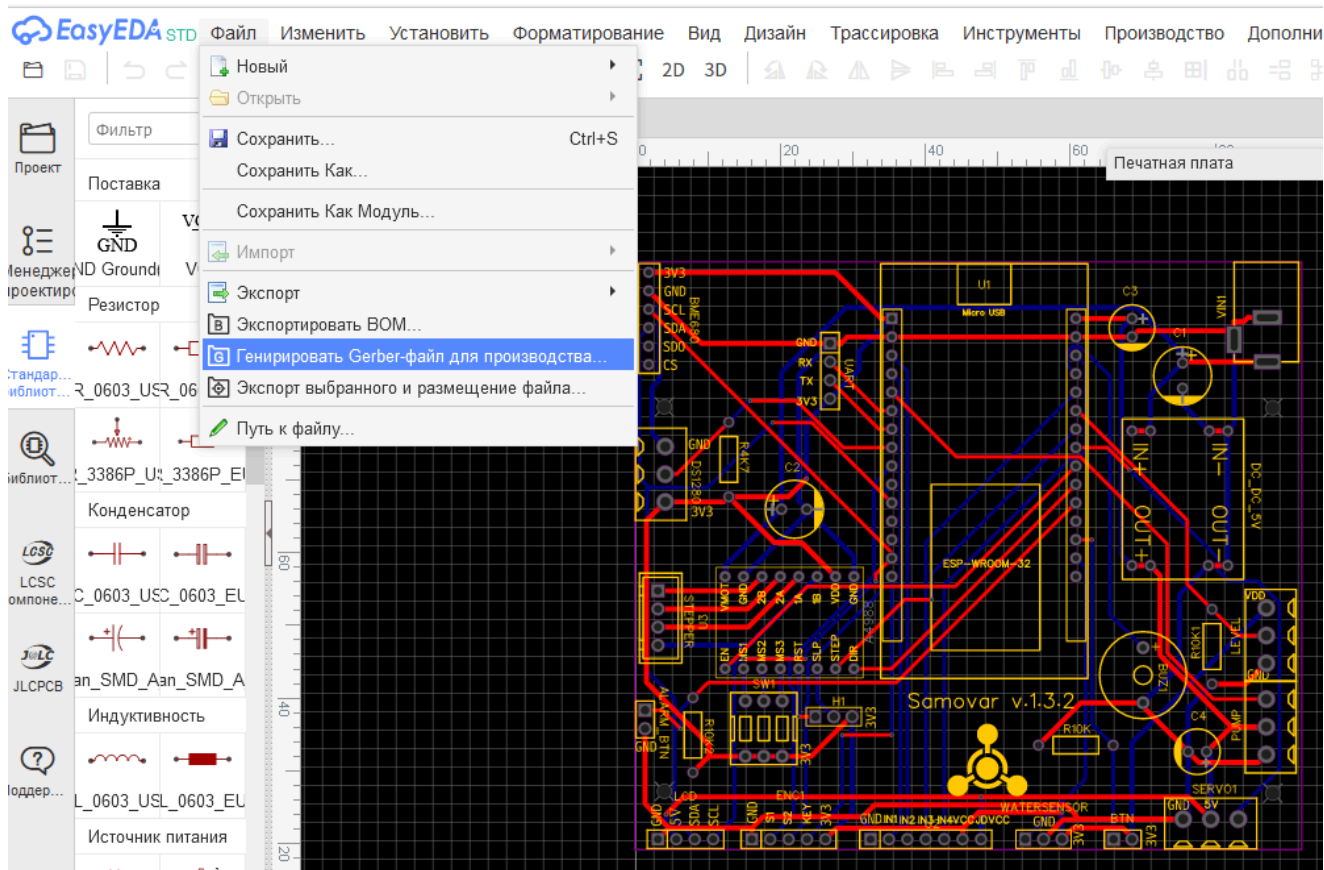
The developed board is hosted on easyeda.com. At the moment it is not possible to order manufacturing through this service. But you can use the service.

For manufacturing we need the files in Gerber format, which is a description of the printed circuit board project for manufacturing.

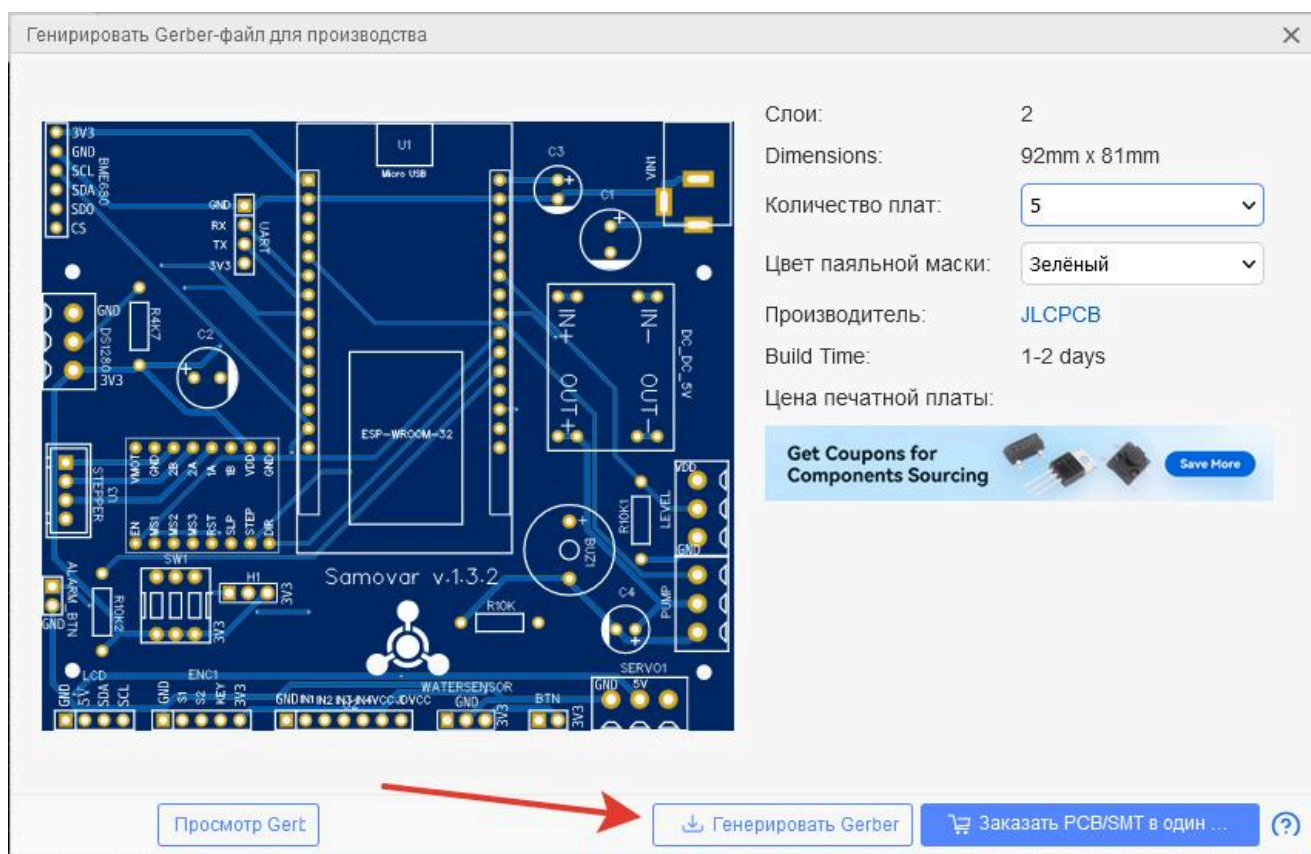
The Gerber archive for manufacturing of the printed circuit board version 1.3.2, as of September 2024, can be downloaded [from this link](#) from our server.

Or:

Go through the menu: File>Generate Gerber file for production... (in the Russian-language easyeda interface: «Файл>Генерировать Gerber-файл для производства...»)



Click, and after the check prompt choose: "No, generate Gerber file" («Нет, генерировать Gerber-файл»), a window will appear



You can change:

- the number of boards - a multiple of 5 (you cannot order fewer than 5).
- the solder mask color - this is the color of the board. Choose green or blue.

Other colors may cost extra.

Click generate Gerber. The created file will be saved to your computer as a zip archive.

Now you need to order manufacturing.

There are plenty of services that manufacture and, most importantly, ship. For example nextpcb.com there are instructions on the site or on the internet, payment via AliExpress.

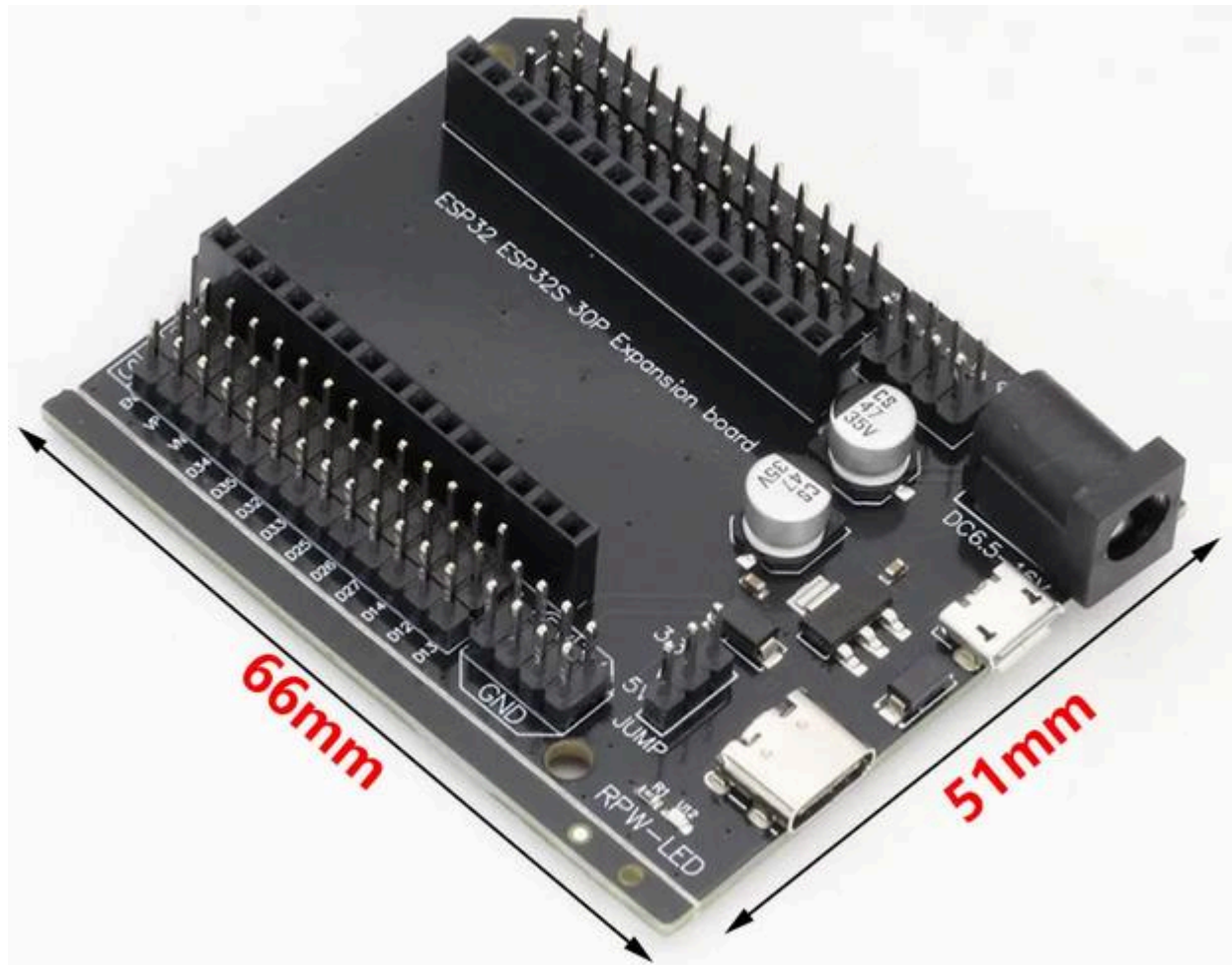
Or go straight to AliExpress and type in the search "custom printed circuit board manufacturing". Seller offers will appear. Choose by reviews.

The ordering principle is the same - you send the seller the whole Gerber archive (if the file name is in Cyrillic - rename it to Latin). The seller determines the price and tells you. If it suits you, you pay - and in about a month you have the printed circuit boards.

The cost of 5 boards with delivery is approximately 15-20 USD.

Using a breadboard (shield)

A special board is made for the ESP32 ([ESP32 Shield 30Pin breadboard expansion board](#)) that allows quick prototyping of various devices. It looks like this:



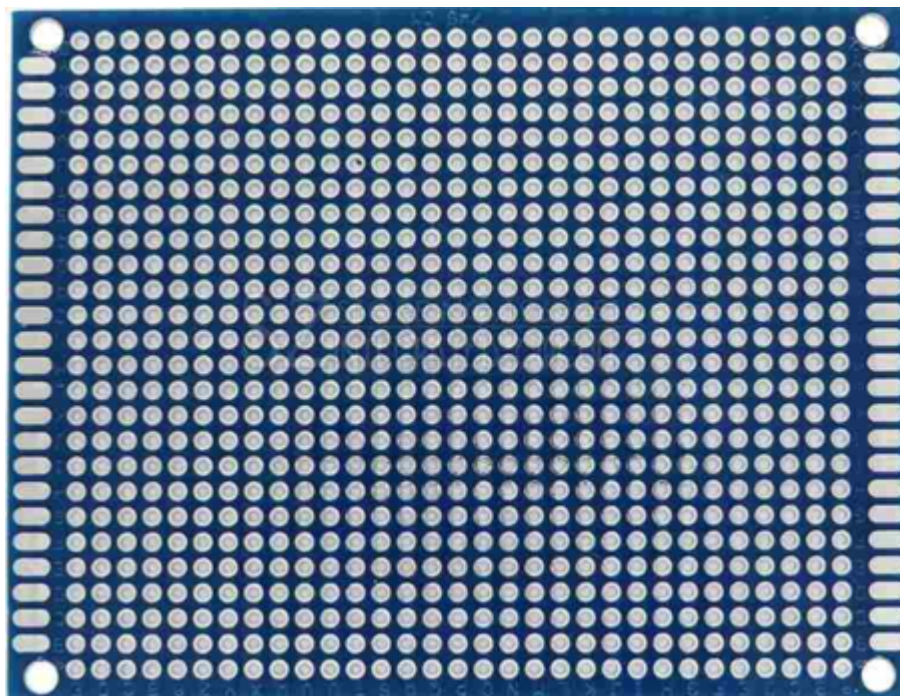
The shield already has a 16...6.5-5v power converter and USB connectors - 5 volt power. The ESP32 is inserted into a female pin header, and the peripherals are connected to the pins (male header). All wiring has to be done with wires; the resistors and some of the capacitors are soldered either on the shield itself or on the enclosure connectors.

Using a shield is a decent solution, and quite inexpensive. But for the final assembly it is highly desirable to solder all wires rather than leave them on detachable connections, otherwise failures will follow one after another.

Shields are also made for ESP32_38 pin. boards, which lets you use an ESP32 with 38 pins too.

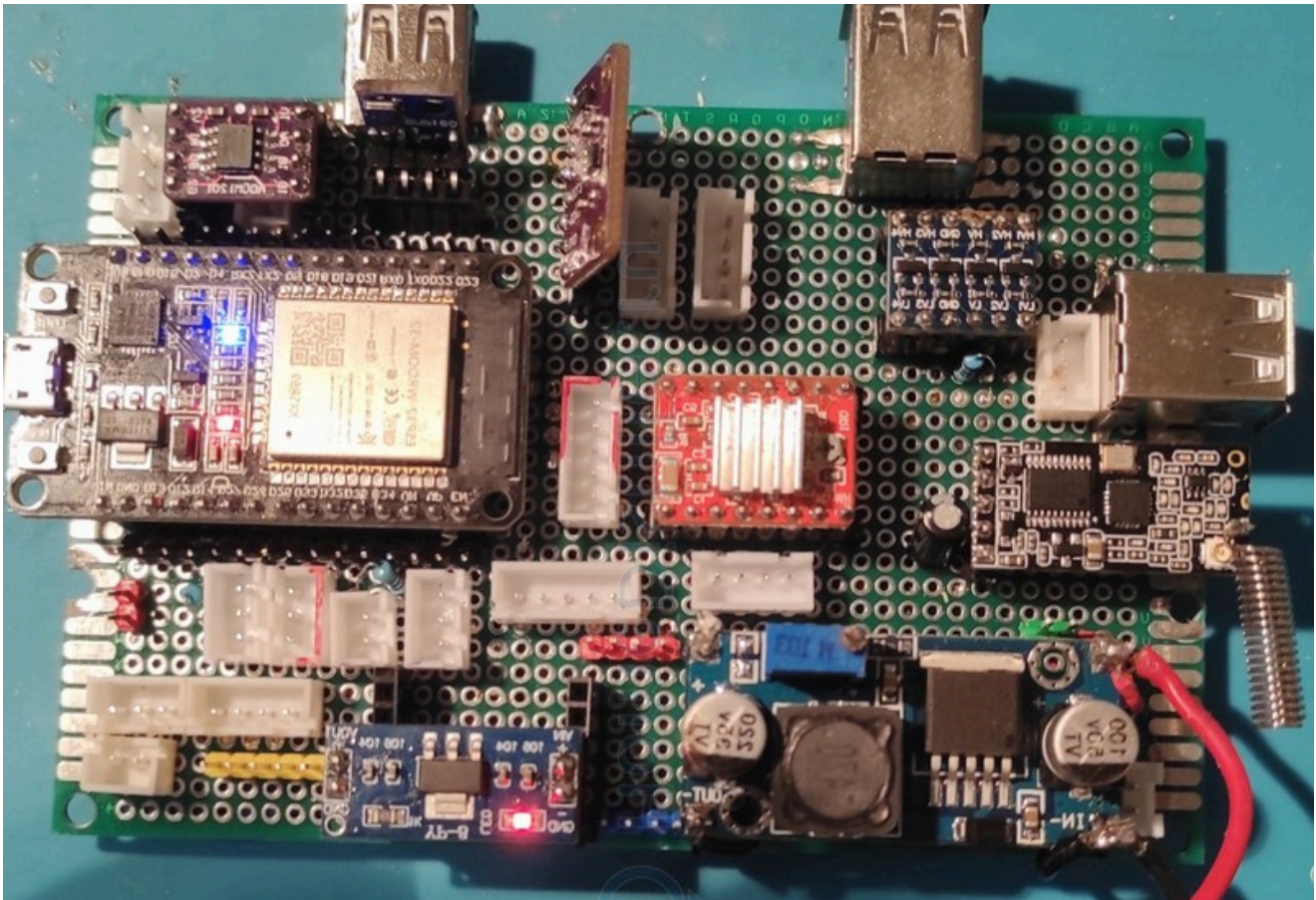
Using a soldering breadboard.

You can use a printed prototyping board designed for soldering components. (A printed circuit board prototype, breadboard, universal board)



All the elements can be placed on such a board, and there will even be room left for future upgrades. But working with such a board requires certain skills. A warning - building a Samovar on such a board is not exactly easy. Nevertheless, the project can be implemented this way.

An example of building Samovar on such a breadboard:



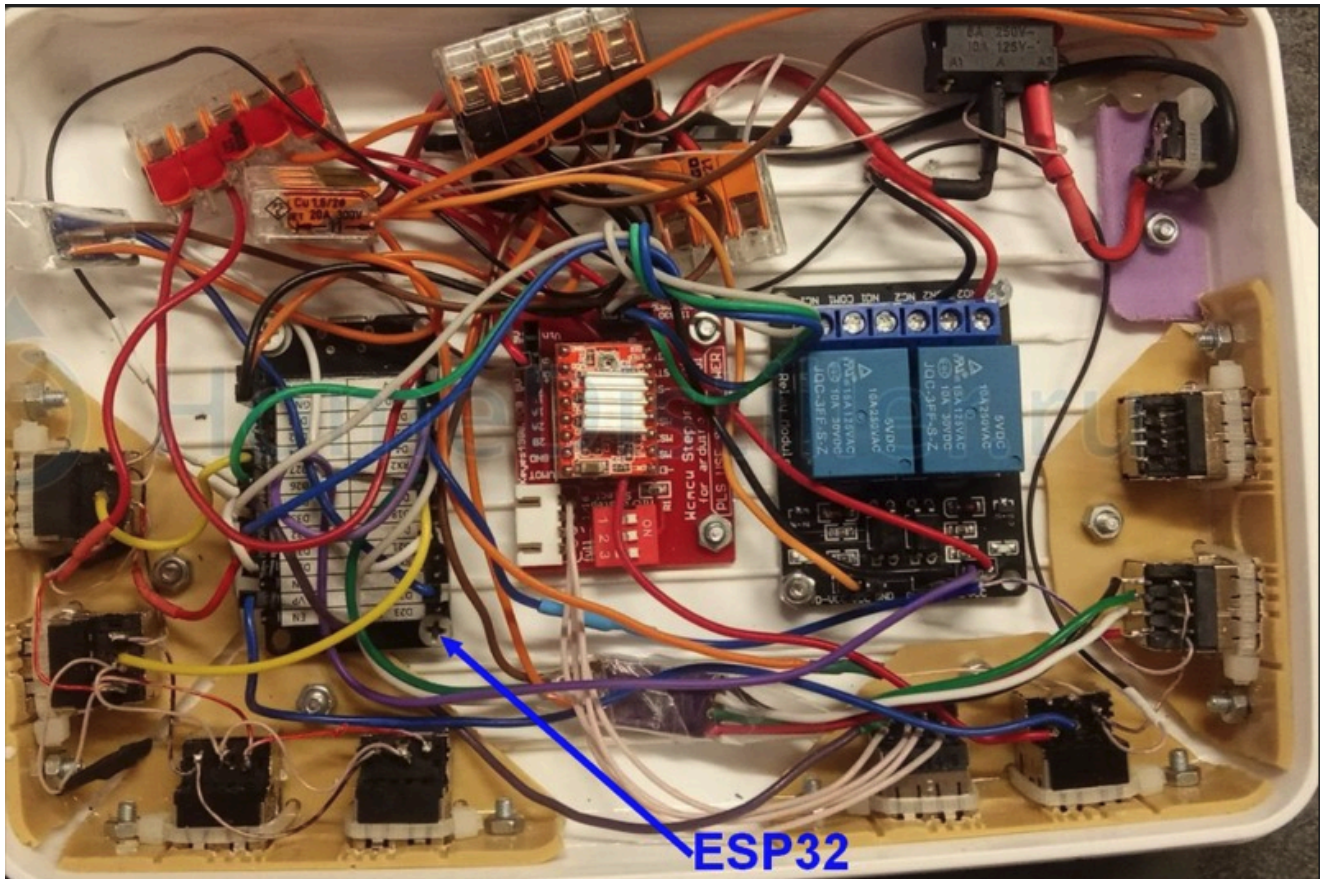
More photos [in the forum thread](#).

Assembly without a board

And finally the last option - without any printed circuit boards. Point-to-point wiring.

A label with the pin designations is printed on a printer and glued to the underside of the ESP32, which is turned upside down and screwed to the enclosure on standoffs (for cooling). The wires are attached to the ESP32 pins with connectors such as Dupont.

An example of an implementation:



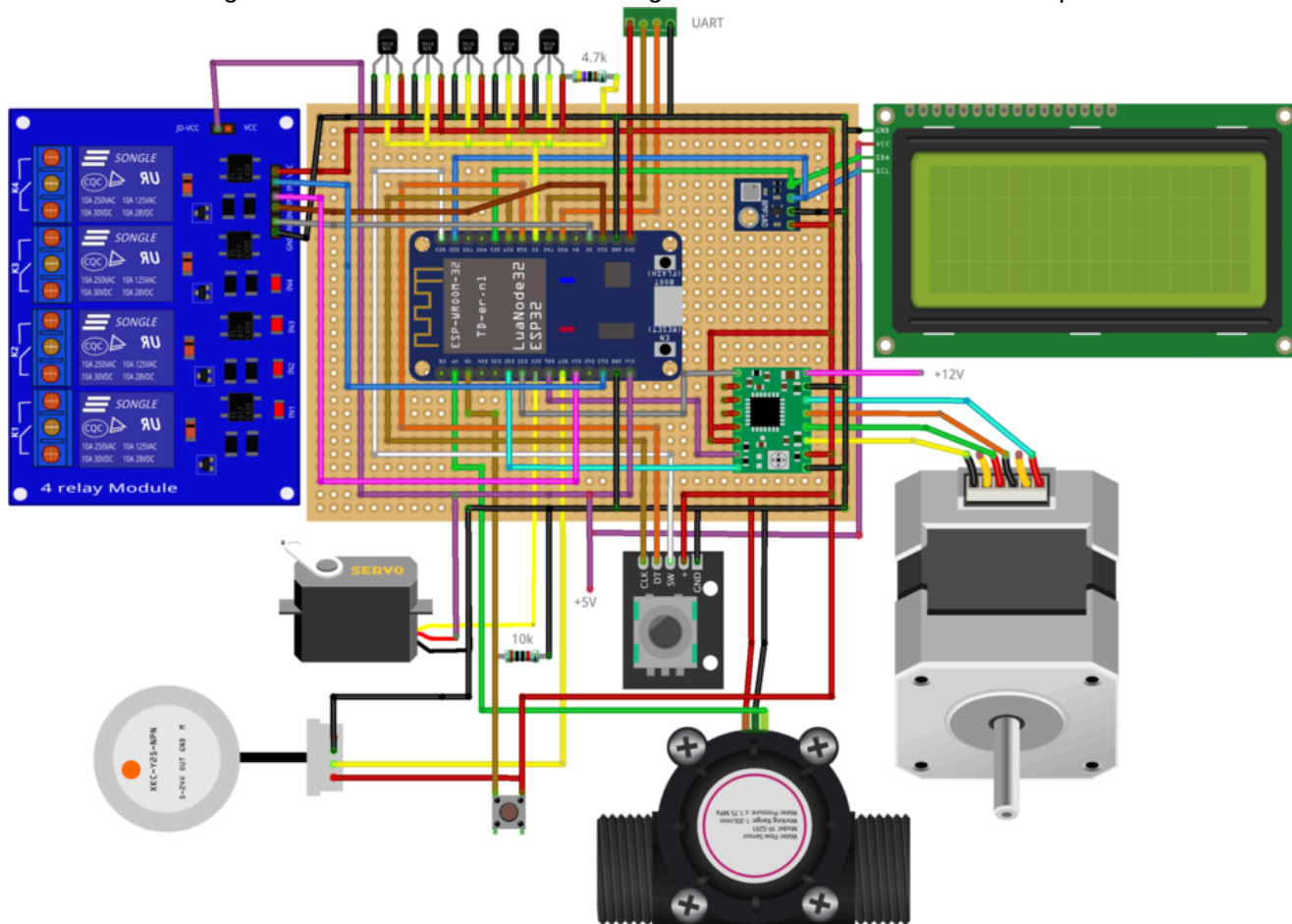
[View on the forum](#)

Recommendations:

- Having a printed circuit board makes assembly and the subsequent setup much easier. But most importantly, it spares you most of the problems caused by possible mistakes and poor connections between the ESP32 and the peripherals.
- If you want to start building, but nobody has shared a special printed circuit board and you do not want to order manufacturing yourself - using a shield is the most acceptable option.

Minimal configuration

The minimal configuration below describes monitoring without automatic control of the power section.



fritzing

A higher-resolution diagram is in the release archive, in the Fritzing scheme folder.

It is assumed that when the ESP32 was mounted on the printed board, the [atmospheric pressure sensor](#) and the [buzzer](#).

were installed as well. Next, connect the [temperature sensors](#). Since the computer is usually in one place and the column in another, the ESP32 can be powered from any phone charger.

Here Samovar becomes your observer. It shows messages on the monitor and sends them to your smartphone: boost finished, turn on the water, stabilization finished, temperatures exceeded, pause, and so on. You also see all the temperatures. If vapor breaks through, a message pops up; there are charts, and you can analyze the behavior of the column, and so on.

But stability and safety are up to you. It cannot control anything yet. It only informs.

Using Samovar with only temperature sensors and charger power can be called rather a test. It lets you make sure the ESP32 and the WiFi network work stably.

Further assembly will require placing the Samovar board in an enclosure and using a more serious power supply. If you plan to use a cooling pump, you will need one 12 volt 80-100 watt power supply.

You can use two power supplies: 40-50 watts to power the Samovar board and the stepper motor, and 60-70 for the cooling pump.

Next steps:

- A stabilized [voltage regulator](#) will let you trust Samovar with heating control almost completely: boost, stabilization, holding the temperature, and switching the heating off when the process ends.

- [Emergency shutdown](#) is strongly recommended right after connecting the regulator.
- [Cooling setup](#) fully automates holding the set temperature at the outlet, opens the water at the start and closes it at the end of the run.
- [Reflux level sensor](#) monitors column flooding. Together with the voltage regulator, Samovar automatically reduces heating and prevents liquid from being thrown out through the TCA (atmospheric vent tube). With this sensor the column can be kept in the pre-flooding mode automatically throughout the whole take-off.
- [Take-off pump](#)
- [Distribution by fractions](#)
- -

Buzzer

Signals events: the signal is a series of five short beeps. Turn it on or off in the web interface: "Settings" («Настройки») → "Main" («Основные») tab, checkbox **"Use buzzer for notifications"** («Использовать зуммер для нотификации») (with the checkbox cleared, the buzzer is completely disabled). The same tab has the checkbox **"Sound buzzer on program change"** («Включать зуммер при смене программы»).

It connects directly to an ESP32 pin (the BUZ1 connector on the printed board): D12 on ESP32 DevKit, GPIO0 on LILYGO T-Relay 4, GPIO39 on ESP32-S3.



Active buzzer DC 3V (red label); 5V (blue label); 12V (black label)

Use an active buzzer. A passive one will not work. Observe polarity when installing. The marking should be on the case, or the long leg is plus.

If you have a buzzer and want to find out whether it is active or not: Set the power supply to 3 volts, connect the minus, and touch the other contact with the plus a few times. If there is no sound, raise the voltage to 5 volts. If you hear a tone, a beep, it is active; if you hear only clicks when the plus touches, it is passive. The same goes for 12 volts.

A 3 volt buzzer is recommended, but a 5 volt one has been used too. The volume is quite decent.

There is no volume control for the buzzer. You can try adding a resistor in the power line, or simply cover the hole slightly with tape.

Atmospheric pressure sensors

Samovar 7.00: the sensor type is selected on the "Sensors" («Датчики») tab of the configurator; editing `Samovar_ini.h` by hand is only needed for a non-standard build.

Note.

Samovar works without this sensor, but it is better to install it, especially since the sensor costs very little.

The project can use BMP180, BMP280, BME280 and BME680 sensors. Some of these sensors can measure not only atmospheric pressure and temperature, but also humidity and even air quality. Samovar uses only the atmospheric pressure data, so it makes no difference which of these sensors you use.

The sensor is connected to the I2C bus together with the display: SDA D21 and SCL D22 on ESP32 DevKit, GPIO33/GPIO32 on LILYGO T-Relay 4, GPIO8/GPIO9 on ESP32-S3. The readings are converted to mm Hg and used to correct the boiling temperature.

The printed circuit board has pads for a 6-pin sensor, of which only 4 pins are connected.

So if you are buying a sensor rather than using one you already have, it is better to take the simple and reliable BMP180. You can read more about this sensor [in this review](#).

The sensor needs no additional setup.

Warning

*Because the sensors work over the I2C bus, some **BMP280** sensors may have a different address. In that case, uncomment the line `#define USE_BMP280_ALT` in the `Samovar_ini.h` file*

That is, it should look like this:

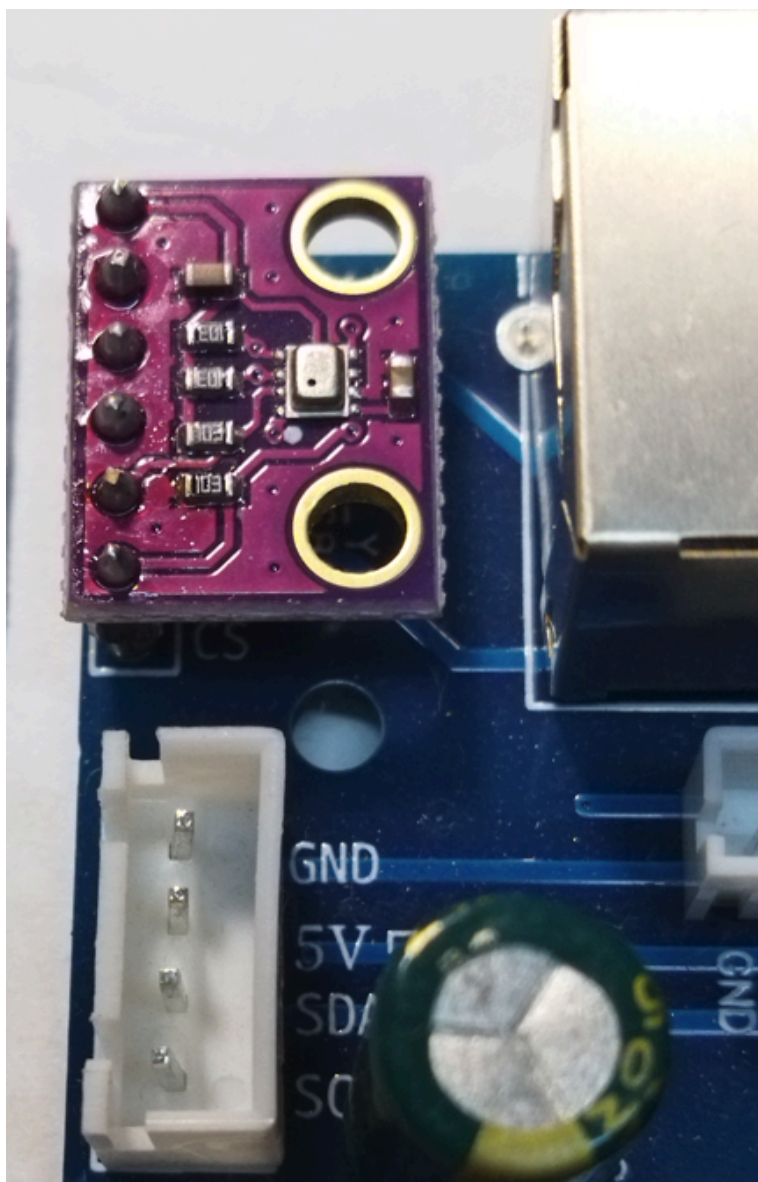
...

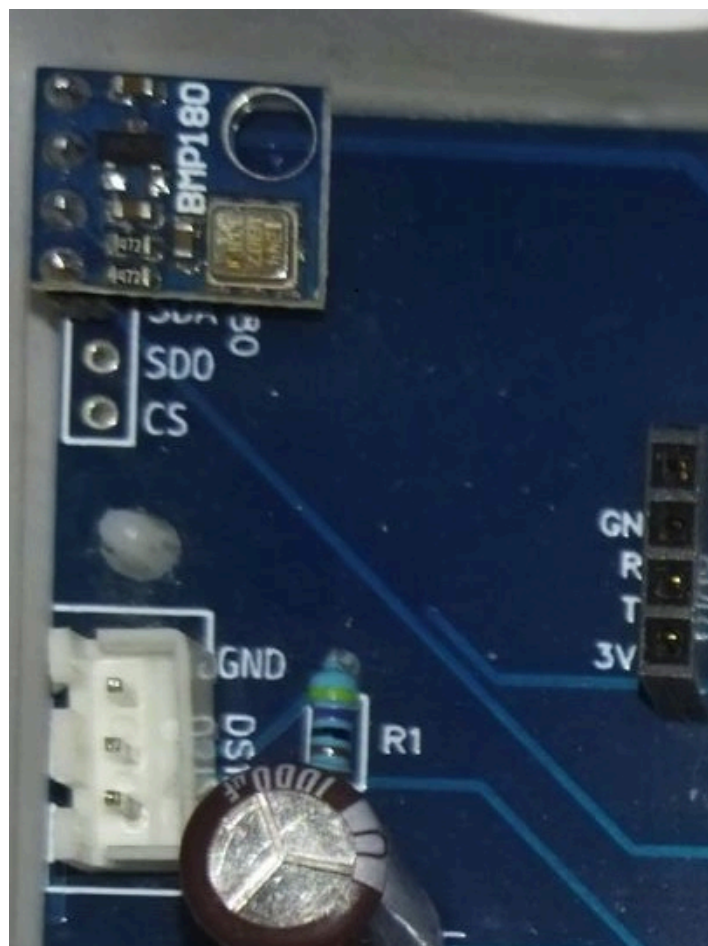
```
//#define USE_BMP280 //use the BMP280 pressure sensor
```

```
#define USE_BMP280_ALT //use the BMP280 pressure sensor with the alternative address.
```

*The standard Samovar 7.00 configuration has the BME680 enabled. In the configurator, the sensor is selected on the "Sensors" («Датчики») tab in the "Atmospheric pressure" («Атмосферное давление») field: "BMP180/BMP085", "BMP280", "BMP280, alternative address" («BMP280, альтернативный адрес»), "BME280", "BME680" or "Do not use" («Не использовать»). When building in the Arduino IDE, leave exactly one matching directive active in **Samovar_ini.h**; for a BMP280 with the alternative address, use `USE_BMP280_ALT`.*

It is best to solder the sensor board directly onto the printed circuit board.





Temperature sensors

Samovar uses **DS18B20** digital sensors. Connect them with three wires: 3.3 V power, GND and a shared data line with a 4.7 kΩ pull-up resistor to 3.3 V.

Data line: GPIO5 on ESP32 DevKit, GPIO12 on ESP32-S3 and GPIO27 on LILYGO T-Relay 4. All sensors are connected in parallel to one line; the firmware tells them apart by their unique addresses.

Five roles are supported: steam temperature, column section, cooling water or reflux, boiler and TCA (atmospheric vent tube). After installation, assign the detected addresses to the matching sensors in the Samovar settings ("Settings" («Настройки») → "Temperature" («Температура») tab, fields "Steam sensor address before the dephlegmator" («Адрес датчика пара перед дефлегматором»), "Column section sensor address" («Адрес датчика в царге»), "Water sensor address" («Адрес датчика воды»), "Boiler sensor address" («Адрес датчика в кубе»), "TCA sensor address" («Адрес датчика в ТСА»)) and check the readings before you start heating. The sensors are polled at 12-bit resolution. If a sensor was connected at the start of the process and then disappears, Samovar reports it regularly.

For long lines, use good-quality cable, reliable connections and a separate power supply for the sensors. Parasitic power over two wires is not recommended.

Voltage Regulators Used

Samovar 7.00: the regulator type is selected on the "Regulator" («Регулятор») tab of the configurator: KVIC, RMVK, SEM_AVR or "Do not use" («Не использовать»).

WARNING!

All regulators connected to Samovar must be physically isolated in terms of voltage. These can be relays, optocouplers, data bus isolators such as ADUM, etc.
Never connect the regulators directly, otherwise the mains voltage will reach you and Samovar.
The best thing that can happen in that case is that Samovar burns out.

Samovar supports several options. In Arduino IDE the choice is made with directives in **Samovar_ini.h**. In PlatformIO you can pick a ready-made environment: **Samovar** — KVIC, **Samovar_rmvk** — RMVK, **Samovar_sem** — SEM_AVR, **Samovar_no_power** — a build without a UART regulator. Only one option can be selected at a time.

Voltage regulators with UART protocol control

Algorithm for a regulator with UART.

Turn-on: Samovar turns on relay No. 1 and waits for the time **SAMOVAR_USE_POWER_START_TIME** (2000 ms by default; in the configurator it is "Regulator start delay, ms" («Задержка запуска регулятора, мс»)) on the "Regulator" tab; for SEM_AVR a further 3 s is allowed before that for the regulator to start), after which it sends the regulator the working command. If the regulator does not respond, heating is turned off with the message "Regulator command timeout" («Таймаут команды регулятора»). Relay No. 4 is controlled separately by the algorithm of the selected mode and does not have to turn on at the same time.

Turn-off: relays No. 1 and No. 4 are turned off immediately. After 700 ms the SLEEP command is sent to the regulator, and after another 200 ms the stop sequence ends.

1. Regulator [KVIC](#)

Fully compatible with Samovar. The adjustment step on the encoder is 0.5 volt. Stabilization is 0.1 volt. It can be shut down in an emergency by an independent sensor.

2. Regulator [RMVK](#) (only versions No. 5 and 6 work with Samovar)

All RMVK models have power isolation on the control contacts by default, and no modification is needed to work with Samovar.

The adjustment step on the regulator and from Samovar is 1 volt. Stabilization is 1 volt. Version No. 6 can be shut down in an emergency by an independent sensor. Through the RMVK, Samovar fully controls the boost, stabilization and emergency shutdown of heating.

Warning! The RMVK works only with whole values, that is, it has an adjustment step of 1 volt, which has its own particularities when Samovar works in [pre-flooding](#) mode.

3. Regulator [SEM_AVR](#)

Compatible with Samovar. It has one particularity: in the Samovar interface the power delivered to the load is set and displayed not in volts (as with the previous two regulators) but in watts.

For correct operation with Samovar you need to use the sketch Stab-avr.ino, which comes with Samovar and is located in the directory ...Samovar\Stab-avr

Voltage regulators without external control over protocols.

For example:

[RM2](#), [RM2M](#), [RM2-Pro](#) (hereafter simply RM) .

All RM models have power isolation on the control contacts by default, and no modification is needed to work with Samovar.

Control of these regulators comes down to two parameters:

- Turn the regulator itself on/off.
- Start/stop the "Boost" mode

The function is implemented with two relays whose contacts control the regulator. On Samovar's "Turn heating on" command,

the contacts of relay No. 1 turn the RM on. Relay No. 4 turns on the "Boost" mode only when the algorithm of the selected mode requires it.

Relay No. 4 opens its contacts and the RM turns off the boost mode. When the run is finished, Samovar turns the RM off with the contacts of relay No. 1.

Note:

- All the listed regulators work as standalone units and can be used with Samovar without a UART connection, following the principle and algorithm of the RM regulator.

Samovar and the voltage regulator show different power readings. How do I fix it?

By and large, the absolute power is not that important, but it is possible to get identical power readings on the regulator and on Samovar, and in essence there is only one method.

None of the regulators, Samovar included, measures power. They calculate it like calculators, from two values: the voltage currently applied to the heating element and the entered resistance value of that heating element.

The entered resistance is constant, and the voltage changes. The whole adjustment comes down to choosing the heating element resistance.

You can take the calculated resistance: the voltage squared divided by the power.

The voltage is taken from the nameplate of the heating element. Example for a 3 kW, 220 volt heating element: $220 \times 220 / 3000 = 16.1333333333$ Ohm.

Or measure it with a multimeter.

For the **RMVK** regulators and the Simple one based on Chinese modules (**Stab-avr**).

Warning! If we are not actually going to heat the boiler, we disconnect the heating element.

- In the **RMVK** you can enter the power and voltage values, and the resistance is calculated automatically. Or enter the resistance and the power is calculated. [Video on how to set it up](#). Watch it, do it, and do not forget to switch the regulator to the display in watts.

- In the **Stab-avr**, set the actual mains voltage at the moment with the potentiometer and enter the power value of your heating element.

Warning! The heating element power for the Stab-avr must be entered for a mains voltage of 230 volts. If the heater manufacturer gave the power for 220 volts in the datasheet, you need to recalculate it for 230 volts.

Start Samovar — heating. There is no point in adjusting anything in the Boost mode — the voltage will keep changing all the time, and the power with it. From Samovar, give the command to set some specific voltage on the regulator (say 170 volts for the RMVK, or 2000 watts for the Stab-avr). The regulators will switch to stabilization mode.

Compare the power values displayed in Samovar and on the regulator.

Now there are two ways — either change the resistance value in Samovar (web interface: MAIN page - Settings («ГЛАВНАЯ - Настройка») - Basic («Основные») - the Other («Прочие») tab), which is much more convenient. Or change the power value entered in the regulator. The goal is to get values on the regulator and in Samovar as close as possible.

*The **Kvic** regulator .*

It has no power readout. So only Samovar is adjusted, to the power you want to see...

That is the first option

The second is to use a chain: a separate [voltage and current meter](#) — regulator — Samovar.



Вольтметр Амперметр переменного тока 50-300 в 20 А, измеритель мощности KWS, цифровой электронный измеритель мощности, ваттметр переменного тока, Электрический измеритель с функцией сброса 0-а

★★★★★ 4,8 265 отзывов 445 продано

[Характеристики](#) [Описание](#)

Цвет: 100A-Close



Название бренда	Interest Maker
Номер модели	AC 50-300V 20A 100A Voltmeter Ammeter KWS Power Energy Meter LED Digi
Происхождение	Китай
Сертификация	Евротест (CE), RoHS
Размеры	84.6*44*22.3mm
Товары для ремонта	ELECTRICAL

This is convenient if different heating elements are used from time to time and/or the heating elements of the heating element assembly itself are switched.

The voltage and current meter really measures the current drawn and shows the actual power on its display. So if you want Samovar to show this power, just change the heater resistance value in Samovar on the fly.

See also:

- Help with choosing: [overview of regulators](#)
- Besides controlling regulators, Samovar can [control an additional boost heating element](#)
- How to connect the regulator to Samovar without wires [over a radio channel](#)

Overview of regulators

This overview will help you choose a regulator to work with Samovar.

All the regulators presented below are output voltage stabilizers. They deliver a stabilized voltage at the output regardless of mains surges. They are not step-up devices. Stabilization happens only if the output voltage is lower than the input voltage. They can work as standalone devices.

Attention.

In this overview, the positive and negative features of the regulators are considered solely in terms of working with Samovar.

KVIC regulator

A stabilizer based on an Arduino microcontroller. You can build it yourself if you have the components and skills in assembling and/or repairing electronic devices. Manual control - an encoder. Display: 0.96-inch LCD

A ready-made device can be purchased. The form factor, equipment, warranty and other terms are agreed individually with the manufacturer.

The source code is written in assembly language, is open and [available](#) for you to modify yourself.

Pros:

- Fast start, output voltage stabilization. Down to 0.1 volt
- Manual control with an encoder with a button. Adjustment step 0.5 volt
- Easy-to-read display.
- Voltage stabilization indication.
- The ability to manually preset the stabilization voltage that the regulator will switch to after the "Stop Boost" («Стоп разгон») command is executed.
- Control via UART or I2C
- Repairability and the ability to update the program as new versions come out. The ability to install alternative firmware with a port speed of 9600 baud (38400 baud by default)
- Control of the boost heating element through a separate power relay.
- A connector for the external "Stop Boost" («Стоп разгон») command
- A connector for the external "Emergency" («Авария») command
- Smooth application of voltage to the load at start and smooth removal when switching off.

Cons:

- **All elements of the regulator** are at high voltage.
- To replace the program, you may have to remove the microcontroller from the regulator.
- The preset stabilization voltage is not saved in the device memory on power-off or reboot.
- Repair and flashing require special skills.

Buy "made to order" or build it yourself: [support forum thread](#).

RMV-k regulator



A stabilizer built in a 2DIN enclosure with DIN-rail mounting. Manual control and setup — with buttons. Information display: a three-digit indicator. It cannot be built yourself.

The stabilizer is on sale in 6 versions and 5 configurations. (at the time of writing the overview).

Attention! Only versions 5 and 6 are suitable for working with Samovar over UART.

Pros:

- Service life of the product — 10 years.
- 6-month warranty
- Voltage holding stability 1 Volt. Adjustment step 1 volt
- Memory for 10 preset values
- Heating element resistance is saved in memory for power calculation

- Voltmeter calibration
- It can be switched to display readings in watts.
- Control from Samovar over UART (versions 5 and 6 only).
- A connector for the external "Emergency" («Авария») command (present on version 3 and higher).
- A connector for the external "Stop Boost" («Стоп разгон») command (present on version 4 and higher).

Cons:

- In case of repair, special skills and equipment are required.
- **All elements of the regulator** are at high voltage.
- The firmware cannot be updated
- If the regulator is used as a standalone unit, an additional power source is needed to use the "Stop Boost" («Стоп разгон») and "Emergency" («Авария») functions (a 9 V "Krona" battery is enough)
- Complex manual programming and entering values into memory.
- Manual control quirks (for example, to lower the voltage you first have to raise it by 1 volt, and only then lower it).

You can buy the regulator from the developer through a [support forum thread](#) (the regulator only, without the triac, etc.)

Buy it fully assembled in a box, ready to connect the heating element and work, in the [online store](#) (there you can also buy a kit of components for self-assembly).

Stab-avr regulator

A simple stabilized heating element power regulator made from Chinese modules

A stabilizer based on an Arduino microcontroller. Control with buttons. Display: 0.96-inch LCD. You can build it yourself with minimal soldering skills. Not on sale.

The source code is written in C++, is open and can be modified yourself with the Arduino IDE.

Pros:

- Simple assembly from ready-made modules.
- Fully repairable.
- Control from Samovar over UART
- The ability to update the program, open source code.

- All elements of the regulator are at safe voltage, except the power ones.
- It can be connected to Samovar without voltage isolation (the voltage on the Arduino NANO and the ESP32 must be matched).
- External "Stop Boost" («Стоп разгон») connector
- External "Emergency" («Авария») connector
- Voltmeter calibration
- The ability to preset and store in memory the output power value that the regulator will switch to after the "Stop Boost" («Стоп разгон») command is executed.
- The nominal heating element power is saved in memory.

Cons:

- A relatively long start of the original firmware when working with Samovar. It is recommended to use the modified firmware supplied with the Samovar sketch.

How to build it yourself: [support forum thread](#).

RM regulator (RM2M, RM2M-Pro)



A stabilizer built in a 3DIN enclosure with DIN-rail mounting. Manual control and setup — with buttons. Information display: a three-digit indicator. It cannot be built yourself.

The stabilizer is on sale from various manufacturers. Accordingly, the equipment and warranty differ for each. You can buy either the stabilizer alone (without the triac etc.) or a complete device ready for work.

Pros:

- Stability of holding the set voltage: plus or minus 1 Volt.
- Adjustment step 1 volt
- Indication of the power consumed by the load: from 0 to 9.99 kW.
- Memory for 10 voltage presets.
- Built-in voltage-time timer: from 0 to 999 minutes.
- Needs no additional power source when using the "Stop Boost" («Стоп разгон») and "Emergency" («Авария») functions

Cons:

- There is no control channel.
- Complex programming and entering values into memory.

Conclusions.

For use with Samovar, the first three regulators are recommended. Provided that UART control is used, Samovar will control the regulator completely: from starting the boost, to changing the voltage at any moment and switching heating off.

The ability of emergency shutdown from a third-party device can be used as an option. For example, when using Samovar with a PVK (steam-jacketed boiler), to switch heating off if the pressure in the steam-water jacket is exceeded. An electrical contact pressure gauge can be used as the sensor.

The **KVIC** regulator. Convenient manual control; the adjustment step of 0.5 volt lets you bring the column to the pre-flooding mode quite precisely and hold it in that state for a long time. But only an experienced electronics engineer can build it.

The **RMV-k** regulator.

A stably working device with quite a wide choice of configurations. Reliable; as far as is known, there are practically no complaints. When connected to Samovar over UART it is convenient and simple to control. You can buy a box completely ready for work together with the "Stop Boost" («Стоп Разгон») and "Emergency" («Авария») sensors. The sensors can be used when the regulator works standalone and are a KSD9700 bimetallic thermostat. When connected to Samovar, the "Stop Boost" sensor is not used, since the

connector is used for the data exchange line - accordingly Samovar controls the "Stop Boost" mode itself. The "Emergency" sensor can be used as a standalone option. The sensors are not suitable for use in Samovar, but can be used as components when building an additional [line of protection](#).

The **Stab-avr** regulator is not inferior to it either. The ability to build and repair it yourself makes it very attractive. A feature of this regulator is that the power setting is displayed in watts (and not in volts, as with the regulators reviewed above).

The adjustment step is 1/200 of the heating element rating. Due to the way power is set, the actual value may differ slightly from the one set by Samovar (for example, 1803 W instead of 1800). This is absolutely not critical.

All the advantages of the **RM** (RM2M, RM2M-Pro) regulators are cancelled out by one big drawback. It is the lack of UART communication. And therefore the impossibility of setting the voltage on the regulator "from outside". The stabilization voltage can only be set manually.

When working with the RM, Samovar can only switch the regulator on/off and switch Boost on/off. Because this regulator provides no other ways of control from third-party devices.

Recommendations.

If you have even a little experience with a soldering iron, it makes sense to build the regulator **Stab-avr**.

If you have serious doubts about your own abilities or no time, it makes sense to buy the regulator **RMV-k** fully assembled.

How to connect a regulator to Samovar

Wired connection.
Kvic and RMVK regulators

The KVIC regulator uses an ADUM1201 for voltage isolation. Samovar talks to it at 38400 baud, or at 9600 baud when **KVIC_USE_9600** is enabled.

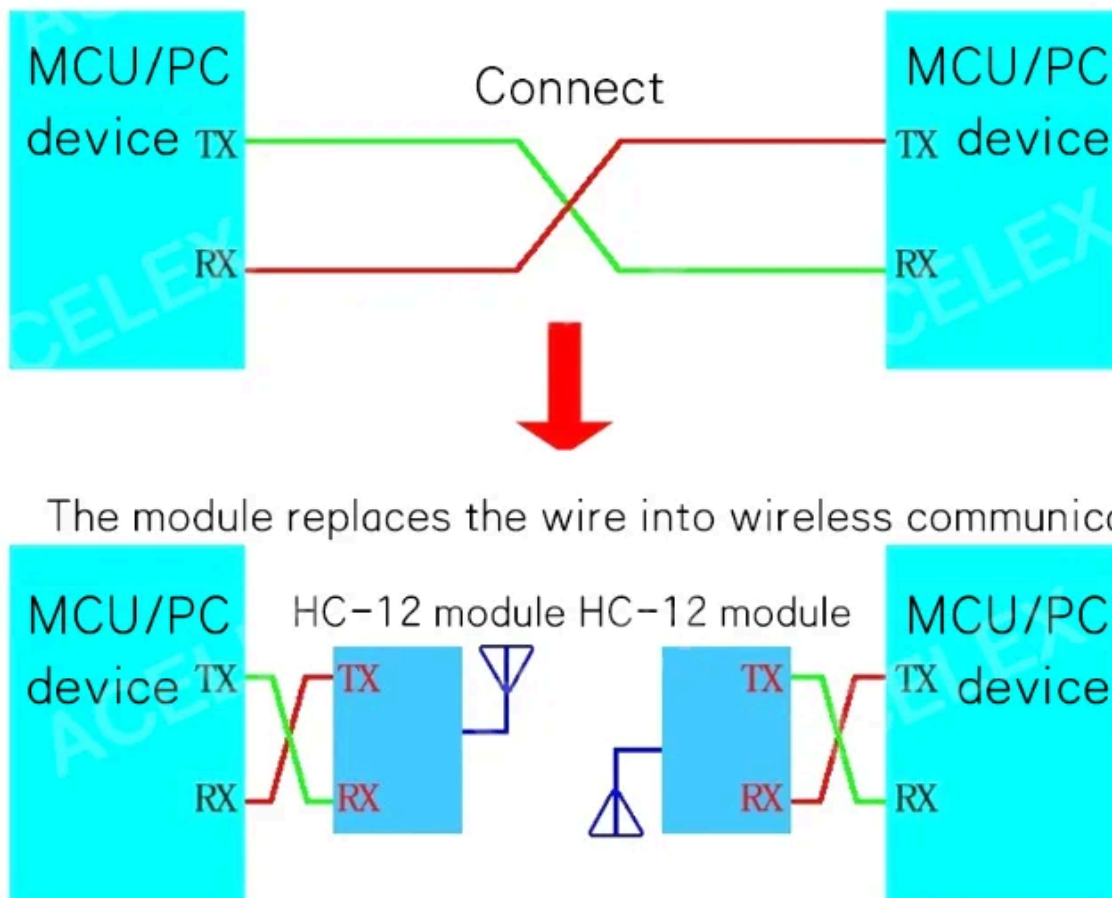
The RMVK regulator uses PC817 optocouplers for isolation. The port speed is 9600 baud.

These regulators are connected to Samovar in the same way. The Rx of one device must be connected to the Tx of the other.

On the Samovar side, UART2 is used (on the printed circuit board it is the UART connector): RX D16 and TX D17 on ESP32 DevKit, RX GPIO25 and TX GPIO4 on LILYGO T-Relay 4, RX GPIO18 and TX GPIO17 on ESP32-S3. The SEM_AVR regulator (Stab-avr) is also connected to this port, at 9600 baud.

How to connect a regulator to Samovar wirelessly

A Samovar with a regulator controlled over UART (COM port) can be connected without a cable using two RF modules (radio-frequency modules) HC-12. This is very simple; just connect them as shown in the diagram:



The ADUM is not needed here, because the modules are designed to work at voltages from 3.1 V to 5 V, so they can be connected to both the ESP32 (Samovar) and an Arduino Nano.

The manufacturer states that the transmission range of this module is up to 1 km (in open terrain and with an external antenna), which is more than enough for our purpose. The module comes with a default speed setting of 9600 bps. If you use a STAB-AVR or RMVK regulator controlled over UART, just connect one module to Samovar and the other to the regulator as shown in the diagram, and everything should work. If you use a Kvic regulator, you must configure both modules. This is not difficult, but you will need to connect them to a computer to change the speed at which the modules exchange data with each other.

Both HC-12 modules must be set to the same speed as Samovar and the regulator. For the RMVK and SEM_AVR it is 9600 baud. For KVIC use 38400 baud, or 9600 baud if **KVIC_USE_9600** is enabled in the build.

The module has many options, but we are interested in only one: changing the module's port speed. To put the module into setup mode, connect the SET pin to GND. Then power the module and connect it to a computer through a UART-USB adapter (an Arduino Nano can be used for this). If you chose 38400 baud, send the command **AT+b38400** to the detected COM port and wait for the reply **OK+b38400**. At 9600 baud you do not need to change the HC-12 factory setting.

Disconnect SET from GND, remove the power, connect the modules to Samovar and the regulator, and everything should work.

Warning!

The Kvic regulator has no galvanic isolation from the mains voltage; even the HC-12 antenna will be at mains potential! So the module must be placed inside the regulator's enclosure or otherwise reliably insulated.

Heating element

You can use any heating elements that the voltage regulator you have can control. Here we look at the most common type of heating element.

In Russian, the abbreviation ТЭН (heating element) stands for "tubular electric heater". So the assembly contains as many heaters as it has tubes. If you divide the total power of the heating element by the number of tubes, you get the power of one tube. All tubes have the same power.



As a rule, a 3 kW model is chosen. Each tube then has a power of 1 kW. For 1.5" columns this is a good choice, because to run in pre-flooding mode a column 1.5 meters high needs 1.2-1.4 kW of power. That leaves a power reserve.

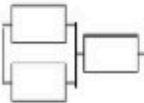
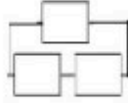
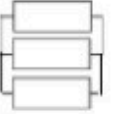
So you can use a combined connection:

- Connect two tubes through the regulator; this gives 2 kW. During rectification the regulator keeps the output voltage within 170-180 volts. That is quite enough, even when the mains voltage "sags".
- Connect the remaining tube through a solid-state relay, and control that relay through the contacts of relay No. 4. This will be the boost heating element.

For 2" columns, a 3 kW heating element is not enough with this wiring scheme. A 4.5 kW element is just right.

By switching the tube connections you can use much more powerful heating elements, and they will run in a gentler mode. Of course, follow the principle of reasonable sufficiency.

Below is a table of possible heater connections in the assembly.

	1	2	3	4	5	6	7
ТЭНБ							
3 kw	0,33	0,5	0,63	1	1,5	2	3
4,5kw	0,5	0,75	1	1,5	2,25	3	4,5
6kw	0,67	1	1,33	2	3	4	6
9kw	1	1,5	2	3	4,5	6	9
12kw	1,3	2	2,67	4	6	8	12

How to read the table, using a 4.5 kW heating element as an example

In the left column, select the power - 4.5 kW. The top row, by number, shows the heater connection scheme.

For example, number 1 is a series connection of all three heaters. At the intersection of the row and column is the power the assembly delivers when 220 volts are applied to it = 0.5 kW, or just 500 watts. Connection No. 6 gives 3 kW on 2 heaters and leaves one heater unused - so it can be used as the boost heating element.

Using a Boost Heating Element

Samovar can control an additional boost heating element in the "[Rectification](#)" («Ректификация»), "[Distillation](#)" («Дистилляция») and "[Beer](#)" («Пиво») modes

To use this function, connect either a contactor or a solid-state relay, with a load current high enough to drive the second heater, to the normally open contacts of relay No. 4.

Boost heating element operation in "Rectification" mode

To use the additional boost heating element you do not need to change anything in the sketch. This function is enabled by default.

The threshold temperatures mentioned below are set in the configurator on the "Temperatures" («Температуры») tab ("Boost exit temperature, °C" («Температура перехода из разгона, °C»), "Cooling opening temperature, °C" («Температура открытия охлаждения, °C»), "Boost heating element threshold, °C" («Порог разгонного ТЭНа, °C»)); when building in the Arduino IDE, they are set by the same directives in [Samovar_ini.h](#).

The logic is similar to that of the [PM](#) in "Boost" («Разгон») mode:

On the "Turn on heating" («Включить нагрев») command, in addition to Samovar turning on the main heating element, relay No. 4 turns on the contactor or solid-state relay, which in turn supply voltage to the boost heating element. When the steam temperature reaches the value set in the settings (39° by default):

```
//Значение температуры датчика пара, при котором колонна перейдет из режима  
разгона в рабочий режим  
#define CHANGE_POWER_MODE_STEAM_TEMP 39
```

Relay No. 4 opens its contacts and turns off the boost heater.

With a UART-controlled regulator (KVIC, RMV-K, SEM_AVR) there are two more conditions: relay No. 4 is on only while the "Boost" («Разгон») status is active and the boiler temperature is not above **OPEN_VALVE_TANK_TEMP** (77° by default, "Cooling opening temperature"). As soon as the boiler warms up above this value, the boost heating element is turned off, even if the steam has not yet reached 39°. If the boiler sensor is not connected, Samovar does not control the boost heating element. Without a UART regulator, relay No. 4 turns on together with relay No. 1 when heating starts and turns off when boost is finished.

If the "Turn on heating" command has not been given, or the sensor temperature is above the value set in the sketch, boost will not turn on.

Operation in "Distillation" mode. The boost output turns on when heating starts. After boiling begins, it turns off if the "**Use boost heating element during boiling**" («Использовать разгонный тэн при

кипении») checkbox is cleared in the settings ("Settings" («Настройки») → "Beer" («Пиво») tab); if the checkbox is set, the output stays on until heating stops normally.

Boost heating element operation in "Beer" mode

The heating element is connected the same way as described above.

But the algorithm is different

The boost heating element during mashing is governed by the "Boost heating element threshold, °C" parameter (the "Temperatures" tab of the configurator); in [Samovar_ini.h](#) it is this line:

```
#define ACCELERATION_HEATER_DELTA 4
```

```
//Если разница между целевой температурой и текущей температурой (для режимов  
пиво и су-вид) больше заданной, то при использовании регулятора, управляемого по  
UART дополнительно Реле № 4, будет включать разгонный тэн, при разнице меньше,  
разгонный тэн отключится
```

Enabled by default, 4 is the value in degrees C°. Works only with a UART-controlled regulator. Without such a regulator, relay No. 4 turns on together with the main heating whenever the temperature is below the target, and turns off together with it.

When turned on, Samovar commands the regulator to apply full power to the main heating element. If the boiler temperature is 4° below the one set in the mashing program line, Samovar turns on relay No. 4, which in turn commands the boost heating element to turn on. If the difference is less than 4°, the boost heating element does not turn on and heating is done by the main heater only.

When the next temperature rest is completed, or the operator gives the "Next program" («Следующая программа») command, Samovar compares the boiler temperature with the one set in the next program line. If the difference is greater than 4°, the boost heating element is engaged.

If the command "Turn on heating" has not been given or the delta is less than the set one, boost does not turn on.

While the wort is boiling, the boost heating element is on if the "Use boost heating element during boiling" checkbox is set, otherwise it is off.

Samovar 7.00: in "Cheesemaking" («Сыроварение») mode, relay No. 4 is used by the brine drain valve. Connect to it either a cheese valve or a boost heating element for the other modes, but not both devices at the same time. There is no separate boost heating element logic in the "BK" («БК»), "NBK" («НБК») and "Sous-vide" («Су-вид») modes: with a UART regulator, relay No. 4 is not turned on in them.

Contactor. RCD. Emergency shutdown

Additional protection for abnormal situations.

Recommended to build.

The proposed protection works independently and switches the equipment off in critical abnormal situations.

The main danger is a breakthrough of alcohol vapors. If for some reason any of the following happens:

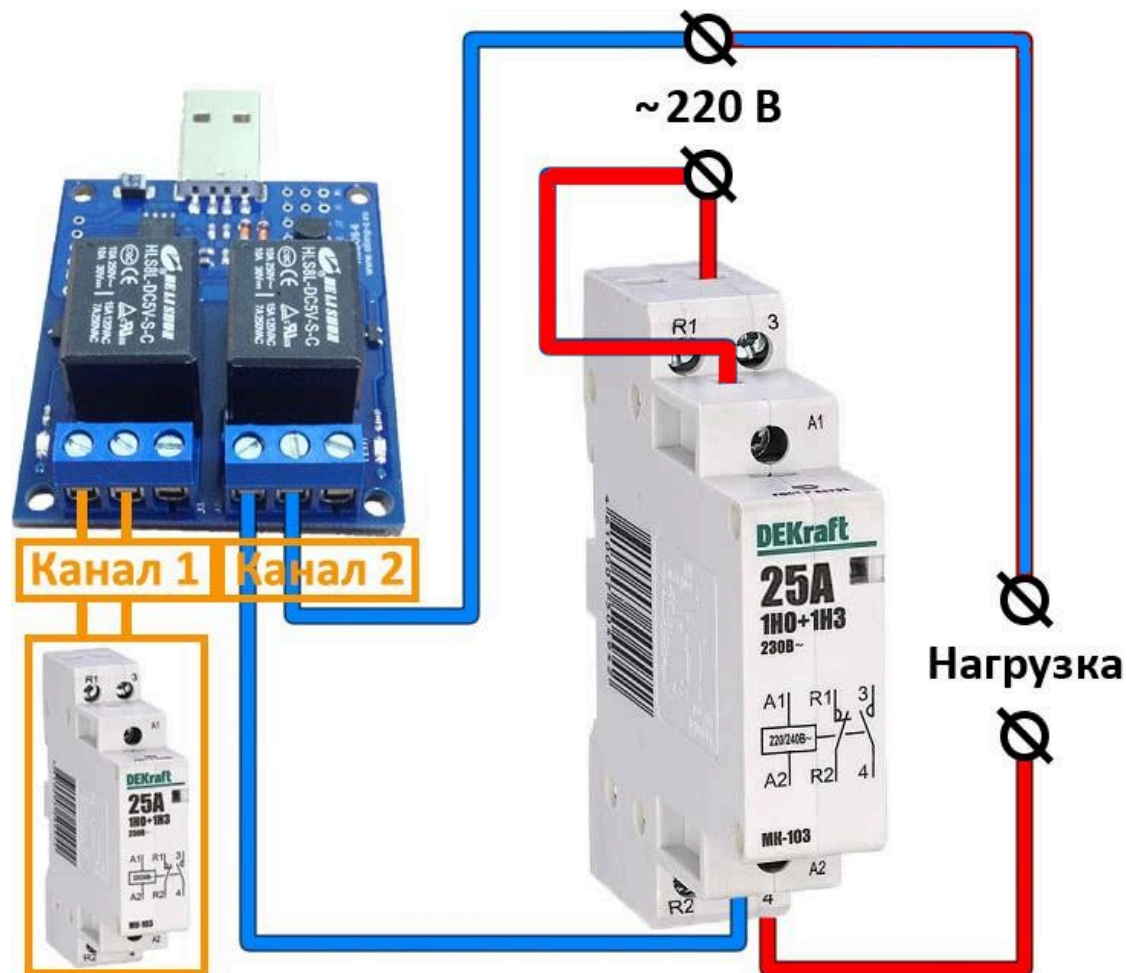
- The triac fails (short circuit).
- The voltage regulator hangs or works incorrectly.
- The automation fails.
- The 12 V power supply fails.
- There is no cooling.

Then you must switch off the heating and stop the vapor release. The monitoring point is the TCA (atmospheric vent tube).

Option 1

This applies if a contactor is used together with Samovar.

Connecting the contactor to the [relay module](#)



You will need a bimetallic thermostat KSD9700 NC (normally closed) with a trip temperature of 55-70°C. Preferably with an insulated body: ceramic or plastic.

How to build it:

The negative voltage from the 12 V power supply is fed to the Samovar board not directly but through the KSD9700, which is attached to the TCA. It is most convenient to do this through a two-pin connector on the Samovar enclosure and a wire of the required length.

How it works.

If an emergency occurs that Samovar cannot handle (lost communication with the regulator, a failed triac, etc.), vapor starts to break through the TCA. The thermostat heats up and breaks the circuit. Samovar switches off, the control relay releases the contactor, and the contactor in turn cuts the heating. The vapor release stops. If a water valve is used, it closes.

After some time the TCA cools down, the thermostat closes its contacts, and Samovar switches on. But all the other devices will not switch on without your

And of course, when you connect Samovar to the column, if you forget to plug in the connector, Samovar will not switch on, because there is no voltage, which is equivalent to a failed power supply.

The diagram illustrates the electrical connections for a digital voltmeter using a relay. The main components and their connections are as follows:

- Power Source:** A 220V AC supply (~220V) is connected to the circuit.
- Relay:** The relay has two coils, A1 and A2, and two contacts, 8 and 9.
 - Coil A1 is connected to the 220V supply through switch 5.
 - Coil A2 is connected to the 220V supply through switch 7.
 - Contact 8 is connected to the 220V supply through switch 6.
 - Contact 9 is connected to the 220V supply through switch 8.
- Digital Display Module:** The module has four terminals (1, 2, 3, 4) for AC input and four terminals (GND, TX, RX, Vcc) for serial communication.
 - Terminal 1 is connected to the 220V supply.
 - Terminal 2 is connected to the 220V supply.
 - Terminal 3 is connected to the 220V supply.
 - Terminal 4 is connected to the 220V supply.
 - The GND, TX, RX, and Vcc terminals are connected to a computer (represented by a monitor and tower icon).
- Load:** The load (Нагрузка) is connected to the 220V supply through the relay contacts 8 and 9.

45 / 103

The contactor is switched on by the "Turn on heating" («Включить нагрев») command from Samovar.

Accordingly, the regulator needs some time to start, and only then does it receive the "Boost" («Разгон») command from Samovar and apply voltage to the load. This delay is set by the **SAMOVAR_USE_POWER_START_TIME** parameter (2000 ms by default; in the configurator it is "Regulator start delay, ms" («Задержка запуска регулятора, мс») on the "Regulator" («Регулятор») tab), and for SEM_AVR another 3 s are added to it. If the regulator does not respond within this time, Samovar switches the heating off with the message "Regulator command timeout" («Таймаут команды регулятора»).

Weigh all the pros and cons before installing the contactor at this point.

Point 6. Strongly not recommended. A contactor at this point essentially just switches the regulator off, and that is all. The regulator itself consumes next to nothing, and if you need to switch off only the regulator, you can fit a low-power switch.

With this connection, if the regulator hangs, the triac fails, and so on, the heating element will keep heating.

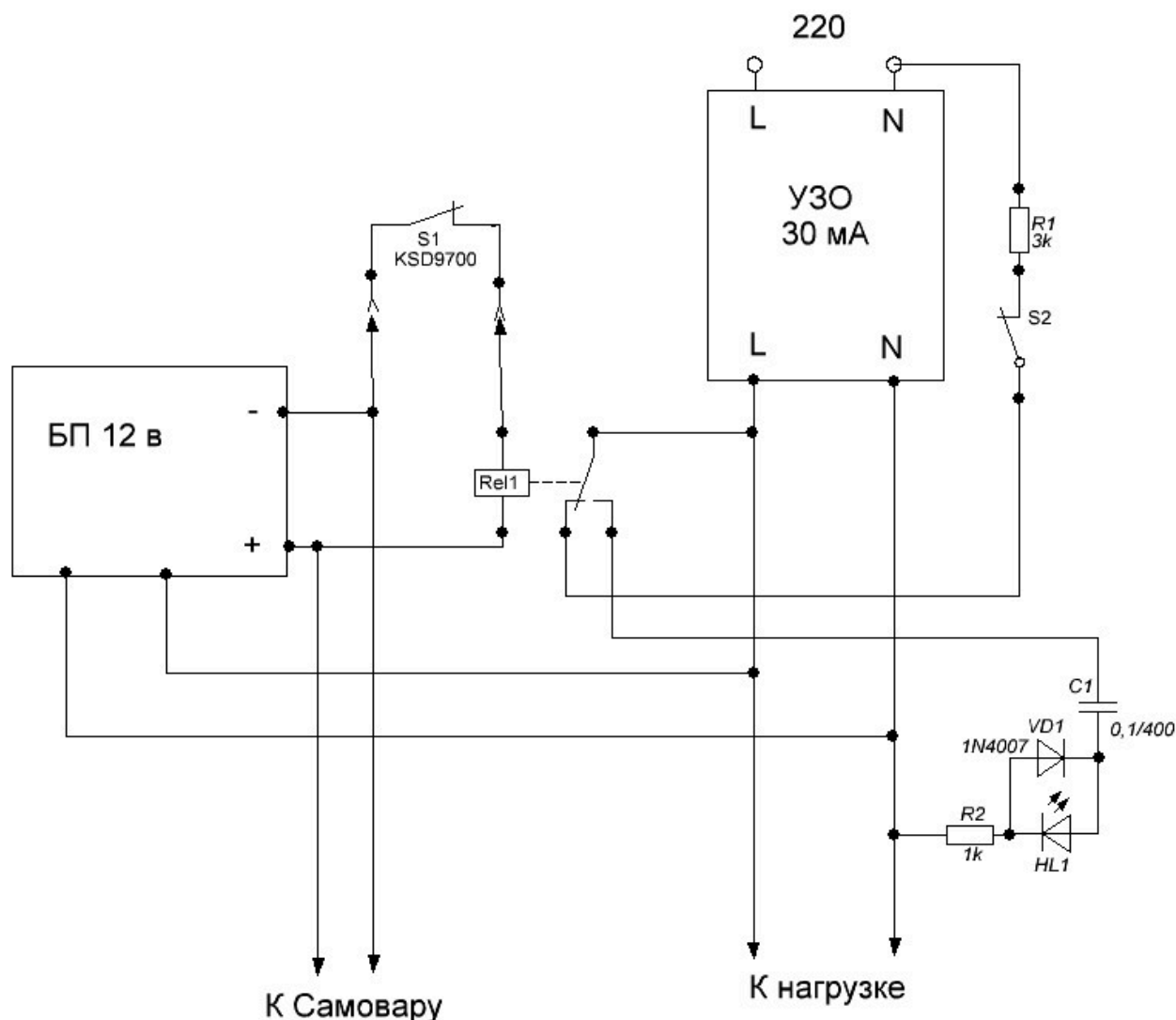
Points 7 and 8. When the main power is applied, the regulator switches on. It starts for as long as it needs. You can adjust it, set voltages, and so on. But nothing goes to the load, because the contactor breaks the power circuit to the triac. Only when Samovar gives the "Turn on heating" («Включить нагрев») command does the contactor close and voltage go through the triac to the load.

Point 9 - recommended for installation. With the contactor installed here and the triac closed, the heating element is completely de-energized.

Option 2

Instead of a contactor you can use an RCD (residual-current device), which also adds safety against voltage leakage to the unit's body.

Sensor S1 (normally closed KSD9700) is installed on the TCA.



How it works.

To switch everything on, Samovar included, press and hold the normally closed button S2. The switching on is done by the RCD itself. After voltage is applied, the 12 V power supply starts and powers the coil of relay Rel1. The relay contacts (NO) close and LED HL1 lights up. Button S1 can now be released. The guard is ready. If the LED is not lit and you release the button, the RCD trips.

Let us consider the possible situations:

- The operator forgot to connect sensor S1 – the relay does not switch on – the RCD trips
- The triac fails – vapor through the TCA – sensor S1 breaks the circuit – the relay is de-energized – the RCD trips
- Insufficient cooling – vapor through the TCA – sensor S1 breaks the circuit – the relay is de-energized – the RCD trips

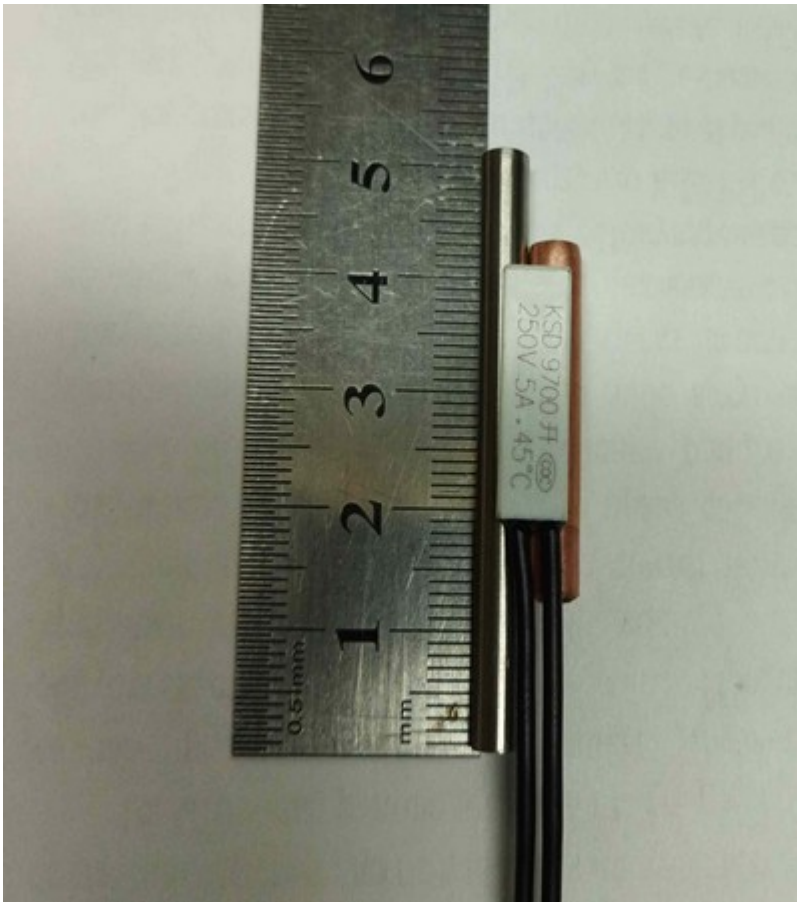
- The 12 V power supply fails – the relay is de-energized – the RCD trips
- The automation and/or the regulator hangs – vapor through the TCA – sensor S1 breaks the circuit – the relay is de-energized – the RCD trips.
- A short power outage – when the power returns, the RCD trips.

Example of building the monitoring unit

The TCA monitoring line unit.

Materials: stainless steel tube 6 mm in diameter, copper tube 6.1 mm in diameter (assuming the 18B20 sensor has no sleeve; if it has a sleeve, the tube is not required), a KSD9700 sensor, heat-shrink tubing, silicone tubing.

Place the stainless steel tube next to the copper one (or next to the temperature sensor sleeve). Choose the tube length so that the stainless one is a couple of centimeters longer. Lay the KSD9700 sensor on top and wrap the whole assembly with heat-shrink tubing. Insert the 18B20 sensor, which is assigned in Samovar as the TCA sensor, into the copper tube. Just in case, seal one end of the copper tube with any sealant to keep liquid out





For rectification, connect the stainless steel tube into the break of the line from the dephlegmator TCA to the vapor condensate drain container. From experience, it is better about 30 cm from the TCA.

When distilling wash and running a wash column, connect the stainless steel tube directly to the product outlet spout. That is, the product flows through the tube and on into the receiving container. In addition, you also see the product temperature at the outlet as the TCA temperature. Since you can adjust the temperature of the water leaving the condenser in real time, you can tune the pump to its optimal output.



You can skip the copper tube altogether and press a sleeveless sensor directly against the stainless steel tube, but then you will have to take care of sealing the sensor, and the thermal contact area for the KSD9700 will also be smaller.

For correct operation, the temperature at which Samovar switches the heating off based on the TCA sensor ("TCA emergency temperature, °C" («Аварийная температура TCA, °C») on the "Temperatures" («Температуры») tab of the configurator; in [Samovar ini.h](#) it is the line)

//Maximum TCA temperature at which the power is switched off

#define MAX_ACP_TEMP 75

must be set 5-10° below the KSD9700 trip temperature. The default value of 75° is above the thermostat threshold of 55-70°, so it must be lowered.

Then, when vapor breaks through, Samovar acts first and issues the shutdown command, and if it is not carried out, the contactor or RCD trips.

Cooling

Samovar 7.00: the cooling opening temperature and the pump parameters are set on the "Temperatures" («Температуры») and "Pumps" («Насосы») tabs of the configurator.

In rectification and distillation modes, Samovar maintains the set cooling water temperature.

To organize cooling, Samovar uses:

1. Relay output No. 3: GPIO14 on ESP32 DevKit and GPIO41 on ESP32-S3, which controls the valve that supplies water from the mains. Or any other equipment that must switch on at the "Open the water supply" («Откройте подачу воды») command.
2. The pump control output: GPIO4 on ESP32 DevKit and GPIO11 on ESP32-S3; a PWM modulator works on it, automatically changing the speed of the water supply pump. This output can also control a valve that increases the water flow, provided the pump is not used.

The algorithm (using Rectification as an example, with the original sketch). When the boiler temperature **OPEN_VALVE_TANK_TEMP** is reached (77 °C by default), the valve opens and the pump starts. For the first ten regulation cycles the pump gets the starting level **PWM_START_VALUE** (40% by default), then the regulator maintains the set water temperature, not dropping below **PWM_LOW_VALUE**.

Cooling is set up in two places:

1. In the configurator ("Temperatures" («Температуры») and "Pumps" («Насосы») tabs) or in the **Samovar_ini.h** file — these are the same parameters, the configurator writes them to this file. The lines:
//The boiler sensor temperature at which the water supply valve opens and the water supply pump switches on

```
#define OPEN_VALVE_TANK_TEMP 77
```

At a boiler temperature of 77° the command to open the water supply valve will be given, the PWM control signal to switch on the pump will be sent at the same time, and a message will be sent to the operator that the water must be opened. The message is sent in any case, even if neither a pump nor a valve is used.

```
#define DELTA_T_CLOSE_VALVE 20
```

//The difference between the set cooling water T and the water T down to which the pump will run after the distillation process ends in distillation and rectification modes. The value is in degrees C°, it can be changed. Disabling it is strongly discouraged. (for details see [here](#))

//The lower value (in percent) for the PWM regulator of the pump to work. Pick a value that ensures water flow

`#define PWM_LOW_VALUE 10`

As is well known, a motor needs more voltage to start than to keep running. This line sets the minimum possible value in % at which the motor can keep the water supply pump running without stopping.

//The lower value (in percent) from which the PWM regulator for the pump will start working. Pick a value that is guaranteed to start the motor

`#define PWM_START_VALUE 40`

The value in % of the rated voltage at which the pump motor is guaranteed to start.

That is, if your pump (provided it is already installed in the system and has liquid in it) starts reliably at 4 volts and does not start at less, then with its rated voltage of 12 volts this is 30% of the rated value. So the value 40 can be replaced with 30.

2. In the settings on the "Temperature" («Температура») tab — the water sensor setpoint. The pump will maintain this temperature during the process.

After heating stops, cooling is switched off when the water has cooled to the setpoint minus **DELTA_T_CLOSE_VALVE** and at least three minutes have passed. The second option is that the previously heated boiler has cooled below 70 °C. For the mode with the TCA sensor, its cooling is also checked.

In **Wash column** mode, automatic cooling control is available after the **"Auto"** («АВТОМАТ») button is switched on in the web interface. In **Beer** and **Cheesemaking** modes the pump and valve are switched on at the cooling and transfer stages according to the program (see [Cheesemaking mode](#)). The valve is used only in the modes and stages where the firmware requests cooling.

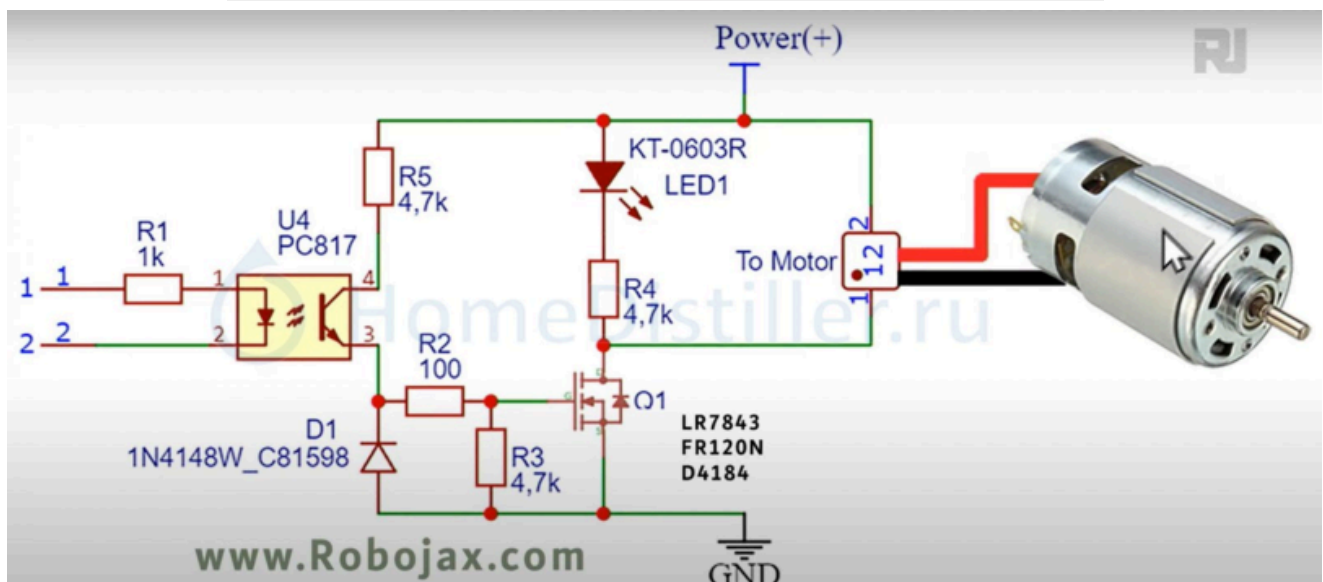
Implementation:

To control the pump (or the flow valve) you need an intermediary: a solid-state relay or a power board.

Attention.

Some solid-state relays may not trigger if connected directly to the ESP32 - there is not enough voltage and current.

It is recommended to use a board with a field-effect transistor (MOSFET) [LR7843](#)



This MOSFET controls the motor on the negative (low) side. Although it is claimed that the MOSFET hardly heats up, it is better to place the board in the enclosure in the zone of air movement.

Next: [diaphragm pump](#), [centrifugal pump](#) and [cooling without a pump](#)

Using a diaphragm pump

It is not recommended to connect the pump directly to a pressurized water line. It is highly desirable to install an intermediate tank with a float valve.

A float valve is simple and reliable. It works for years without problems. To guard against leaks, you can add an emergency overflow line.

Diaphragm pumps are mostly self-priming, but it is still better to take water from the bottom of the tank, through a fitting, so that the pump is always supplied with water.

Recommended pump: 12 V supply, up to 60 W, flow rate 2-3.5 L/min.

Note

During rectification on a 2" column, water consumption is about 40-50 L/h. During distillation with a 3 kW heating element, consumption is higher.

These are the pumps most commonly used.



Although its power is excessive, such models have proven themselves well. Note the screw and the cover on the left. This is a device that automatically cuts off the power when the pressure is exceeded. It may be adjustable or not. Some models may not have it.

If you plan to use such a pump, the 12 V power supply of the Samovar must be suitable, for example 12 V 100 W.

The pump is noisy, keep that in mind when choosing where to place it. *

If you plan to use the pump only for rectification, a more modest pump will do. Its small size lets you mount it directly on the tank or on the column. The noise from such a pump is also lower.



Operating voltage: 12 V DC,
Operating current: 0.5-0.7 A (power must be more than 6 W)
Flow rate: 1.5-2 L/min (or thereabouts),
Maximum suction lift: 78.7 inches (2 m)

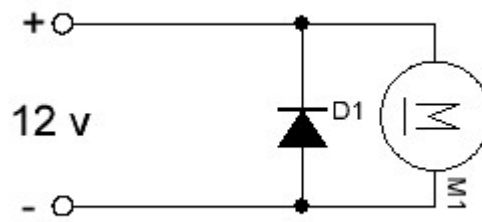
In principle, you can use any 12 V diaphragm pump whose specifications suit you.

* There is a way to reduce the noise and make the pump more comfortable to live with: change the PWM frequency. By default it is set in the file **pumppwm.h**; *to avoid editing that file, add your own line to **user_config_override.h**:*

```
#define PUMP_PWM_FREQ 15
```

15 is the default PWM frequency of the cooling pump, in hertz. You can set up to 450. This parameter is not available in the configurator.

You also need to add a flyback diode across the inductive load. It sounds scary, but it is simple: install a diode as close to the motor as possible.



Any diode will do, rated for a current not less than the motor current. Fast Schottky diodes have proven themselves well, for example **1N5844** or similar.

A little more detail - [post in the thread](#)

Using a centrifugal pump

Pumps come in many kinds: diaphragm, plunger, gear, centrifugal and others. The ones used for our purposes can be divided into 2 types: those through which liquid cannot pass under any conditions when switched off, and those through which liquid can flow by gravity.

The second type includes various centrifugal pumps with an impeller. How to tell? Blow into the inlet pipe; if air passes freely, it is this type.

About using this type of pump.

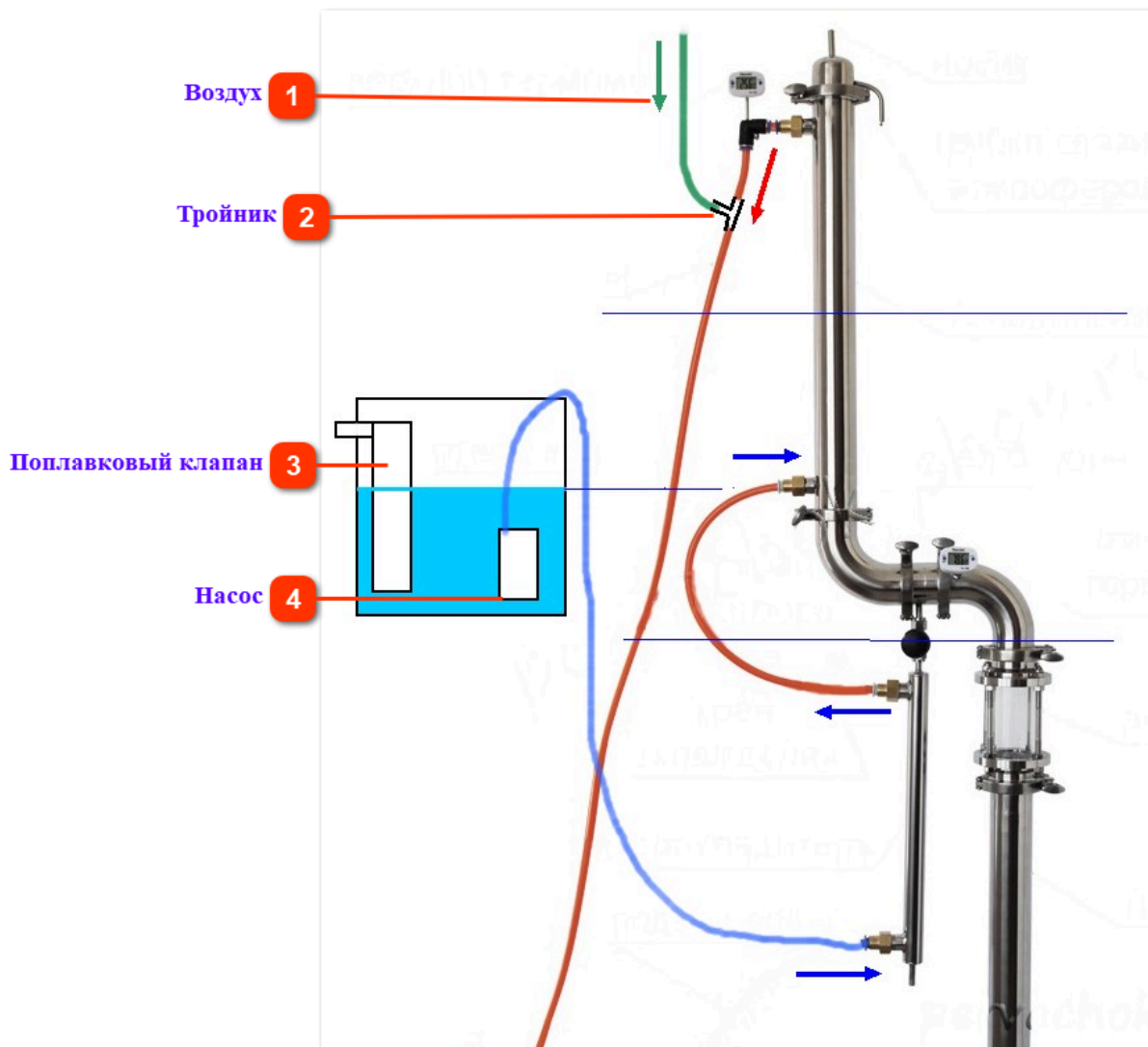
Which factors have a significant influence in use.

- The first factor is capacity. If the manufacturer states that the pump delivers 600 l/h and lifts water to a height of 5 meters, this does not mean that the pump will move 600 liters to a height of 5 meters in an hour. It is either 600 liters at the water intake level, or 5 meters of height and however much it manages to push out (if it pushes anything at all).
- The second factor is PWM. It determines not only the volume the pump moves, but also the height to which a column of water can be lifted.
- The third factor is a parasitic one. That is, the uncontrolled "gravity flow" that may occur in the system.

Let us look at organizing cooling with this type of pump, using a specific setup that our colleague uses and that has been working for more than a year under Samovar control.

The temperature delta is 1-2 degrees at a holding temperature of 60°C at the outlet.

A container (3-5 liters, as long as the fittings and the pump fit) with a float valve and a submersible aquarium pump is raised to the height of the dephlegmator. It is desirable that the water level in the container, held by the float, be at the level of the lower pipe of the dephlegmator. A deviation of 10-15 cm up or down is not critical.



- 1 Air**
Air enters through this tube to eliminate uncontrolled "gravity flow". In essence it is a TCA (atmospheric vent tube) for water
- 2 Tee**
Provides the "jet break"
- 3 Float valve**
Keeps a constant water level in the container.
- 4 Pump**

When using a **shell-and-tube dephlegmator**.

The pump feeds the lower inlet of the after-cooler, and from the after-cooler outlet the water goes to the lower inlet of the dephlegmator. The hot water exits from the upper outlet of the dephlegmator.

At the dephlegmator outlet, install a temperature sensor and make a "jet break".

To do this, install a tee on the discharge tube and raise the tube from it upward. The main thing is that the tee is above the liquid level of the container with the pump.

What this arrangement achieves.

The pump does not need to "know" how high the water goes; it starts providing exactly a water flow, rather than churning the water inside itself with the impeller while trying to build up pressure to lift the water up.

PWM really controls the water flow.

The "jet break" completely eliminates uncontrolled gravity flow.

Note:

"Jet break" is very important. In testing, without a jet break and with the pump switched off (that is, cooling only by gravity flow through the pump), at 1800 watts of applied power the outlet water temperature did not rise above 40°.

When a **Dimroth condenser** is used as the dephlegmator.

The pump feeds the lower inlet of the after-cooler, and from the after-cooler outlet the water goes to the upper inlet of the dephlegmator. The hot water exits from the lower outlet of the dephlegmator.

That is, the water flows against the steam.

At the lower outlet of the dephlegmator, install a temperature sensor, and preferably route the hot water discharge tube first up to the upper inlet of the dephlegmator and then down to the drain. After the discharge tube goes down, make a "jet break" as well. To do this, install a tee on the discharge tube and raise the "TCA for water" tube from it vertically upward. Make sure that the tee is not below the liquid level of the container with the pump.

The problems that occur with centrifugal pumps hardly concern owners of diaphragm pumps.

The only problem that occurred was very cold water and a capricious motor. At a set power of 15% the pump stopped, and at 20% the outlet temperature was too low.

By the way, this can happen if you use a pump rated for a higher capacity.

The way out is to make a return discharge right at the pump outlet. A tee is installed on the pump outlet pipe and a fixed discharge through a jet (a calibrated hole) is made into the same container. This way you can raise the minimum power value, while in fact less cooling liquid will go to cooling.

The controlled discharge principle can be applied to any pumps.

Without a Pump

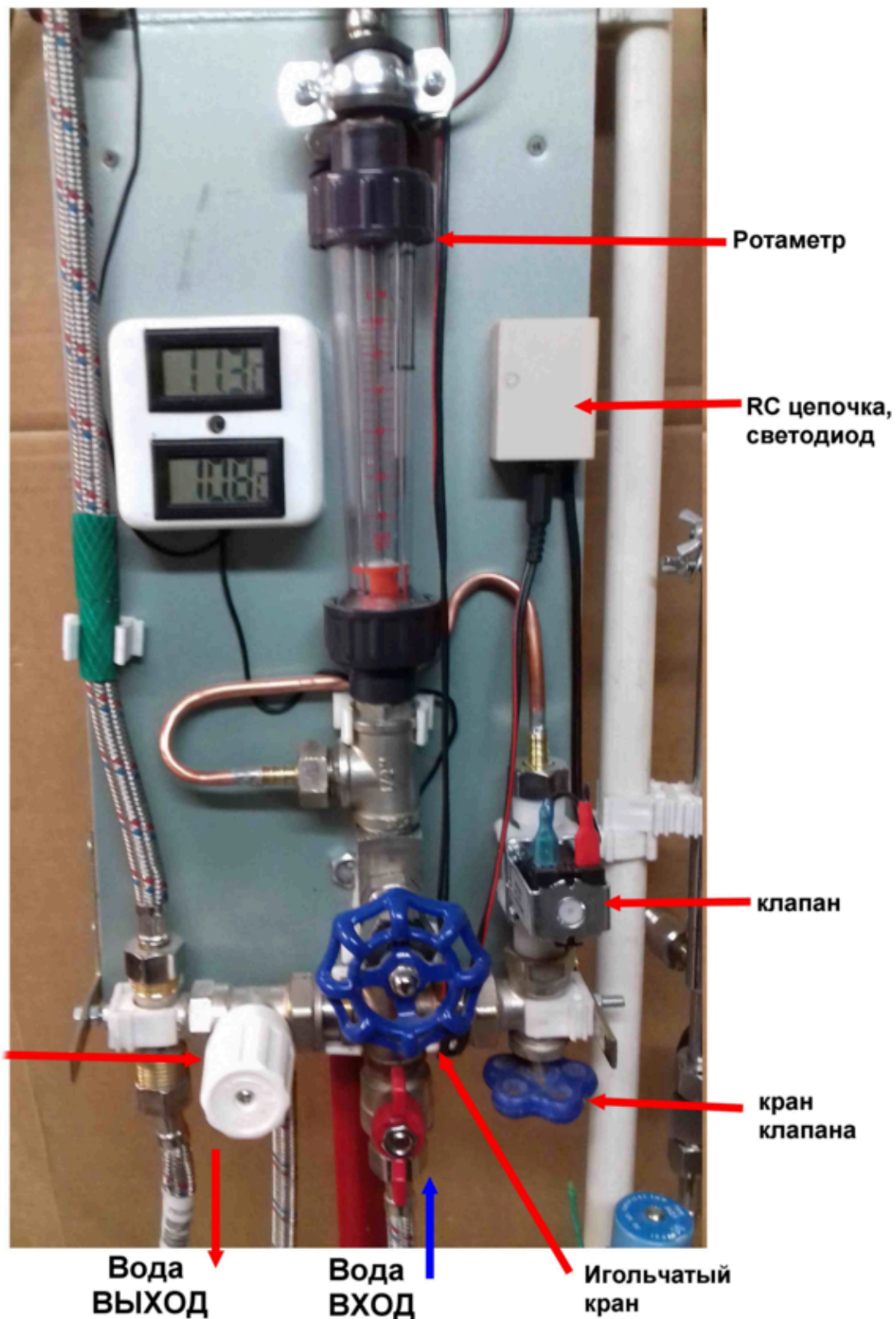
Samovar 7.00: the water pump and water valve are enabled on the "Pumps" («Насосы») tab of the configurator; the directives below are needed only for manual setup.

Semi-automatic mode

The point of this method is that it works more like a watchdog, and it works quite well.

Water is supplied from the mains (or from a pressure tank placed above the column).

The water goes to a needle valve and then to a rotameter, which is installed for adjustment and monitoring. A bypass line with another tap and a valve is built around the needle valve.



Set it up as follows:

Turn on the water. Close the needle valve completely. Apply voltage to the valve by force, so that the valve opens, and use the valve's tap to set the flow, for example 10 liters per hour. Remove the voltage - the valve closes.

The bypass line valve is connected to Samovar instead of the pump - to the pump control output (GPIO4 on ESP32 DevKit, GPIO11 on ESP32-S3) through a transistor switch or a relay, as described on the page [Cooling](#).

In the configurator, on the "Pumps" («Насосы») tab, clear the **"Control water pump or stirrer" («Управлять насосом воды или мешалкой»)** checkbox and enter **LOW** in the **"Water valve control" («Управление клапаном воды»)** field (or HIGH, if your switch turns on with a high level). The configurator writes this to the Samovar_ini.h file, where the same lines can be edited manually:

```
//#define USE_WATER_PUMP //использовать управление ШИМ для  
насоса для воды  
must be commented out
```

and the line:

```
#define USE_WATER_VALVE LOW //использовать управление  
клапаном для регулировки напора воды  
must be uncommented. While USE_WATER_PUMP is enabled, the USE_WATER_VALVE directive has  
no effect - in the firmware they are mutually exclusive.
```

Then start the column and use the needle valve to set the water flow so that the temperature at the dephlegmator outlet is, say, 50°. Bring the column to its working mode, adjusting the flow slightly with the needle valve. That's it - the column is cooled.

If the water temperature goes above the water sensor setpoint, the valve on the bypass line opens and an extra 10 liters per hour goes to cooling; as soon as the temperature drops, the valve closes.

The valve opens when the water temperature rises more than 1 °C above the setpoint, and closes when it falls more than 1 °C below the setpoint. This is how the rectification and distillation modes work. In Beer mode this output is not used - only the relay No. 3 valve opens; in cheesemaking mode the bypass valve is open while the product temperature is more than 1 °C above the target.

In this mode the system can run for quite a long time. But it should not be left unattended for long, and the flow needs to be adjusted periodically with the needle valve.

Reflux Level Sensor

Samovar 7.00: the sensor enabling and polarity are set on the "Sensors" («Датчики») tab of the configurator and are saved to user_config_override.h.

The XKC-Y25-V contactless liquid level sensor is built around a capacitive sensing element that responds to changes in dielectric permittivity. This property allows the level detector to sense liquid through various non-metallic barriers: plastic, glass, ceramic, etc. The liquid sensor will even register the touch of your hand. The maximum wall thickness of the barrier, and therefore the sensing distance of the sensor, is 12 mm. The sensor will not work with metal barriers.

You can read more at: [XKC-Y25-V contactless liquid level sensor](#)

In fact, Samovar can work with any sensor (laser, inductive, contact etc.) that can run on 3.3 volts (or 5 volts, provided a logic level converter is used) and output, when triggered, a logic one or a logic zero. The choice of which sensor to use can be made in the Samovar_ini.h file

Use a reflux level sensor that outputs a logic one when triggered.

```
#define USE_HEAD_LEVEL_SENSOR           //использовать датчик уровня флегмы  
в голове (датчик P-N-P).
```

Use a reflux level sensor that outputs a logic zero when triggered

```
#define WHLS_HIGH_PULL                   //использовать датчик уровня жидкости N-P-  
N
```

(Below, all examples are based on using the XKC-Y25-V P-N-P sensor)

Choosing the output signal mode.

Some sensor models have 4 pins and have a built-in output signal inversion circuit, which is controlled by the state of the MODE pin.

Normal mode

The MODE pin is connected to the positive supply or not connected at all: the OUT signal is not inverted.

- If the liquid level is within the sensing range of the sensor: the LED is on, the sensor output is high.
- If the liquid level is outside the sensing range of the sensor: the LED is off, the sensor output is low.

Inverted mode

The MODE pin is connected to ground: the OUT signal is inverted.

- If the water level is within the sensing range of the sensor: the LED is on, the sensor output is low.
- If the water level is outside the sensing range of the sensor: the LED is off, the sensor output is high.

Where to place it and how to mount it,

The sensor is placed on the sight glass, because it does not work through metal.

Stage 1 - temporary-permanent.

You will need:

- a plastic card
- two office clips
- the sensor
- a marker
- a step drill
- a knife



Take the card, fasten it with a clip to the sight glass posts (the photo is not very good), and mark where the second post is.

Cut it and drill a hole with the step drill, slightly smaller than needed. Trim it with the knife so that the sensor fits tightly.



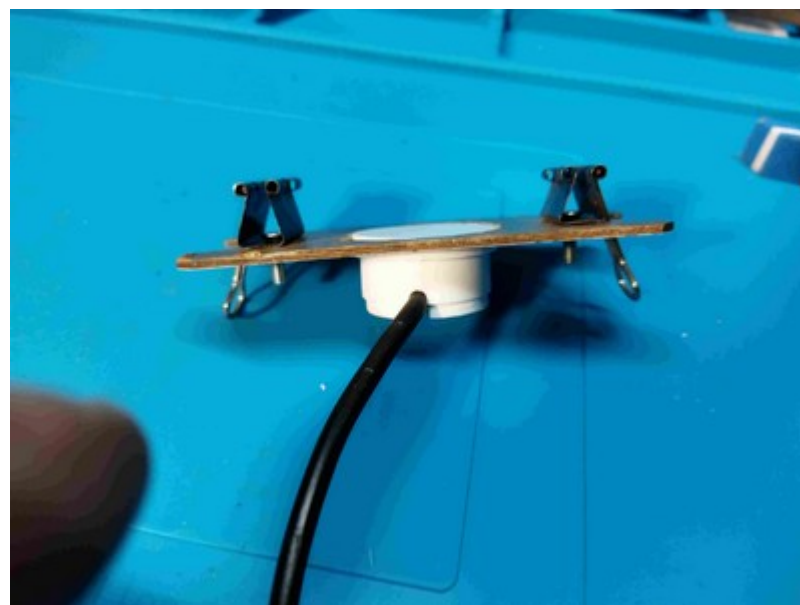
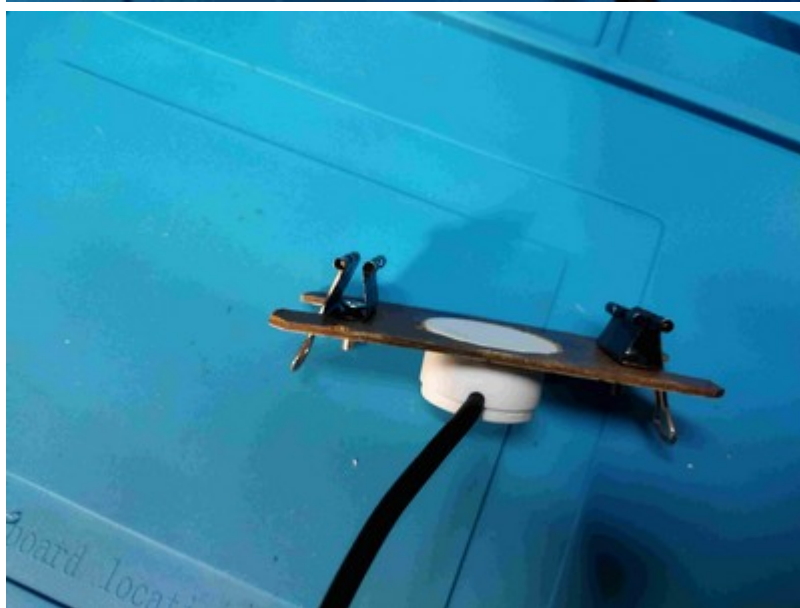
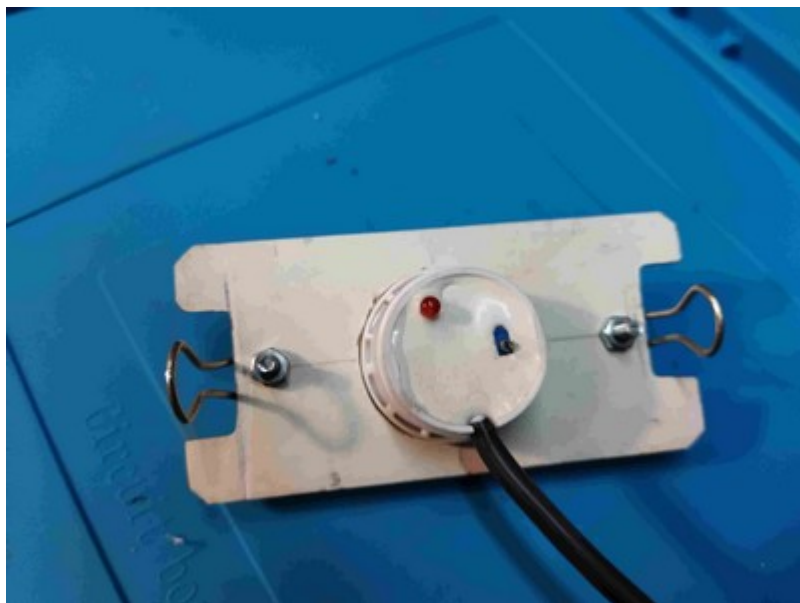
Draw + (plus) and - (minus), this is the sensitivity adjustment, remove the cover from the sensor, insert it into the hole and fasten it to the sight glass with the two clips.



Done! Don't put the cover back - you will have to turn the trimmer to catch the trigger point.

Permanent.

Take a piece of rigid plastic and make a hole in it in the same way, but now the clips are drilled and fastened with M3 countersunk screws.



Remove one spring and mount it on the sight glass in the same way.



What this gives. First, the rigidity of the structure. Second - the ability to mount it as low as possible and even at an angle.

The sensor is capacitive. Adjust it with a screwdriver with a long insulated handle, preferably one that is entirely plastic or mostly plastic.

Connect the sensor to the LEVEL connector on the printed circuit board. The check is simple - connect it, bring your hand close, and the LED lights up. Samovar counts a reflux level sensor trigger as a period of 3 seconds during which the sensor LED is on continuously. If the LED blinks at an interval shorter than 3 seconds, this is not counted as a trigger.

If Samovar determines that the sensor has triggered, the regulator is commanded to lower the voltage: by 1 V for KVIC, by 2 V for RMVK, by 3% of the current power for SEM_AVR. The next reduction is possible no sooner than 40 seconds later - the process is slow to respond. During the "Pre-flooding" («Предзахлёб») program, the sensor triggering is a normal event and no flooding message is sent.

After heating up, when the column reaches stabilization mode and the sight glass has warmed up, you need to set the trigger threshold.

First, reduce the sensor sensitivity, bring the column to its working mode, and start taking off heads. Use a screwdriver to turn the trimmer screw until the sensor triggers, and then turn it back a little.

While taking off heads, you can flood the column by raising the power, to make sure the sensor triggers correctly and to fine-tune it. This episode will not affect the quality of the final product.

The finer the adjustment, the earlier the sensor reacts to reflux appearing in the sight glass, but false triggers are then possible.

If the sensor was removed or the column stood idle for a long time, the adjustment must be repeated.

Possible problems.

- The sensor is not connected, but Samovar reports that the sensor has triggered.

When setting up the sketch, you can clear the "Use reflux level sensor" («Использовать датчик уровня флегмы») checkbox on the "Sensors" («Датчику») tab of the configurator, or comment out the line in the Samovar_ini.h file:

```
//#define USE_HEAD_LEVEL_SENSOR           //использовать датчик уровня  
флегмы в голове (датчик P-N-P)
```

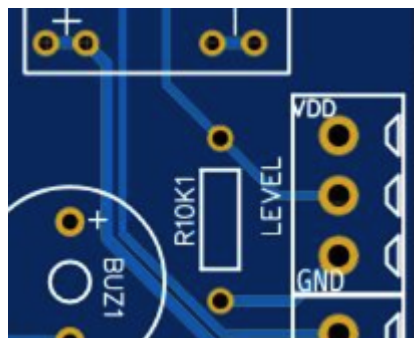
Or another way:

In the Samovar_ini.h file, comment out the line

```
//#define IGNORE_HEAD_LEVEL_SENSOR_SETTING //Если датчик флегмы  
используется, то игнорировать команду на отключение датчика в  
WEB-интерфейсе. Then, in the interface, a checkbox will appear on the Settings  
(«Настройки») tab with which you can enable or disable the use of the reflux sensor.
```

- The sensor is connected, does not trigger, but Samovar reports that the sensor has triggered.

When using the sensor, the R1-10k resistor (located next to the LEVEL connector) must be installed.



If the sensor outputs a logic one when triggered, the level input must be pulled to GND: GPIO27 on ESP32 DevKit or GPIO19 on ESP32-S3. On the standard DevKit printed circuit board this pull is already provided.

If the sensor outputs a logic zero when triggered, the level input must be pulled to +3.3 V: GPIO27 on ESP32 DevKit or GPIO19 on ESP32-S3.

Driver. Stepper Motor.

Samovar 7.00: the motor direction and smooth acceleration are set on the "Stepper motor" («Шаговый двигатель») tab of the configurator.

The project uses stepper motor drivers (hereinafter the motor) A4988 or DRV8825.

Which driver to choose? Read this article to get an idea of the subject; the main differences between these drivers are given at the end.

It is best to install the driver the same way as the ESP32 - on a pin header. Before setting up the driver, the ESP32 must be installed on the board and the Samovar sketch must be uploaded to it.

Do not connect the power or the motor.

Take the driver, going by the pin labels on the driver board and the printed circuit board, and insert it into the header.

Check with a multimeter in continuity mode. The GND pins must be at ground, VMOT must have +12 volts from the VIN power connector, and VDD must have +3.3 from the first pin of the ESP32.

Apply 12 volts of power to the VIN connector. You can check once more that the voltages are present.

The printed circuit board has switches for the driver:

- H1 - connect all three pins
- SW1 - if there are switches, set them all to the OFF position. If there are no switches - do not install the jumpers yet.

Now the driver needs to be set up.

Do not connect the motor.

A driver is a delicate thing. There are even recommendations to set it up with the power turned off.

That is, turn the trimmer resistor a little, apply power, measure the voltage, remove power - turn it again. And so on all the time, until you set the value you need.

There is another, faster way.

A screwdriver with an insulated handle. Connect one multimeter probe anywhere on the printed circuit board to ground (GND), and with the other either carefully touch the trimmer and turn the wiper with the screwdriver, or put

an "alligator clip" on the probe, clip it to the screwdriver tip and turn the screwdriver through the clip.

See the setup procedure at the links.

[How to set up the DRV8825 stepper motor driver](#)

[How to set up the A4988 stepper motor driver](#)

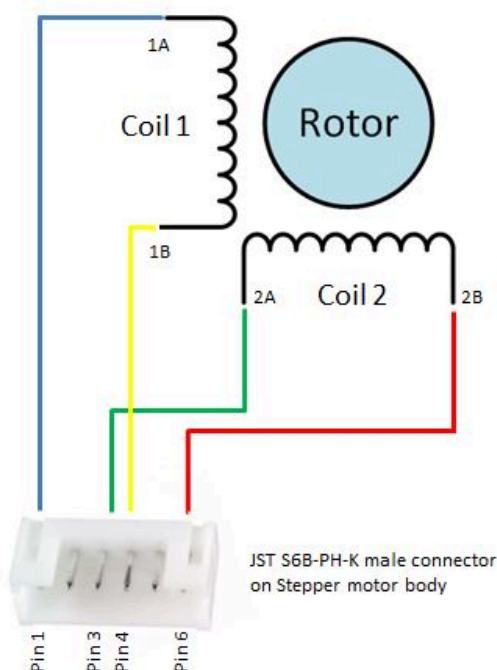
Connecting the stepper motor.

The project uses a bipolar NEMA17 stepper motor. First check that the motor matches the project.

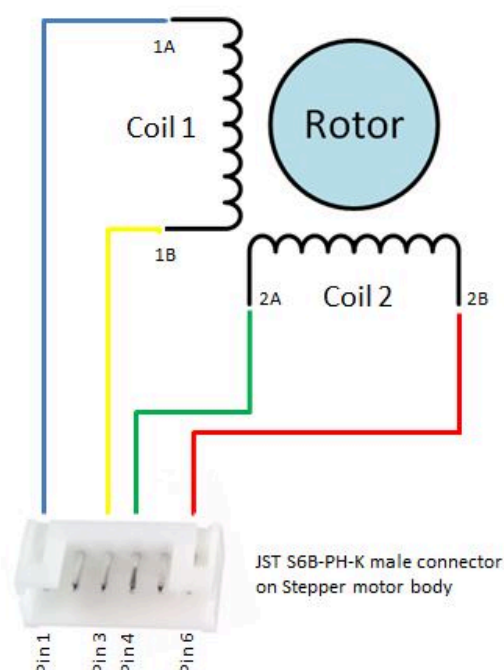
Bipolar motors have two independent windings. Motors from different manufacturers may have different pin layouts.

Nema 17 internal connections of coils to S6B-PH-K male socket

Crossover



Straight



Connect the connector with wires that came in the kit to the motor, take any wire and use the multimeter continuity test to find its pair. This is one winding. Mark both wires. Check the second pair. If it beeps, good. Now take any wire from one pair and any wire from the second. Test them. They must not beep. If they do beep, it is not a bipolar motor.

Now connect the motor to the connector on the printed circuit board. According to the picture above 1A-1B is one winding, 2A-2B is the other. "A" is the start of a winding and "B" is the end. You can, of course, connect strictly by the labels, but it is not necessary.

The main thing is to connect one winding to pins 1A-1B of the driver and the second to 2A-2B. It does not matter which goes where, and it does not matter whether it is the start of the winding or the end.

Connected. Apply power. The motor does not turn. Open the web interface and give the take-off command.

The motor shaft should start to turn.

The motor hums and does not turn; the motor turns in jerks

- Check that the motor is connected to the driver correctly and at all. To the driver itself, not only to the printed circuit board.
- Check that +3.3 volts is present on the driver's RST and SLP pins.
- In the web interface, on the Settings page, "Pump" («Насос») tab, press "Pump calibration" («Калибровка насоса»). If the "Stepper motor speed" («Скорость шагового двигателя») field is set to 8000 (or similar), change it to 800

([forum post](#): ...I had a similar situation when setting up Samovar. The difference was the TMC2208 driver. The problem was solved by choosing the step division; the pump worked stably only at 1/8, and in the other modes it either hummed or ran in jerks with wild skips (although on another device this driver and pump worked at all divisions; the only difference was in the control signals, 5 V rather than 3.3 V).

Perhaps it is somehow connected with the frequency of the STEP pulses; then try to choose the "stepper motor speed" («скорость шагового двигателя») parameter in the "Stepper motor calibration" («Калибровка шагового двигателя») window (say, set 4000 rather than 8000). And then calibrate for 100 ml.

Remember to select the motor operating modes with the power of the Samovar printed circuit board completely disconnected.)

If the shaft turns in the wrong direction, swap the wires of either winding at the connection to the board.

Note.

In the file Samovar_ini.h contains a line that lets you change the rotation in software:

```
//#define STEPPER_REVERSE //change the stepper motor rotation direction
```

In the configurator this is the "Reverse stepper motor direction" («Обратное направление шагового двигателя») checkbox on the "Stepper motor" («Шаговый

двигатель») tab; the same tab has "Smooth stepper motor acceleration" («Плавный разгон шагового двигателя») (USE_STEPPER_ACCELERATION). But it is more convenient to swap the wires once than to watch the parameter at every update.

Run the motor at different speeds through the take-off program. Set the take-off to 70-150 ml/h. Hold the motor body with one hand and try to stop the shaft with the other; if the shaft stops easily, go back to the driver settings and raise the voltage a little, literally by 0.02-0.04 volts.

Check again how easily the shaft can be stopped.

Glue the heat sink that comes in the kit onto the driver chip.

Now you can install the pump head. First lubricate the rollers and the tube with neutral silicone grease. Lubrication will need to be done periodically (once every 5-7 distillation runs); this extends the life of the rollers and the tube, and the motor will have an easier time too.

Run it with the pump, see whether there are any stops, whether the motor sounds overloaded, and how hot the motor and the driver chip heat sink get...

Now you can move on to fine tuning, if it is needed.

A little theory.

Unlike traditional DC and AC motors, stepper motors move in discrete steps or increments, hence the name "stepper".

Each step of a stepper motor corresponds to a precise angular rotation.

Typically the step angles of stepper motors are 1.8° or 0.9° , which corresponds to a full revolution divided into 200 or 400 steps respectively.

Standard NEMA17 motors have a rotation angle of 1.8° per step (200 steps per full revolution)

The motor can work in several modes - full-step and microstep.

Full-step mode is when the motor turns the shaft by 1.8° in 1 step

In full-step mode the motor has the highest torque of all the modes.

Microstepping is dividing the standard step into several microsteps. The standard step can be divided into 2, 4, 8, 16 and 32 parts

The step division can be set with the SW1 switches

MS1, MS2, MS3 – microstep resolution selection pins. With them we can choose one of five modes:

For the **A4988** driver

MS1	MS2	MS2	Steps
off	off	off	Full
on	off	off	1/2
off	on	off	1/4
on	on	off	1/8
on	on	on	1/16

For the **DRV8825** driver

MS1	MS2	MS2	Steps
off	off	off	Full
on	off	off	1/2
off	on	off	1/4
on	on	off	1/8
off	off	on	1/16
on	on	on	1/32

Explanation.

Suppose the 1/4 mode is chosen. Then in 1 microstep the motor shaft will turn by $1.8^\circ/4 = 0.45^\circ$, which corresponds to $200 \cdot 4 = 800$ (or, counting differently, $360^\circ/0.45^\circ = 800$) microsteps per 1 full revolution of the shaft.

In microstep mode the motor has reduced torque but increased smoothness and accuracy (which is not particularly important for the project). Stepper motors in full-step mode produce higher levels of vibration and noise. Microstepping significantly reduces vibration and noise, which makes the motor run more smoothly and quietly.

In general, watch and listen to how your motor works. You can try different modes and compare them.

Attention.

The motor operating modes are selected with the power of the Samovar printed circuit board completely disconnected.

After you decide on the motor operating mode and set the DIP switches to the right position (or finally solder the jumpers), you will need to calibrate the pump. The calibration must be repeated every time you decide to change the motor operating mode.

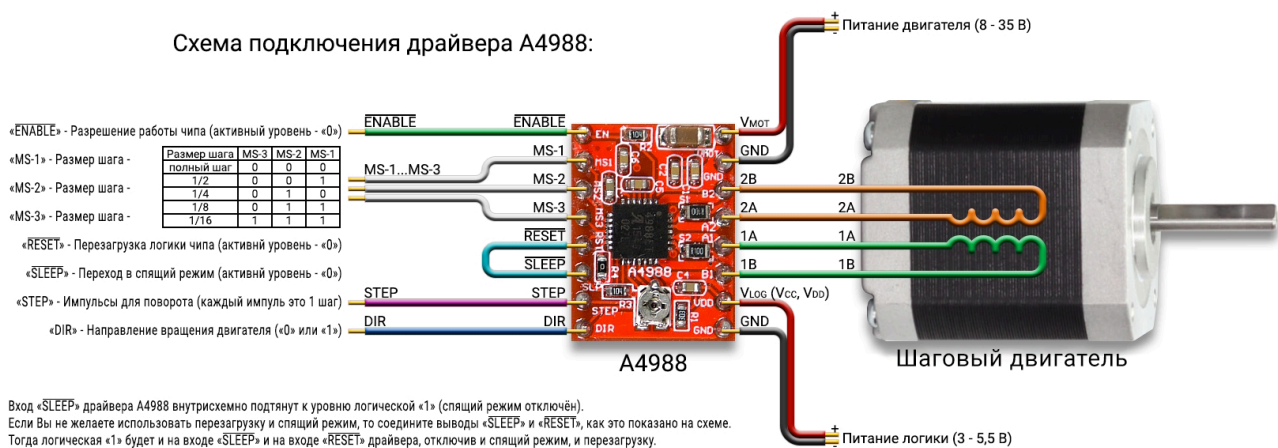
The accuracy of take-off is also affected by how regularly the pump is lubricated. The less often you lubricate, the lower the actual take-off rate.

Differences between the DRV8825 and A4988 motor drivers (in simple terms, without specific details)

- Different placement of the potentiometer that sets the current limit.
- The DRV8825 driver supports 1/32 step in microstep mode.
- The A4988 driver supports at most 1/16 step.
- The DRV8825 driver provides up to 1.5 A per phase without a heat sink,
- The A4988 driver provides 1 A per phase without a heat sink.
- The A4988 driver is cheaper than the DRV8825 driver.

Reference information.

Схема подключения драйвера A4988:



Fraction Distribution

Samovar 7.00: the corrections for the 11 servo positions are set in the "Servo corrections (11 numbers)" («Поправки сервопривода (11 чисел)») field on the "Equipment" («Оборудование») tab of the configurator; it writes them to the servoDelta array in the file Samovar_ini.h.

Samovar can divide the whole run into 11 parts, according to the take-off program you have made.

Note.

If you do not need the division into parts, or you will change the containers yourself, this unit is not required. Samovar works without it.

To build it you need a servo of the kind used in Arduino projects.

Servo [SG90](#)

(Rc Mini Micro 9g 1.6 kg servo SG90 180° For RC 250 450 for Arduino DIY)



Warning!

Samovar uses a **servo with a 180° rotation angle**; any other will not work correctly (servos can have a rotation angle of 90°, 180°, 360° or continuous rotation).

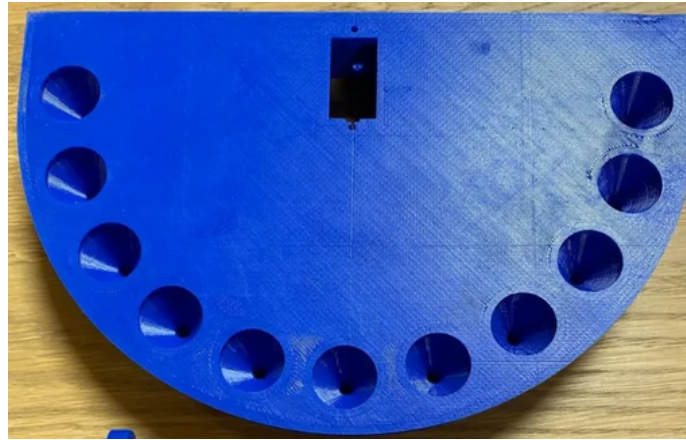
Choose the servo carefully before buying. If in doubt, it is better to pass and buy from another seller who clearly states the rotation angle.

Wire pinout: red — +5 V, brown — GND, yellow or orange — control signal. The signal input is connected to GPIO25 on ESP32 DevKit or to GPIO10 on ESP32-S3.

To check the servo rotation angle you can use a special [sketch](#). Put the sketch in the same directory as the Samovar files (... Documents\Arduino\...) and install it on the ESP32 instead of Samovar. See the [User Manual](#) for how to install a sketch.

After the sketch is installed on the ESP32, the servo should start moving from the initial position to the extreme one and back, in a loop. After checking the servo, you will need to upload the Samovar sketch to the ESP32 again.

A model of a collection table for 3D printing has been developed for the SG90 servo. The model files are in the 3D directory, which is included in every Samovar release archive. These are the files **Samovar_plate.stl** and **Samovar_horn.stl**. Print with a plastic resistant to acetone and alcohol, for example PLA.



The SG90 is one of the cheapest and most popular servos. Some users have had problems with its stable operation.

It showed up as the servo struggling to reach the required position or, at some point, the rocker arm starting to "tremble". Experience shows this is related to an unfortunate placement of the product feed tube, its excessive size, stiffness or, on the contrary, floppiness.

The SG90 can be replaced with:

Servo [MG996R](#) , MG996, MG995.



This servo is more expensive than the previous one, but it is also more powerful (and so consumes more energy), can have either a plastic or a metal gearbox (it makes no difference for our purposes), and comes with more accessories.

Samovar uses a **servo with a 180° rotation angle**; any other will not work correctly.

Wire pinout: red — +5 V, brown, gray or black — GND, yellow or orange — control signal. The signal input is connected to GPIO25 on ESP32 DevKit or to GPIO10 on ESP32-S3.

A collection table model for 3D printing has also been developed for these servos. These are the files **plate_996_servo.stl** and **horn_996_servo.stl**. The model files are in the 3D directory, which is in the Samovar release archive. Print with a plastic resistant to acetone and alcohol.

Examples of collection table builds and modifications:



See on the forum: [post 1](#), [post 2](#)

Position correction

Every servo has its own errors and is tuned individually to the table positions.

If the collection table is printed or made from a template, with equal distances between the funnels, the rotation angles will need to be corrected.

The servo rotation angles are set in the file `Samovar_ini.h` (see [Uploading and Updating the Sketch](#) and [Sketch Setup](#)),

these are the lines:

//Correction for the servo rotation angle.

```
int8_t servoDelta[11] = {0, -2, -3, -4, -3, -2, 0, 0, 0, 0, -2};
```

The values in the curly braces are the angle correction for each position. There are 11 in total. The receiving funnels are counted starting from position zero. The correction values are given in degrees. Each position is corrected individually. It does not depend on the correction of neighboring positions.

For the correction, make a test take-off program for the "Rectification" («Ректификация») mode

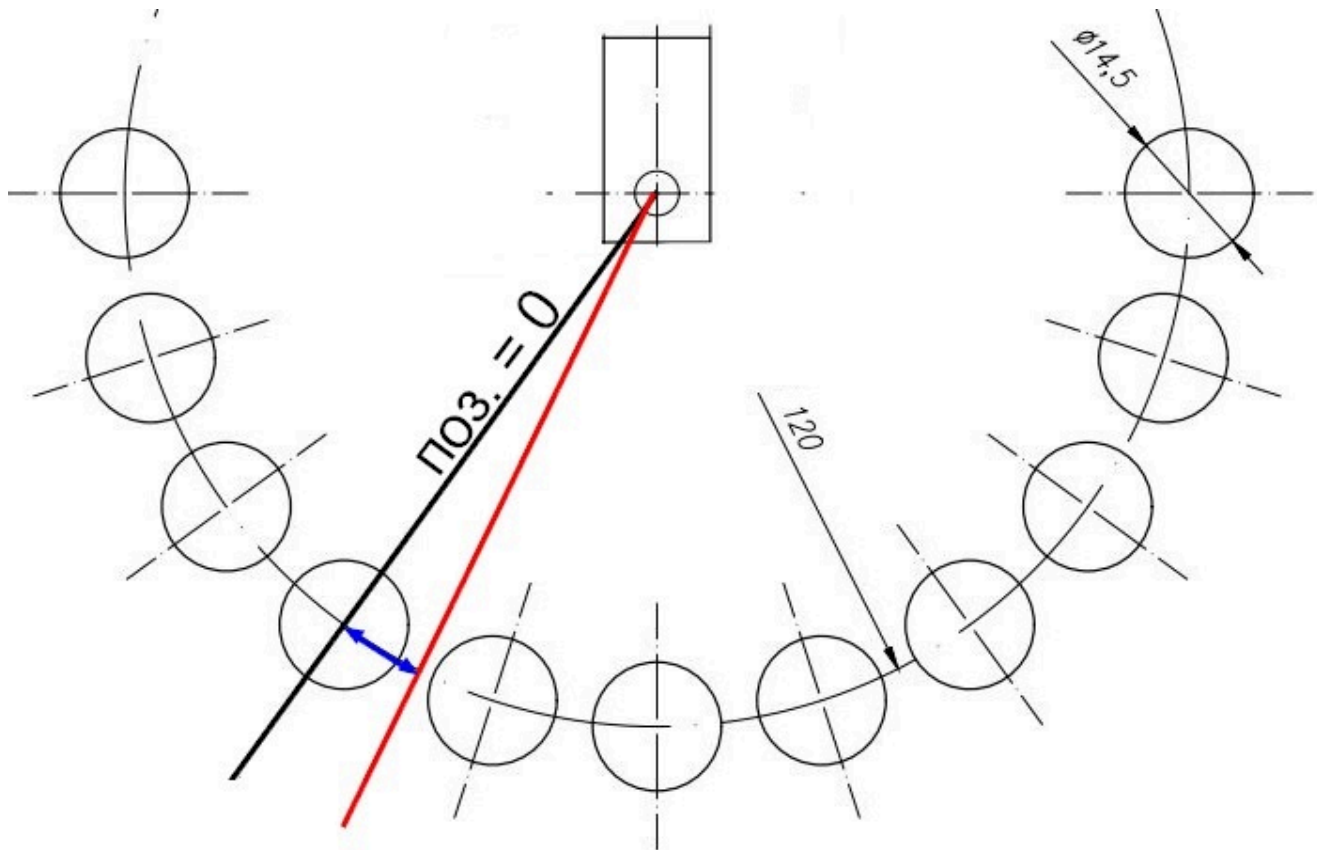
```
H;450;0.10;0;0.00;45
H;450;0.10;1;0.00;45
H;450;0.10;2;0.00;45
H;450;0.10;3;0.00;45
H;450;0.10;4;0.00;45
H;450;0.10;5;0.00;45
H;450;0.10;6;0.00;45
H;450;0.10;7;0.00;45
H;450;0.10;8;0.00;45
H;450;0.10;9;0.00;45
H;450;0.10;10;0.00;45
```

Copy this block of text and paste it into the take-off [program text field](#) on the "Program" («Программа») tab of the Samovar web interface main page, and save.

You can leave the rocker arm off the servo, or install it in a way that lets you remove it easily. Connect the servo and turn on Samovar. At startup Samovar puts the servo in its start position. Install the rocker arm so that it points at funnel zero.

From Samovar give the "Turn heating on" («Включить нагрев») command (the heater must be disconnected) and then the "Start take-off" («Начать отбор») command. The arm stays at funnel zero. The "Next program" («Следующая программа») command moves the arm to the next position - funnel No. 1. If the arm position coincides with the imaginary line passing through the center of the servo's axis of rotation and the center of the receiving funnel, good. That is zero. No correction is needed. Give the "Next program" command again, and the arm moves to the next position.

If the arm has not reached the required position or has overshoot it, a correction is needed. Take funnel No. 3 as an example (remember that funnel numbering starts from zero). For this funnel the correction value in the original sketch is 4 (minus 4)



Suppose the arm has "overshot" and stopped on the red line. Measure the length of the arc (blue arrow) between zero and the actual position.

Say it is 10 mm.

Now you need to find the arc length for a rotation of 1 degree. Find an online [calculator](#) . Enter the radius value (120 in this case) and set the angle = 1.

We find that for a rotation of 1 ° the arc length is 2 mm. Our arm is 10 mm away from the required position.

$10/2=5$ degrees.

The arm has overshoot and must be brought back, so this will be a negative value. The sketch already has - 4 (minus 4)

$-4+(-5)= -9$

So the rotation angle correction for funnel No. 3, which is position No. 4, is minus 9°

This is true for a particular servo and table, with funnels placed on a circle of 120 mm radius.

In the same way, run through all the arm positions and correct the values in the curly braces of the line `int8_t servoDelta[11] = {0, -2, -3, -9, -3, -2, 0, 0, 0, 0, -2};`

After the correction, upload the sketch to Samovar again, set the test take-off program and check how the arm moves between positions.

Making a collection table yourself

A collection table with fixed funnel positions. A post [on the forum](#). The template for making it can be downloaded there too.

A collection table that needs almost no correction of the servo rotation angle values.



The point is that the centers for drilling the holes are marked according to the actual position of the rocker arm.
Making instructions [on the forum](#) .

Funnels



- Buy:

[Metal kitchen mini funnel with a removable filter, 5 pcs.](#)

[SUEF small plastic oil funnels, 2/5/10/20 pcs., liquid oil funnels.](#)

[Stainless steel kitchen mini funnel with a removable filter, 5 pcs](#)

-

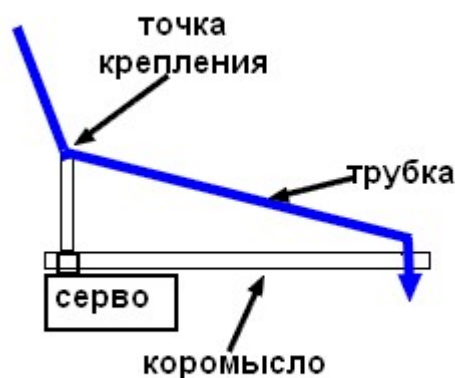
- Make on a 3D printer - the funnel file is in the 3D directory.
- Use [vial caps](#) or medical [syringes](#). Syringes smaller than 5 ml are not recommended - they may overflow; 10 or 20 ml is better.

If the receiving funnels overflow, the simplest fix is to place the receiving containers as low as possible and raise the table higher. The column of liquid in the tube from the funnel to the container adds extra speed to the flow.

Troubleshooting possible problems.

Assuming the servo is working

- Samovar reboots while the servo is working - the 12-5 volt converter may not have enough power, or the servo motor draws too much. What you can do:
 - As a temporary measure, increase the capacitance of capacitor C4. C3 too, if you like.
 - Add a second 12-5 volt converter. Join the grounds and feed +5 to the servo (you can power all the 5 volt consumers from this converter)
 - The rocker arm "trembles" over the funnel, even without the tube attached - *poor horizontal stiffness of the rocker arm.*
 - The rocker arm "swings" over the funnel with the liquid feed tube attached.
 - The tube is too stiff. It constantly "pulls" the arm back. The servo keeps trying to return it to the position.
 - The tube "dangles" above the table and pulls the arm first one way and then the other.
- As an option, additionally fix the tube at the center of the axis of rotation of the servo gearbox.*



Emergency button. Additional emergency sensors

Verified against Samovar 7.00.

Connect the button or sensor between the emergency input and GND. It triggers on a low level: GPIO35 on ESP32 DevKit or GPIO48 on ESP32-S3.

When triggered, Samovar switches off the heating, the water supply, the take-off and the stirrer, and blocks a restart until the board is rebooted. Several normally open buttons and sensors can be wired in parallel. For ESP32 DevKit, use the PlatformIO environment **Samovar_alarm_button** or add **#define USE_ALARM_BTN** to **user_config_override.h**. In the **Samovar_s3** environment the GPIO48 emergency input is already enabled.

Alcohol spill sensor. [Beginning](#). [Implementation](#).

On ESP32 DevKit the emergency input is GPIO35; on ESP32-S3 it is GPIO48. In both cases it triggers when the input is shorted to GND.

For GPIO35 on the classic ESP32, an external 10 kOhm pull-up resistor to +3.3 V is required: pins GPIO34–39 have no internal pull-up. For GPIO48 on ESP32-S3, the firmware enables the internal pull-up.

On the Samovar 1.3.2 board, the emergency button connector is labeled ALRM_BT (older materials also call it ALARM_BTN). It is connected to GPIO35 and must have a 10 kOhm pull-up resistor to +3.3 V.

Links to the posts: [link 1](#), and **ais77** [link 2](#), [link 3](#).

Recommended Configuration

The recommended configuration adds to the minimal build the actuators needed for automatic operation:

- a power regulator and a contactor for heating control;
- a water supply valve and, if needed, a cooling pump;
- a stepper motor and a driver for take-off;
- temperature sensors for the boiler, steam, column section and water;
- an atmospheric pressure sensor;
- a reflux level sensor or other emergency sensors you use.

An LCD display, an encoder and a separate button are convenient for local control but are not required: the main settings and controls are available through the web interface.

LCD display, encoder, button

Samovar 7.00: the control button is enabled on the "Hardware" («Оборудование») tab of the configurator.

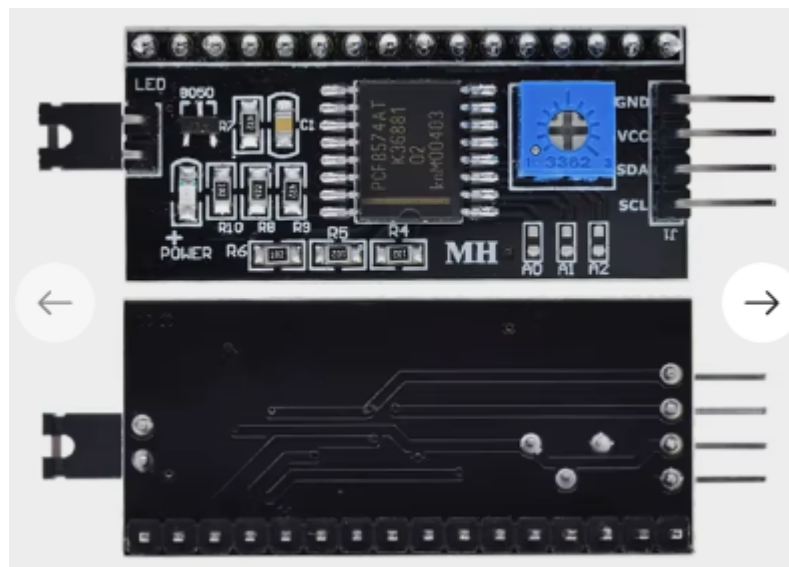
Warning.

Samovar works without a display, an encoder and the universal button. All the information is available in the web interface and, if the integration is enabled, in Blynk as well. A display is handy next to the unit; an encoder without a display is usually not needed, while a separate button can be useful.



The project uses a four-line display: 20X4 LCD2004 with an I2C module. Without the module you will not be able to connect the display to Samovar.

Here is the module I2C



And this is a 20x4 display with the I2C module soldered on.



You can buy a display with the module already soldered on, or buy them separately as a kit and solder it yourself.

The display backlight can be blue, green, yellow or white (this affects the price). If you do not need the backlight or it gets in the way, you can turn it off after setting up the display by removing the LED jumper on the I2C module. The text stays readable.

Most displays support only the Latin font, but some support Cyrillic. Samovar uses Latin.

Connection.

The display is connected to the Samovar board with 4 wires from the I2C module, according to the pinout of the module and the board.

As a rule, on the first power-up the backlight comes on but there is no text. Turn the trimmer resistor on the I2C module. The text will appear. Set the highest contrast.

That completes the display setup.

Note.

If some lines or characters on the display are shown incorrectly, carefully re-solder the connections between the module and the display. Remove the flux.

Encoder

The project uses an EC11 encoder with a "breakout board". It looks like this.



It is connected to the Samovar board with 5 wires according to the pinout. Its main job is "scrolling" through the lines on the LCD2004 display. If the encoder scrolls the display lines in the wrong direction, swap the wires on the S1 and S2 contacts.

Button.

How Samovar reacts to a button press depends on the selected operating mode (see [LCD display and universal button](#)).

A normally open tactile button will do. Connect it between the input and +3.3 V: GPIO39 on ESP32 DevKit or GPIO47 on ESP32-S3. The input needs an external 10 kOhm pull-down resistor to GND.

If you do not use the button, the resistor is not needed on ESP32 DevKit, but you must clear the "Use control button" («Использовать кнопку управления») checkbox on the "Hardware" («Оборудование») tab of the configurator, or comment out the button in the Samovar_ini.h file:

```
//#define USE_BTN //use the button
```

On ESP32-S3 the button input GPIO47 is always enabled regardless of this setting, so the 10 kOhm resistor to GND is needed even without a button; otherwise false presses are possible.

Warning!

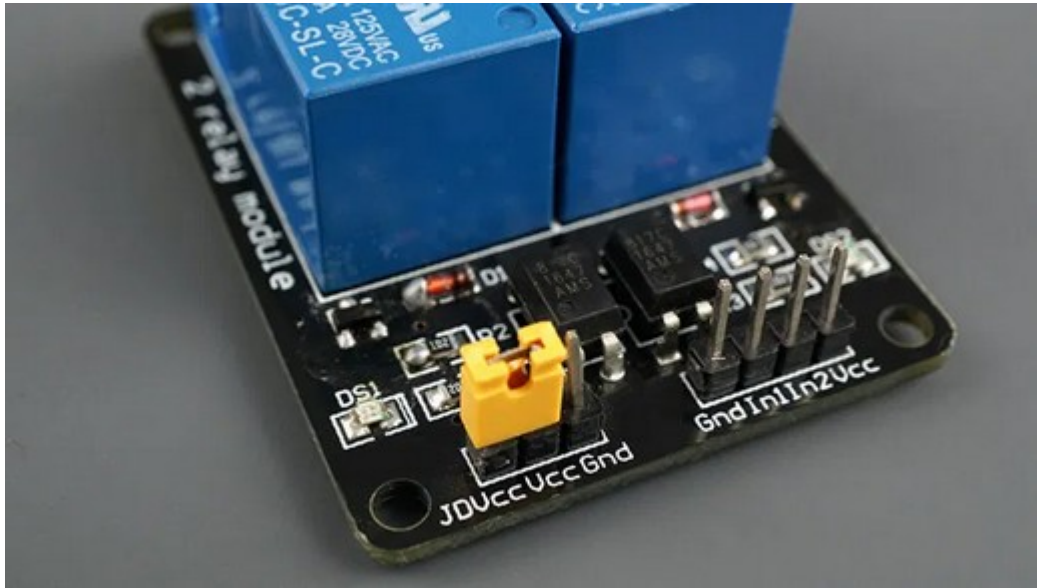
At startup, holding the separate button for about two seconds starts the Wi-Fi access point. Holding the encoder button for about two seconds deletes only the saved Wi-Fi settings. The other hardware settings are not reset

How to connect a relay module correctly

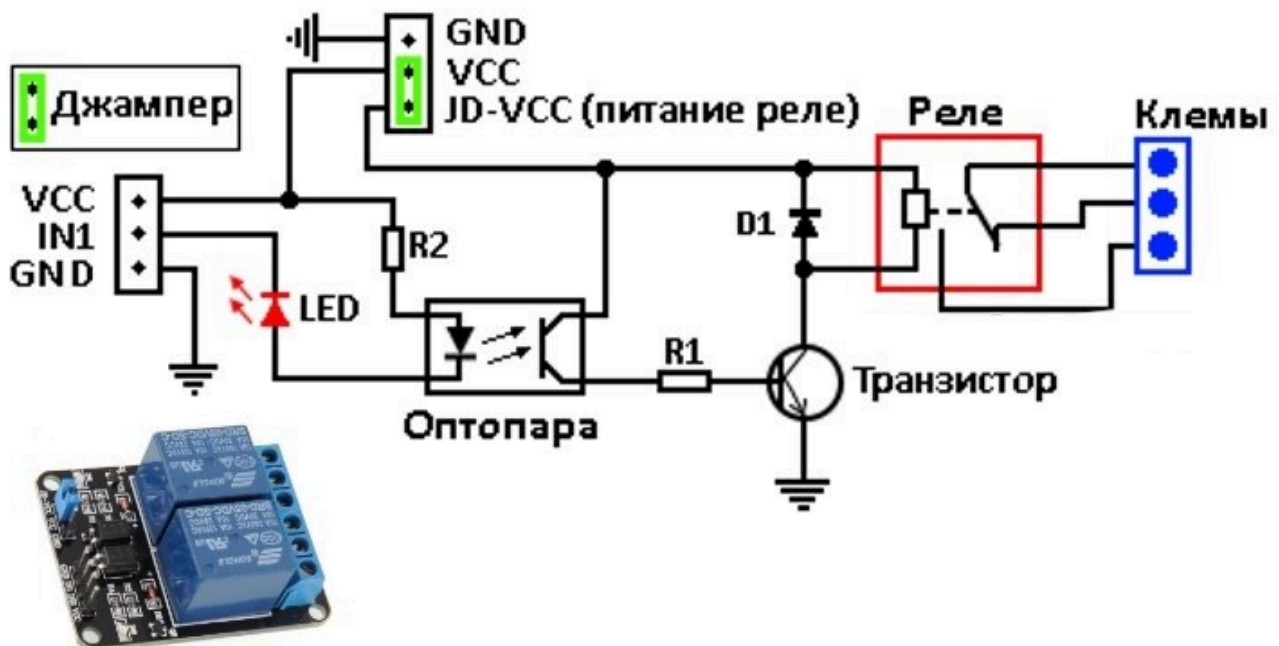
You can connect 5-volt or 12-volt relay modules to the ESP32. The module must have a JDVCC - VCC jumper.

Warning:

If the JDVCC-VCC jumper is missing, the module has no galvanic isolation of the power supply. Using such modules is not recommended.



Circuit diagram, using one channel as an example:



To connect to the ESP32:

The upper connector in the diagram:

REQUIRED! Remove the jumper!

- Apply the negative of the relay power supply (-5 or -12 V, depending on the relays used) to the GND pin.
- The VCC pin is not used (that is, it is not connected to anything).
- Apply the positive of the relay power supply (+5 or +12 V) to the JDVCC pin.

The second connector:

- Connect VCC to +3.3 volts from the ESP32
- Connect IN1, IN2 to the required ESP32 output. For full galvanic isolation, you can leave the GND pin of this connector unconnected to the ESP32 GND pin.

This way the whole module, including the relays, is powered from a separate power supply (5 or 12 volts), and only 3.3 volts comes from Samovar: to resistor R2, the optocoupler LED and the indicator LED.

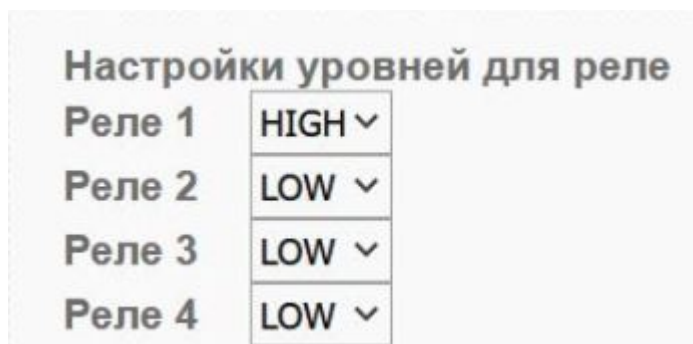
If the IN1 pin is at logic zero, the relay turns on; if it is at logic one (+3.3 volts), the relay turns off

Note.

Most relay modules are of the Low Level Trigger type (LLT, enabled by default). This means that to turn the relay on, the relay control input (IN1) must be at logic "0". When logic "1" is applied to the input (IN1), the relay turns off. Such relays are used when the microcontroller controls many devices, to reduce the microcontroller's heating and current consumption. They are also used in systems where the relay must be on when power is applied or "by default".

To control the module with logic "1" instead of logic "0":

- In the Samovar web interface, on the [Settings – Other](#) («Настройки – Прочие») tab, select the required module trigger level.



The **rele1...rele4** fields set the level at which the relay **turns on**. For the module described here, which turns on with logic zero, set the level to **LOW**.

Warning. On the ESP32 DevKit, relay No. 2 (stirrer) is connected to GPIO15. This is a strapping pin that the hardware pulls high when the board is reset. A module that turns on with logic one will briefly switch on on this channel at every Samovar boot. For relay No. 2, use a module that turns on with a low level (LOW).

Extended configuration

The extended configuration builds on the recommended one and adds hardware for specific modes and scenarios.

You can connect an [LCD display with an encoder](#), a [servo for fraction distribution](#), up to ten [I2CStepper](#) modules for the stirrer and the pump, an [emergency button](#), additional sensors, an [external meter](#) and [NBK](#) (continuous wash column) hardware. Connect only the parts that are enabled in the selected build environment and configured in **Samovar_ini.h** or **user_config_override.h** (the easiest way is through the configurator, which is started from **flash_windows.bat** or **flash_macos_linux.sh**).

Cheesemaking hardware

Samovar 7.00: the [Cheesemaking](#) mode uses the standard board outputs; no separate connectors are needed:

The pH sensor is a PH-4502C analog module. Its output is connected to the analog input LUA_PIN: GPIO34 on the ESP32 DevKit, GPIO4 on the ESP32-S3. The MPX5010DP pressure sensor and the Lua analogRead() function use the same input, so they cannot work together with the pH sensor. The readings are converted to pH using the calibration, see [pH calibration](#).

The stirrer uses relay No. 2 (as in Beer mode) or the working I2CStepper stirrer with the lowest available odd address; select it in the program line.

The drain valve is relay No. 4. In other modes this output controls the boost heating element, so connect either a valve or a heating element to it.

Cooling is a water pump or valve, as in the other modes; without USE_WATER_PUMP or USE_WATER_VALVE the cooling stage will not pass validation in the cheesemaking program.

Meter

The meter by dranek is a separate external device for additional measurements. It is not required for basic Samovar operation and is not part of the Samovar firmware: Samovar does not read its values.

The meter source code is in the Samovar repository, in the **pro_mini_ntc** directory (an ESP8266 sketch, a "Схема и плата" ("Schematic and board") folder with the circuit diagram and Gerber files, and a "Справочная информация" ("Reference information") folder with a description). This directory is not included in the Samovar firmware build.

[Description, schematic and discussion of the meter on the forum](#). When building it, use the circuit from the author's original post.

I2CStepper

I2CStepper is a standalone module based on an Arduino Nano with its own stepper motor, four relays, an external sensor, a 16×2 LCD and an encoder. The current firmware uses the I2C v3 protocol.

The module can be used as a standalone device. It can be controlled on its own using the menu on the 16x2 LCD screen and the encoder, or when connected to Samovar.

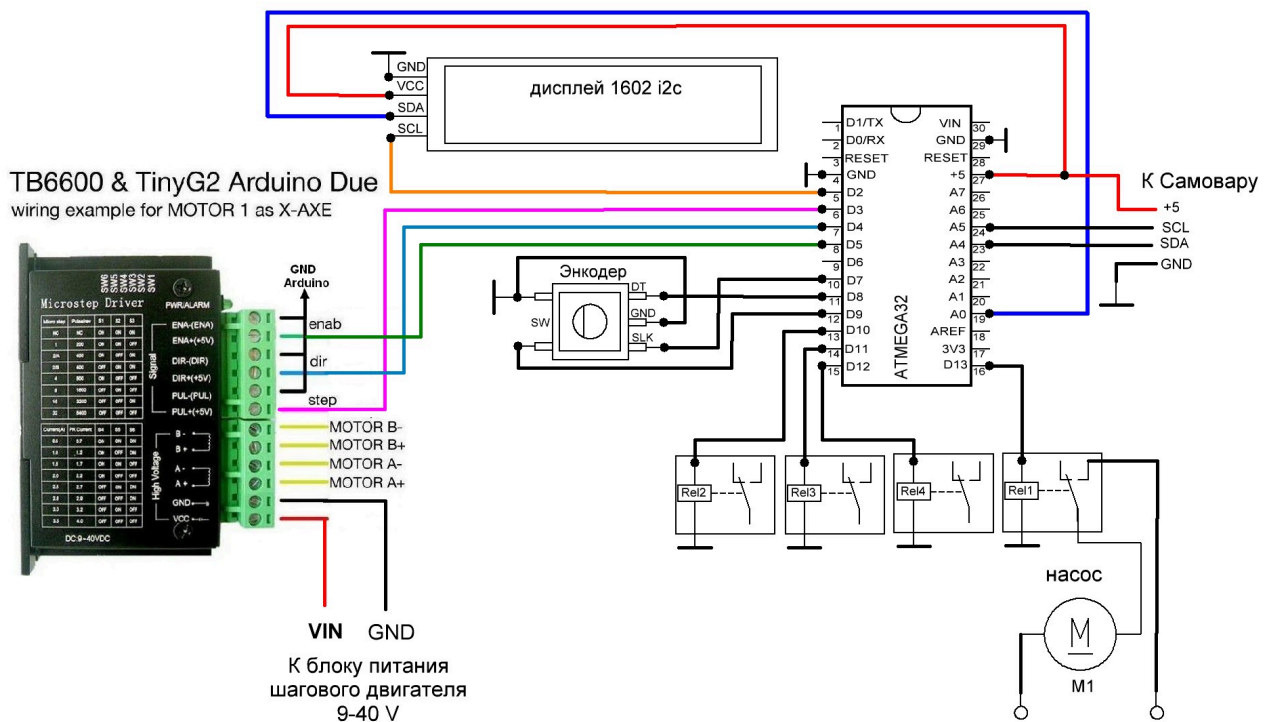
The expander is connected to Samovar over the I2C bus with four wires (like an LCD screen). Samovar detects that the expander is connected and can control it.

Samovar supports up to ten boards with addresses 1–10: an odd address is a stirrer, an even address is a pump/filling board. The boards with the lowest address of each type are the ones used. The stirrer works by speed, run time and pause; the pump works in ml/h; the filling mode works by a set volume and speed.

If the board supports relays, all four can be switched from the I2CStepper page and Lua. In Beer mode, relay 1 of the working board serves the wort transfer pump; the user assigns the other relays.

Boards are searched for once when Samovar boots; discover new boards or address changes with "Rescan I2C devices" («Пересканировать I2C-устройства») in the settings. If at least one board is available, the **"I2CStepper"** button appears on the main page of every mode. The control page sets speed (stirrer: rpm, pump: l/h), direction and pump volume in ml; it provides start, stop and four relays. Board settings are stored in the Nano and edited under Settings → I2CStepper («Настройки → I2CStepper»). Open calibration of the selected external pump with "External pump calibration" («Калибровка внешнего насоса»). See the I2CStepper page in the operating manual for the detailed procedure.

Wiring diagram



Pinout for connecting peripherals:

// Pins for I2C Master - connecting the screen and potentially other devices

// SDA_PIN 0 //A0

// SCL_PIN 2 //D2

// Pins for I2C Slave - communication with Samovar

// SDA_PIN A4

// SCL_PIN A5

// Pins for the stepper motor

#define STEPPER_STEP 3

#define STEPPER_DIR 4

#define STEPPER_EN 5

//Pins for the relays

#define MIXER_PUMP_PIN 13 // RELE_PIN1

#define RELE_PIN2 10 // RELE_PIN2

#define RELE_PIN3 11 // RELE_PIN3

#define RELE_PIN4 12 // RELE_PIN4

// Pins for the encoder

#define ENC_CLK 7 //S2

#define ENC_DT 8 //S1

```
#define ENC_SW 9 //KEY
```

Tip.

If you plan to use the expander as a separate device in its own enclosure, it makes sense to connect it to Samovar with a USB A-B cable (computer - printer). The module's +/- 5 volt power and the two I2C bus signals are taken directly from Samovar's display. A USB-B connector is mounted on the Samovar enclosure.

A USB-A connector is mounted on the expander enclosure.

This way, when connected to Samovar, the Arduino Nano gets both power and the I2C bus at once. When used as a standalone device, a separate power supply is used instead of Samovar, for example a phone charger.

If you use a built-in power supply, a USB-B connector is placed on the PSU output and the same USB A-B cable is used to deliver power, which rules out accidentally feeding +5 volts into Samovar.

Peripherals.

The relay block is used at 5 volts, preferably controlled through optocouplers. 12-volt relays can also be used, with a [connection](#) made on the same principle as in Samovar

Any pump can be used: a DC motor, or an AC single-phase or three-phase one. This is because the pump motor is started and stopped through a relay and then through any power switch: a contactor or a solid-state relay. For small volumes you can look at the 15RM magnetic drive pump. Stainless steel, quiet, with good output.

Any bipolar stepper motor can be used for the stirrer or the NBK (continuous wash column), provided it has enough power and torque. Nema 23 with a suitable driver is recommended.

The current Arduino Nano firmware and the description of the v3 protocol are in the [I2CStepper repository](#). Compatible v2 settings are migrated to v3 automatically; invalid saved data is replaced with defaults. After updating, check the address, mode and pump calibration.

Current modes:

- Mixer: stirrer, revolutions per minute, run time, pause, reverse after the pause and soft start.
- Pump: continuous feed in ml/h with an optional pause.

— Fill: metered filling of a set volume in ml.

The external sensor connects to D6; by default the input has a pull-up and is active when connected to GND. It can stop any active operation. In Pump mode, with sensor pause enabled, pumping resumes after the configured delay following sensor release. If the Nano receives no communication heartbeat from Samovar for more than 10 seconds during remote control, it stops the motor and turns off all relays; the local menu becomes available without separate encoder confirmation.

Warning.

*The Lua function **set_stepper_target(sp, direction, target)** controls only the I2CStepper pump (the working board with an even address): speed, direction and number of steps. If the pump does not respond on the bus, the function returns 0; it does not control Samovar's built-in stepper.*

*--set_stepper_target(sp, direction, target) //sp - speed in steps per second,
direction - forward or reverse direction,
target - number of steps*

When sp = 0 the stepper motor is stopped.

Example

set_stepper_target(950, 0, 1000)

The example sets speed 950, direction 0 and a move of 1000 steps.

The pH acidity sensor and how to connect it

In Cheesemaking mode the pH sensor lets Samovar wait for the set acidity at the "pH wait" («Ожидание pH») stage. The measuring kit consists of a submersible electrode and a PH-4502C board. The electrode connects to the BNC connector of the board; its analog output PO passes the voltage to Samovar. Do not connect the electrode itself directly to the ESP32 or ADS1115: it needs a signal-processing board.

The electrode must be suitable for the product being measured and its temperature. A sensor designed for water can become fouled with protein and fat in milk; whether it is suitable for continuous immersion is determined by the electrode's datasheet. Calibration does not remove fouling or wear.

Connection via ADS1115

The ADS1115 is an external ADC, a converter of voltage to a number. To connect it you need an ADS1115 module, two 10 kΩ resistors with 1% tolerance and a 100 nF capacitor. The values refer to the output of the PH-4502C board, not to the electrode cable.

- Power the PH-4502C with 5 V and the ADS1115 with 3.3 V. Join the GND grounds of both boards and of Samovar. Before connecting, check the pin markings of your board: the pin order differs between modules.
- Connect PO of the PH-4502C to AIN0 of the ADS1115 through the first 10 kΩ resistor. Place the second 10 kΩ resistor between AIN0 and GND. Connect the 100 nF capacitor in parallel with the second resistor. The divider halves the voltage: 5 V on PO becomes 2.5 V on AIN0. The DO output is not used for pH measurement.
- Connect SDA and SCL of the ADS1115 to the lines of the same name of the Samovar I2C — the common two-wire communication line with modules. Use the pins of your ESP32 board given in its schematic; GPIO numbers depend on the build. The SDA/SCL pull-ups must go to 3.3 V.
- Connect ADDR to GND: the ADS1115 address will be 0x48. If this address is already taken by another device, choose a free ADS1115 address and specify it in the build setting. Two devices with the same address must not work on one line.
- Enable the [USE ADS1115](#) build setting with the address value, for example [#define USE ADS1115 0x48](#), and upload the firmware. pH is read from AIN0 automatically. LUA_PIN stays free. If the ADS1115 is unavailable, starting cheesemaking and saving the pH calibration are blocked. Check the voltage between AIN0 and GND with a multimeter before connecting to the ESP32. The ADS1115 measurement range in the firmware is ±4.096 V, but it does not allow applying a voltage above the chip's supply to the input. After changing the circuit or switching to the ADS1115, calibrate again.

Connection without the ADS1115

If USE_ADS1115 is disabled, the output of the same divider is connected to LUA_PIN instead of AIN0. The PH-4502C is powered from 5 V; its GND is joined to the Samovar GND. You must not apply 5 V directly to an ESP32 input. LUA_PIN is also used by the MPX5010DP pressure sensor: connect only one device to it. During cheesemaking this input is occupied by the pH measurement and cannot be read from Lua.

If the readings jump when equipment is switched on

First compare the readings in a separate plastic cup with buffer and in the vat, switching on the pump, the stirrer and the heating one at a time. If the jumps are linked to switching equipment on, electrical interference is a likely cause. A slow drift of the readings in the buffer with the equipment off requires checking the electrode and the PH-4502C; isolation will not fix their fault.

To eliminate interference through the common power supply and ground, you can separate the measuring part: **electrode** → **PH-4502C** → **1:2 divider** → **ADS1115** → **ADuM1250** → **ESP32**. The ADuM1250 is an I2C electrical isolation module: it passes data without electrically connecting the two sides.

- . Power the PH-4502C, the ADS1115 and the measuring side of the ADuM1250 from a separate USB power bank with a 5 V output that supports low load. In this circuit the ADS1115 also gets 5 V; the divider at its input stays. During measurement the power bank must not be charging or powering other parts of Samovar.
- . Power the side of the ADuM1250 facing the ESP32 from the 3.3 V of Samovar. Connect SDA/SCL of the ADS1115 to the 5 V side, and SDA/SCL of Samovar to the 3.3 V side. The pull-up resistors of each side must be connected to the supply of that very side.
- . **Do not join the grounds of the two sides.** If you join them with a wire, a USB cable or through other equipment, the isolation is broken. The other I2C devices of Samovar stay on the ESP32 side.
- . Place the PH-4502C and the ADS1115 close together, away from the heating regulator. Keep the PO wire short and route the electrode cable away from power wires. The ADS1115 address and the USE_ADS1115 setting stay the same.

This is a connection option against electrical interference, not a guarantee of electrode accuracy. After calibration, repeat the check with equipment being switched on; the circuit must not give a reproducible pH shift on switching.

Software filter and limitations

In firmware with the median filter, the pH reading is calculated from five consecutive measurements taken one second apart. The median is the middle number in order after sorting: the series 5.42; 5.41; 3.10; 5.43; 5.42 gives 5.42. The filter removes short spikes but does not fix a constant error or a slow drift of the readings. With a smooth change the delay is about two seconds.

pH becomes available after all five samples are filled. After a read error, an invalid reading, a gap between measurements of more than five seconds, a change of the calibration coefficients or a process reset, the window is filled anew. An old result is not passed off as a new valid measurement. At the "pH wait" («Ожидание pH») stage, confirmation of reaching the threshold is kept for 30 seconds.

The filter applies to both connections — via the ADS1115 and via LUA_PIN. The ADC field on the calibration page shows a fresh raw sample, without this filter; wait for it to stabilize before fixing a point. Samovar does not take temperature into account in the pH conversion: compare results and control samples at the same temperature. Updating the documentation on the website does not update the firmware on your device.

Calibration and care

Calibrate the whole assembled kit with two fresh buffer solutions at about pH 7 and pH 4. Enter the values from the solution label for the measurement temperature. The procedure with the buttons is described in the section [pH calibration](#).

After milk, clean the electrode with a compatible cleaner for protein and fat deposits, following the manufacturer's instructions. Store the sensing end in the recommended electrode storage solution, not dry and not in distilled water. Do not exceed the electrode's permitted temperature. After cleaning, check the readings in a fresh portion of buffer.

Reference materials: [ADS1115](#), [ADuM1250](#), [electrode care](#), [causes of electrical interference in pH measurement](#).

Additional features

Additional functions can be implemented with Lua and PCF8575/PCF8591 expanders over I2C. Lua is enabled by the separate **Samovar_lua** build environment in PlatformIO, by the "Use Lua" («Использовать Lua») checkbox on the "Hardware" («Оборудование») tab of the configurator, or by the **#define USE_LUA** directive in **user_config_override.h**; the expanders are enabled in the same place with the **USE_EXPANDER** and **USE_ANALOG_EXPANDER** directives (in the configurator, the "PCF8575 expander address" («Адрес расширителя PCF8575») and "PCF8591 expander address" («Адрес расширителя PCF8591») fields).

Programming of additional functions is implemented on the basis of the [scripting language Lua](#). The script editor is available on the Settings – Editor page ("Settings" («Настройки») – "Editor" («Редактор»)).

You can attach any actuators or buttons (sensors that work as buttons) to the port expander, and decide for yourself what should happen at a given moment while Samovar is running.

The **init.lua** file runs when Lua is initialized. While Samovar is working, the common **script.lua** and the mode script selected in the settings run about once a second. You can change them in the file editor of the web interface.

You can store variables in the Samovar memory, run up to 10 timers, get the Samovar operation status, and so on. That is, you can connect additional equipment and control it automatically while Samovar is working

You can check that a safety sensor has not tripped, and if it has, switch Samovar off right from the script, or increase or stop the water supply. There can be many scenarios. They depend on the specific equipment.

The basic Samovar functionality does not change.

Scripts can use one analog input of the board — **LUA_PIN** (GPIO34 on ESP32 DevKit, GPIO4 on ESP32-S3), read by the `analogRead()` function. **Samovar 7.00**: in [Cheesemaking](#) mode this input is used by the pH sensor, and accessing it from a script ends with an error. The relay, pump and button outputs cannot be switched from a script — only through the port expander or the standard Samovar functions.

Bluetooth port monitor

The Samovar firmware does not use the built-in Bluetooth of the ESP32 module. You can build a remote serial port monitor with external HC-05 or HC-06 modules: they simply relay over the radio channel whatever Samovar prints to the main diagnostic port (the same one you see in the Arduino IDE or PlatformIO port monitor). The port runs at **115200 baud**.

Connect the RX input of the Bluetooth module to the TX of the main serial port (the TX0 pin of the board), join the GND lines, and set the module to 115200 baud. Choose the supply voltage according to the markings of your particular module board; the ESP32 signal lines are 3.3 V. The module's TX output does not need to go to the Samovar RX if you only want to view the log. Leave UART2 with the RXD2/TXD2 pins (GPIO16/GPIO17 on ESP32 DevKit, GPIO18/GPIO17 on ESP32-S3) to the power regulator — it is a different port.

The output of the **print()** function of Lua scripts goes to the same port. The **Lua_UART** folder of the project sources contains an example of a separate Arduino receiver: it receives a string over UART, splits it by commas into 8 parts (up to 20 characters each), and shows them on a display. Do not connect the module directly to mains circuits or to the regulator outputs.