

SAMOVAR

技术文档

固件 7.00

最新版本: <https://www.samovar-tool.ru/doc/zh/device/>
生成于 01.10.2026

目录

内部说明。请先阅读 :)

从哪里开始

使用 PlatformIO 编译和烧录

ESP32 的安装

最小配置

蜂鸣器

大气压力传感器

温度传感器

使用的调压器

调压器概览

如何将调压器连接到 Samovar

加热管

使用辅助加热管

接触器。漏电保护器。紧急断电

冷却

使用隔膜泵

使用离心泵

不用水泵

回流液位传感器

驱动器。步进电机

分馏分配

紧急按钮。附加紧急传感器

推荐配置

LCD 显示屏、编码器、按钮

如何正确连接继电器模块

扩展配置

测量仪

I2CStepper

pH 传感器及其连接

附加功能

蓝牙串口监视器

内部说明。请先阅读:)

供帮助补充和修正本组装指南的人员参考的内部信息。

本信息 面向参与 编写本指南的人员。

发送材料以及就本 指南进行沟通的地址：device@samovar-tool.ru

请将 文字以 .doc 或 .txt 格式、图片以 .jpg 格式发送。文字与照片分开发送。也可以全部打包成一个压缩文件。

本指南以总帮助为基础编写。如果发现失效的链接，请修正它，或替换为指向相应现有章节的链接。

- 如果某个章节“放错了位置”，这不是问题——在目录树中移动它非常容易。所以暂时不必在意这一点。

建议如下：

随着项目发展，指南的结构可能会调整。

每个章节应尽可能多地包含某一具体硬件的信息：

照片、购买链接、原始名称（以便链接失效后还能找到）、连接方法、特点、可替代的型号、改装方法、等等。

非常希望附上链接，指向论坛上讨论该设备的帖子——您自己的帖子很容易找到，而别人却得把整个论坛翻一遍，还不止一遍.....。

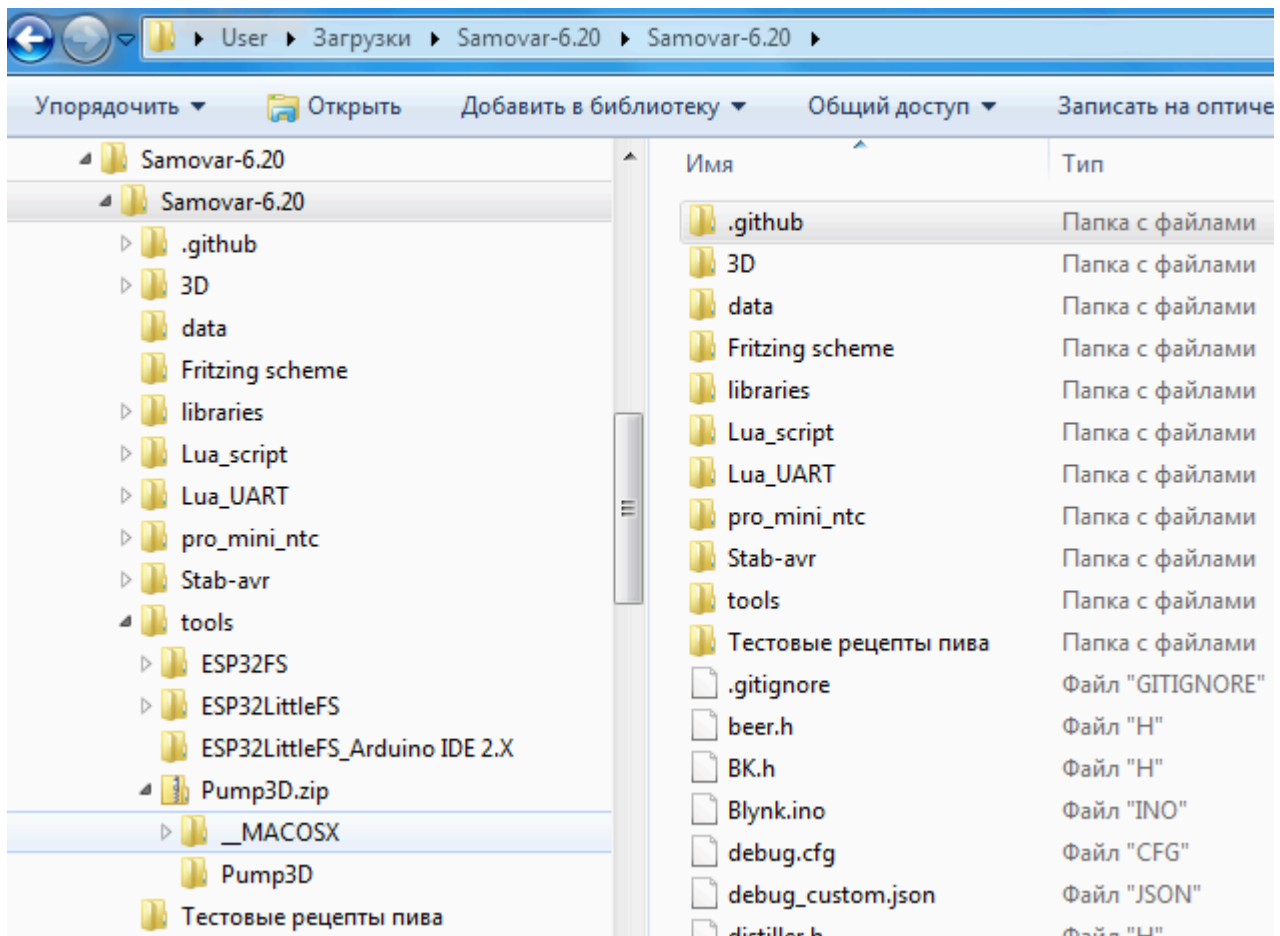
也就是说，一个具体设备一个章节；如果章节太长，很容易再分出子章节。（例如关于[调压器的章节。](#)）其他章节中不放某一具体设备的补充信息。提及可以随意，只需加一个指向该设备章节的链接即可。否则在整理时 很容易弄乱。

也可以有例外——例如蒸馏釜中的压力传感器。专门接到 Samovar 上的 MPX5010D 的实现说明从一处开始，在 dranek 测量仪一节中继续。

署名。如有意愿，可在章节末尾附上材料作者的链接。

从哪里开始

首先，下载 Samovar 最新版本的[发布包](#)，在电脑上解压并浏览其中的内容。



各文件夹的内容：

注意

请先阅读项目根目录中的 **README.md** 文件：其中列出了支持的开发板、引脚限制和安装步骤。

- .github - 软件文件夹。
- 3D - 用于 3D 打印的模型。分馏收集转台、漏斗等。
- **data_raw** — 网页界面源文件。**data** 文件夹是设备文件系统的现成镜像；它完全由命令 **python3 tools/build_web_assets.py** 重新生成，因此不要手动编辑。program_*.txt 文件包含默认程序。
- Fritzing scheme - 组装电路图。
- libraries - 软件文件夹（库）。
- Lua_script - Lua 脚本语言的说明以及各种非常有用的脚本。
- Lua_UART - 如何制作独立的远程 COM 口监视器。
- [pro_mini_ntc](#) - 如何制作 Samovar 的附加装置（测量仪）；它也可以作为独立设备、作为测量仪器工作。
- [Stab-avr](#) - 用现成组件制作、可由 Samovar 控制的功率稳压器方案。

- **tools** — 配置器 (samovar_configurator.py)、网页界面构建脚本和项目自动检查。
- doc - 固件各子系统的说明 (Markdown 文件)。
- **partitions** 和 **ESP32-S3** — ESP32 和 ESP32-S3 的闪存分区表；PlatformIO 会用到它们。
- Stab-avr、Lua_UART、pro_mini_ntc - 独立的 Arduino 草图 (sketch)，不参与 Samovar 的编译。
- Test beer recipes - 几个可以加载到 Samovar 中的现成啤酒配方。

Samovar 最少需要什么配置？

首次启动只需要一块 ESP32 开发板和 USB 供电。固件既可以[在 Arduino IDE 中](#)编译，也可以在 PlatformIO 中编译。硬件设置在 **Samovar_ini.h** 中；更新时需要保留的用户覆盖设置在 **user_config_override.h** 中。发布包中不包含现成的二进制文件。

详细说明：[在 Arduino IDE 中编译](#)和[使用 PlatformIO 编译和烧录](#)。

注：

Samovar 不检查是否存在的传感器和执行机构（因为没有反馈）：

- 接触器 / 调压器通电 (1 号继电器)
- 附加加热器继电器 / 调压器升温 (4 号继电器)
- 蜂鸣器
- 开水阀 (3 号继电器)
- 回流液位传感器
- 蠕动泵步进电机
- 分馏收集转台舵机
- 冷却水泵
- 大气压力传感器
- 蒸馏釜压力传感器。
- 搅拌器 (2 号继电器)

Samovar 会检查是否存在的传感器和执行机构（在以下条件下）：

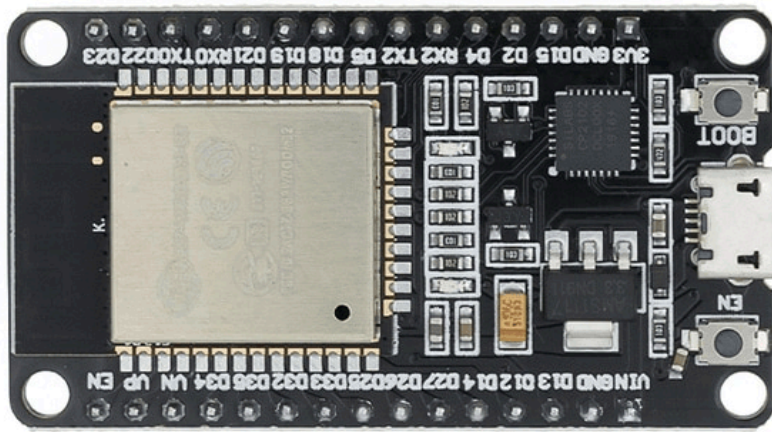
- 温度传感器 - 如果过程开始时传感器存在。传感器丢失时会定期发出消息。
- 水流量传感器 – 如果启用了它。
- 加热调压器 – 如果启用了调压器使用检查。

然后[设置 Wi-Fi 连接](#)并打开 Samovar 的网页界面。在经典的 ESP32 DevKit 上，“开启/关闭加热”（«Включить/Выключить нагрев»）命令会切换 GPIO2，许多开发板上该引脚接着板载蓝色 LED。在 ESP32-S3 上是否有这样的 LED 取决于具体开发板。

一篇介绍 [ESP32 的科普文章](#)

说明。

ESP32 是一颗安装在开发板上的芯片。开发板上另外还有电压转换器、Wi-Fi 模块、蓝牙，可能还有更多部件。这里所说的 ESP32 指的是装有该芯片的开发板。



ESP-32 30P

开发板有很多种。

本项目的主开发板是 **ESP32 DevKit V1 30 pin**，PlatformIO 环境为 **Samovar**. **LILYGO T-Relay 4**（同样是 **Samovar** 环境，开发板在配置器中选择）和 **ESP32-S3 DevKitC-1**（环境 **Samovar_s3**）也受支持。
LILYGO 和 ESP32-S3 的引脚定义不同（见 `Samovar_pin.h`）；为 30 针 DevKit 设计的 Samovar 1.3.2 印制电路板不适用于它们。

ESP32 是芯片

DevKit V1 - 版本

30 pin - 开发板上的触点（脚、引脚）数量

CH340C 是 USB 总线数据传输转换芯片

Type-C - USB 接口的规格

在哪里购买。

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

[Ozon](#) (搜索：ESP32 WROOM-32 开发板 WiFi 蓝牙兼容 IOT 开发板 («ESP32 WROOM-32 Плата для разработки WiFi Bluetooth-совместимая плата для разработки IOT»))

如果不确定 ESP32 型号选得对不对，请到论坛上询问。

现在需要把 Samovar 与设备连接起来。

首先确定[ESP32 的安装方式](#)。

然后确定设备的布局方案：是做成一个整体，还是做成几个相互连接的独立单元。

最后，从[最小配置](#)开始组装，逐步添加所需的部件，直到实现 最远大的目标。

问题与解答。FAQ

*** 能否使用ESP32 DevKit V1 38 pin 这种 38 针的开发板？**

可以，但它无法安装在专为以下型号设计的印制电路板上：**ESP32 30 pin.**

安装 PlatformIO 并首次烧录 Samovar

Samovar 支持两种完整的安装方式：Arduino IDE 和基于 PlatformIO 的配置器。本页介绍 PlatformIO。

使用 Arduino IDE 时，请参阅[“准备 ESP32 以便上传”一节](#)。

配置器可以选择开发板和设备、保存设置、烧录 ESP32、单独上传 LittleFS，以及打开串口监视器。

PlatformIO 是一个编译源代码并把它写入开发板的程序。

Windows 10/11 (仅限 64 位)：下载[最新的 Samovar 发布版](#)，把压缩包解压到一个简单的路径，例如 `C:\Samovar`，用能传输数据的 USB 线连接 ESP32，然后双击 `flash_windows.bat`。该批处理文件会检查 Python 和 PlatformIO，必要时自动安装，然后打开设置窗口。

macOS 和 Linux：在解压后的项目文件夹中打开终端并执行：

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

该脚本会检查带 Tkinter 的 Python 和 PlatformIO，必要时自动安装，然后打开同一个配置器。

如果 macOS 提示 Operation not permitted 或 getcwd: cannot access parent directories：把项目文件夹从“下载”移到您的个人文件夹，在那里运行脚本。或者在“系统设置 → 隐私与安全性 → 文件和文件夹”中允许终端访问“下载”文件夹，然后彻底重启终端。

在 macOS 上，自动安装 Python 和 Tkinter 需要 Homebrew。在 Linux 上支持 apt、dnf 和 pacman；系统可能会要求输入 sudo 密码——即电脑的管理员密码。

使用配置器：选择 ESP32 DevKit、LILYGO 或 ESP32-S3，设置设备，必要时填写初始的 Wi-Fi 连接信息。其余参数分布在“主要”（«Основные»）、“温度”（«Температуры»）、“调压器”（«Регулятор»）、“BK”（«БК»）、“NBK”（«НБК»）、“传感器”（«Датчики»）、“水泵”（«Насосы»）、“设备”

（«Оборудование»）、“步进电机”（«Шаговый двигатель»）、“精馏”（«Ректификация»）、“网络”（«Сеть»）选项卡中——这些就是以前在 `Samovar.ini.h` 中编辑的那些指令。点击“保存设置”（«Сохранить настройки»），然后点击“烧录”（«Прошить»）。设置保存在项目旁边的文件

`user_config_override.h` 中，更新源代码时不会被覆盖。编译和烧录日志显示在窗口底部。

对于新开发板，请另外点击“上传 LittleFS”（«Загрузить LittleFS»）。LittleFS 是存放网页界面和用户文件的存储区。重新上传可能会替换程序、Lua 文件和日志，因此配置器会要求确认。重新上传前，请通过 Samovar 的文件编辑器保存您需要的用户文件。

“串口监视器”（«Монитор порта»）按钮打开一个单独的诊断窗口（速率 115200）。如果烧录长时间停在 Connecting...，请按住 BOOT，短按 RESET/EN，开始写入后松开 BOOT。

通过 Wi-Fi 更新。以后的固件更新可以不用数据线。点击“端口或地址”（«Порт или адрес»）字段旁的“刷新”（«Обновить»）：列表中除了 COM 口，还会列出在网络中找到的 Samovar 设备，以及在串口监视器中点击“获取 IP”（«Получить IP»）按钮后设备报告的地址。也可以手动输入地址（例如 192.168.1.37 或 samovar.local）。选择该设备而不是 COM 口，然后点击同样的“烧录”或“上传 LittleFS”。条件：设备上的固件编译时启用了“允许通过 Wi-Fi 更新”（«Разрешить обновление по Wi-Fi»）（“网络”选项卡），电脑与 Samovar 在同一网络中，并且防火墙允许 python.exe 的传入连接。完全擦除闪存、串口监视器以及“重启 ESP”（«Перезагрузить ESP»）只能通过 USB 进行。

“文件编辑器”（«Редактор файлов»）按钮直接在配置器中打开设备上的文件（程序、Lua 脚本、网页界面页面），而“在浏览器中打开”（«Открыть в браузере»）则打开 Samovar 的网页界面。

通过 PlatformIO 手动操作：

```
pio run -e Samovar -t upload - ESP32 DevKit 或 LILYGO；
```

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

```
pio run -e <environment> -t uploadfs - 上传 LittleFS；
```

```
pio run -e <environment> -t monitor - 打开串口监视器。
```

使用其他调压器的现成环境：`Samovar_rmvk`（RMVK）、`Samovar_sem`（SEM_AVR）、

`Samovar_no_power`（无调压器，带 UART）；`Samovar_lua` - 带 Lua，`Samovar_alarm_button` - 带紧急按钮。手动编译时的用户参数按 `user_config_override.example.h` 中的示例写在 `user_config_override.h` 中。

首次烧录后，请继续[将 Samovar 连接到 Wi-Fi](#)。如果配置器中已经设置了网络信息，而开发板内存中还没有保存其他网络，固件会自动尝试连接该网络。

ESP32 的安装

ESP32 上要连接很多外围设备，所以需要给 ESP32 安排位置并组织好各路连接。

重要。下面给出的引脚定义和 1.3.2 版印制电路板适用于 ESP32 DevKit V1 30 pin。对于 LILYGO T-Relay 4（环境 **Samovar**，开发板在配置器中选择）和 ESP32-S3（环境 **Samovar_s3**），请使用 [Samovar_pin.h](#) 中的引脚定义；1.3.2 版电路板在物理上不适用于它们。

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		30	V5	Вход. Питание 5В
Общий, масса	GND	2		29	GND	Общий, масса
Мешалка ← (Реле №2) ← IN2	D15	3		28	D13	IN4 (Реле №4) → Разгон
Контактор (PM2) ← (Реле №1) ← IN1	D2	4		27	D12	BUZ1 → Бипер,звук
Управление охлаждением ← PUMP	D4	5		26	D14	IN3 (Реле №3) → Охлаждение (вкл/выкл)
Регулятор напряжения ← UART RX	D16	6		25	D27	Level → Датчик уровня флегмы
Регулятор напряжения ← UART TX	D17	7		24	D26	STEP - Драйвер ШД
Датчики температуры ← DS18B20	D5	8		23	D25	SERVO1 → Сервопривод
Энкодер ← S1 ← ENC1	D18	9		22	D33	EN – Драйвер ШД
Энкодер ← S2 ← ENC1	D19	10		21	D32	DIR – Драйвер ШД
Дисп+ давл ← SDL_LCD+BME680	D21	11		20	D35	ALARM_BTN → Аварийная кнопка
Системный, UART_0: RX	GPIO3	12		19	D34	нет → LUA_PIN
Системный, UART_0: TX	GPIO1	13		18	D39	BTN – Универсальная кнопка
Дисп+ давл ← SDA_LCD+BME680	D22	14		17	D36	WATERSENSOR → Датчик потока ОЖ
Кнопка энкодера ← KEY(ENC1)	D23	15		16	EN	Системный Сброс [LOW]

ESP32 引脚的限制。GPIO34–39 引脚只能作输入，且没有内部上拉：紧急按钮（D35）、通用按钮（D39）和流量传感器（D36）必须外接 10 kΩ 电阻到 3.3 V，否则会误触发，上电后立即紧急停机。印制电路板上已预留了这些电阻。流量传感器的导线要与加热管电源线分开走线。D15 引脚（2 号继电器，搅拌器）在复位时被硬件上拉——接在它上面的继电器每次启动时都会短暂吸合；请使用低电平触发的模块，继电器极性在 Samovar 设置中设定（继电器 1–4 字段）。I2C 总线（显示屏、压力传感器、I2CStepper）的上拉电阻为 4.7–10 kΩ。

使用印制电路板。

为了在 Samovar 中使用 ESP32_30pin，专门设计了一块印制电路板。板上安装：

- ESP32 本身、
- 12 伏转 5 伏电源转换器、
- 步进电机驱动器、
- 蜂鸣器，
- 所有电容和 电阻都必须安装。推荐值：

电容：

C1、C2 - 电解电容 $500\ \mu\text{F} \times 25\ \text{V}$ (或 1000×25)

C3、C4 - 电解电容 $500\ \mu\text{F} \times 16\ \text{V}$ (或 1000×16)

注意

推荐值留有很大的电压余量。有了电压余量，即使质量不太好的电解电容也能用得更久、工作更可靠。

但如果您愿意并且对元件有把握，也可以：

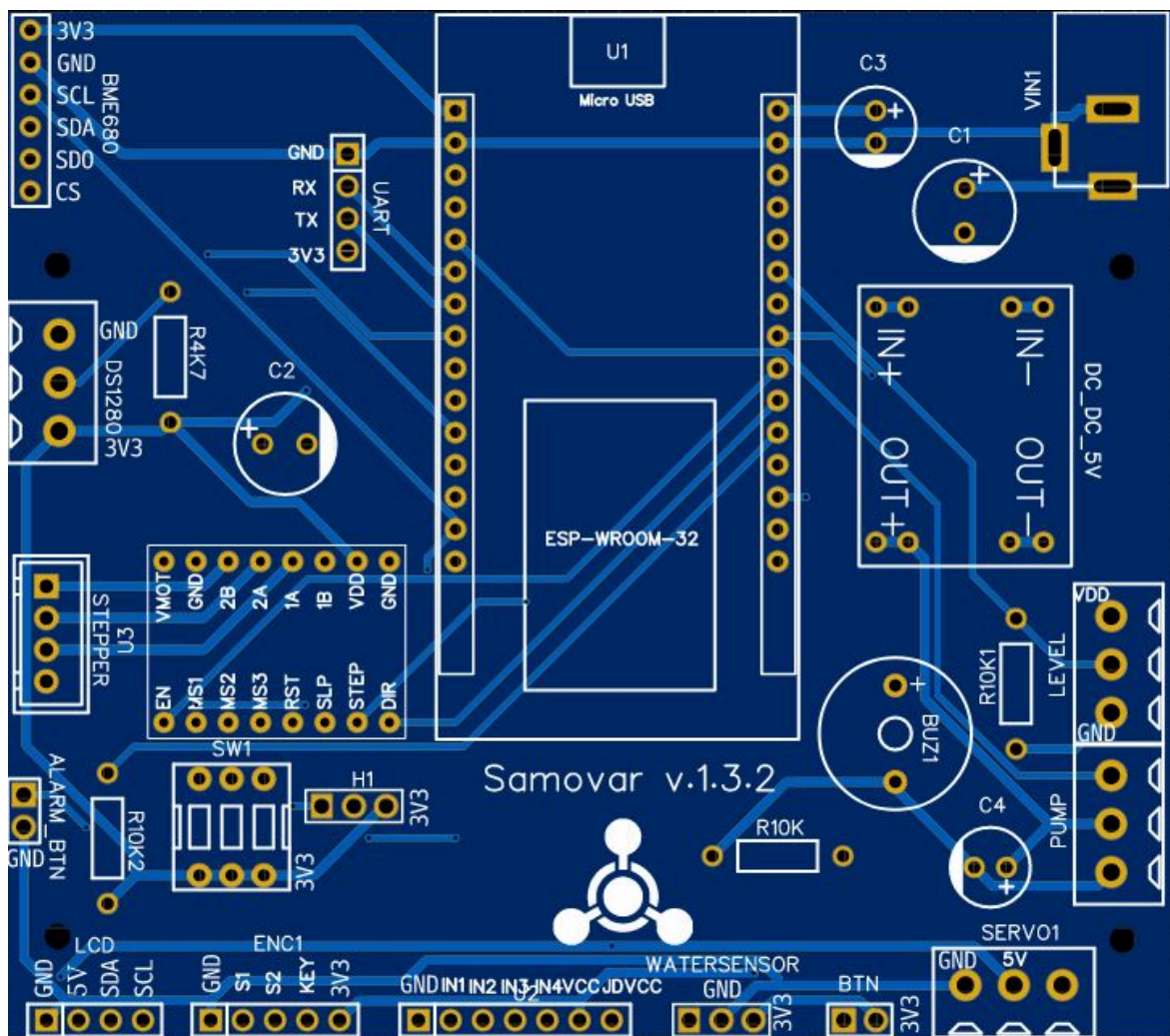
C1、C2 选 16 伏

C3、C4 - 选 6.3 伏。

电阻：R1-R3 = $10\ \text{k}\Omega$ ，R4 = $4.7\ \text{k}\Omega$ 。全部为 0.125 瓦。

• 连接用接插件。（注：部分接插件可以不装，直接焊接导线。）

建议一开始就把所有元件装到板上：蜂鸣器、大气压力传感器、电阻和电容、步进驱动器的排针、拨码开关和接插件。即使暂时不打算使用某些设备也一样。以后扩展 Samovar 的功能会容易得多。



安装位置和元件：

- ESP-WROOM-32 是安装 [ESP32](#) 的位置。建议装在排母 (female) 插座上
- [BME680](#) 连接大气压力传感器 (BMP180、BMP085、BMP280、BME280、BME680) 的位置。
- [DS18B20](#) 连接温度传感器。右侧 (板上) 是传感器的 4.7 kΩ 电源上拉电阻。
- [STEPPER](#) - 连接蠕动取酒泵的 NEMA17 步进电机。右侧是步进驱动器的位置。
- [SW1](#) 和 [H1](#) - 步进细分设置和步进驱动器的附加控制
- [ALRM_BT](#) 连接紧急按钮和报警传感器。
- [LCD](#) - 连接 LCD2004 显示屏 (不支持其他显示屏)
- [ENC1](#) - 带按钮的编码器接口。
- [IN1IN2IN3IN4](#) 连接继电器模块。
- WATERSENSOR - 冷却水流量传感器接口
- [BTN](#) - 多功能按钮
- [SERVO1](#) - 连接分馏转台的舵机
- [PUMP](#) - 冷却水泵控制
- [LEVEL](#) - 连接回流液位传感器
- [BUZ1](#) - 3 伏有源蜂鸣器
- DC_DC_5V - 12 伏转 5 伏电压转换器。
- VIN1 - 12 伏接口
- C1、C2 - 电解电容 500 μF × 25 V (或 1000 × 25)
- C3、C4 - 电解电容 500 μF × 16 V (或 1000 × 16)
- 所有电阻均为 10 kΩ，只有一个为 4.7 kΩ

到哪里获得电路板？

- 在支持帖中询问。也许已经做过的人有多余的板子，可以分给您。
- 自己下单制作。

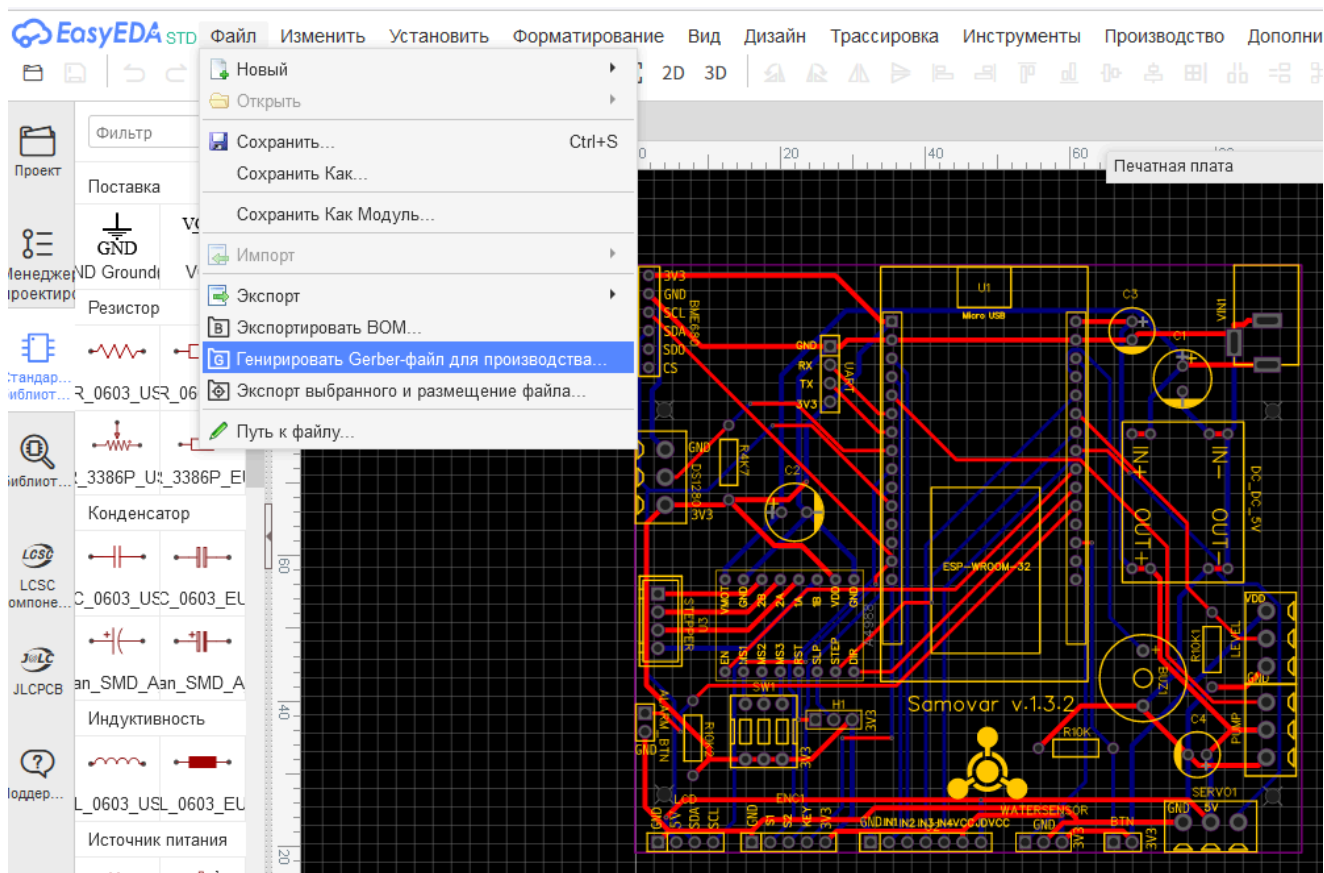
设计好的电路板托管在 easyeda.com。目前无法通过该服务下单生产，但可以使用该服务。

生产需要 Gerber 格式的文件，它是用于生产的印制电路板设计描述。

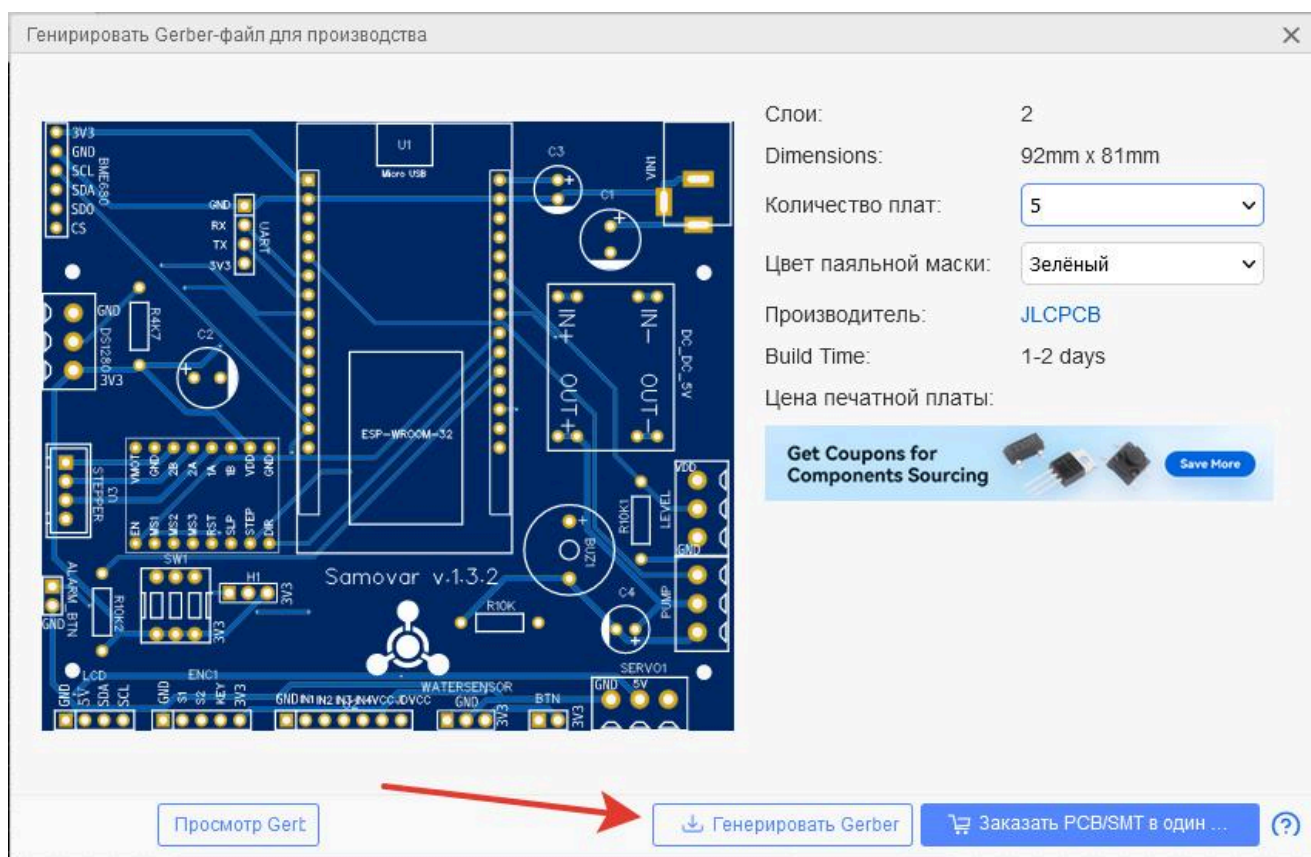
Gerber 压缩包 (用于生产 1.3.2 版印制电路板，截至 2024 年 9 月) 可以[通过此链接](#)从我们的服务器下载。

或者：

依次进入菜单：File>Generate Gerber file for production... (在俄语版 easyeda 界面中为：«Файл>Генерировать Gerber-файл для производства...»)



点击后，在检查提示中选择：“No, generate Gerber file”（«Нет, генерировать Gerber-файл»），会弹出一个窗口



可以修改：

- 电路板数量 - 5 的倍数（少于 5 块无法下单）。
 - 阻焊颜色 - 也就是电路板的颜色。选绿色或蓝色，其他颜色可能要加价。
- 点击 generate Gerber。生成的文件会以 zip 压缩包的形式保存到您的电脑上。现在需要下单生产。

能生产并且（最重要的是）能发货的服务很多。例如 nextpcb.com 网站上或网上都有操作说明，可通过 AliExpress 付款。

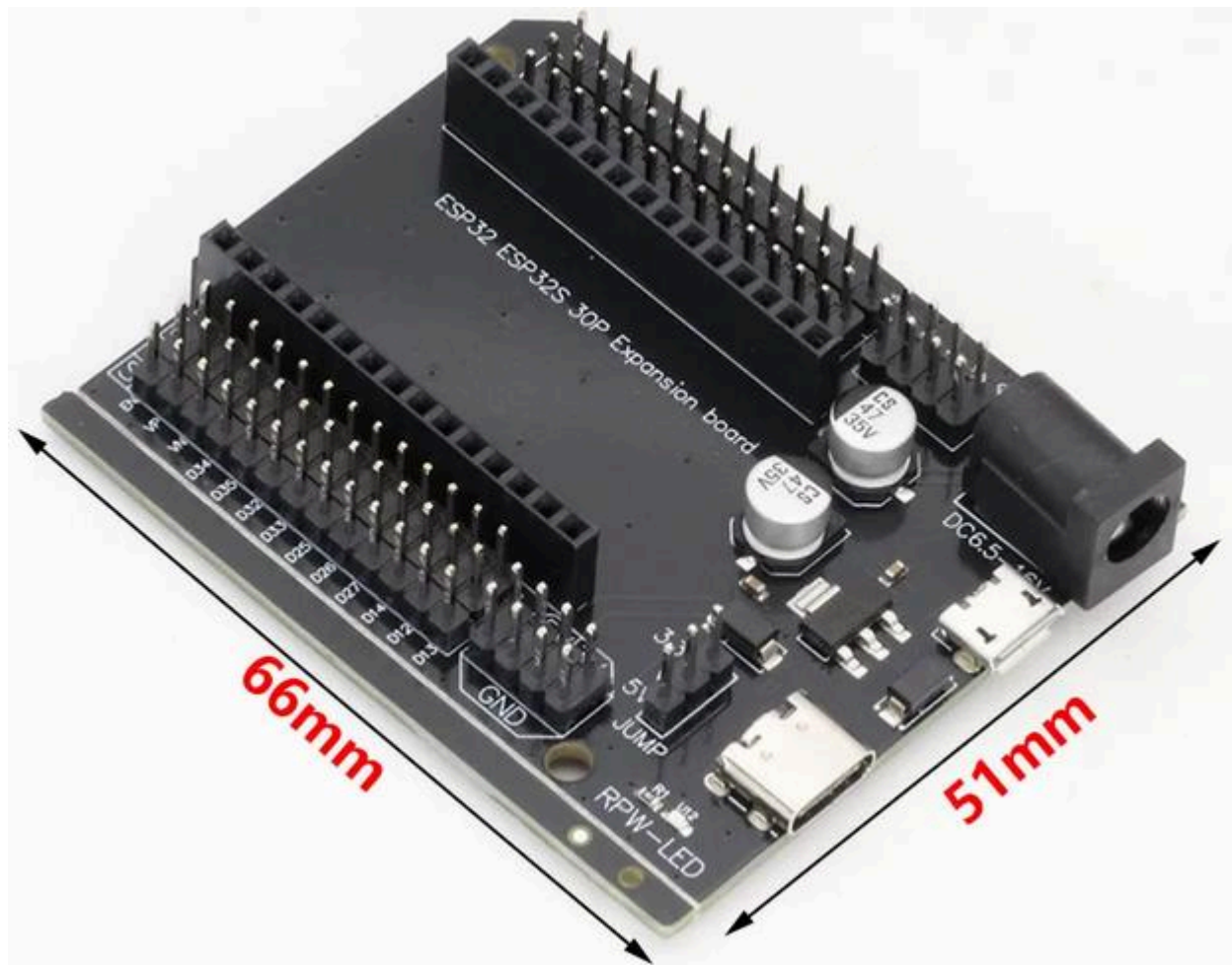
或者直接在 AliExpress 上搜索 “custom printed circuit board manufacturing”（定制印制电路板生产），会出现卖家的报价。请按评价选择。

下单方式相同 - 把整个 Gerber 压缩包发给卖家（如果文件名是西里尔字母 - 改成拉丁字母）。卖家确定价格并告诉您。如果合适就付款 - 大约一个月后就能拿到印制电路板。

5 块板含运费大约 15-20 USD。

使用扩展板 (shield)

ESP32 有一种专用扩展板 ([ESP32 Shield 30Pin breadboard expansion board](#))，可以快速搭建各种设备的原型。它的样子如下：



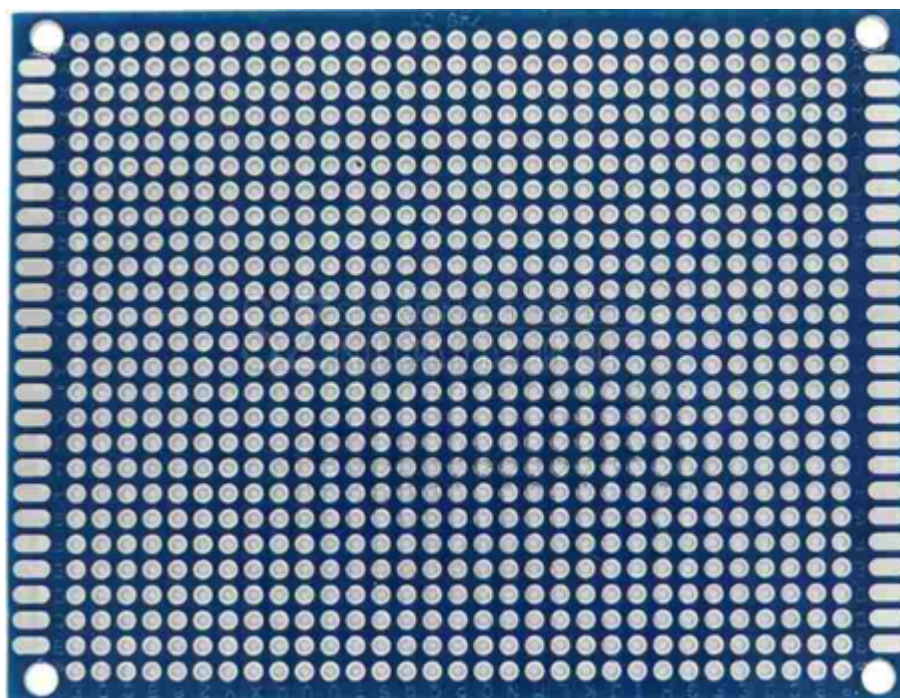
扩展板上已经有 16...6.5-5v 电源转换器和 USB 接口 - 5 伏供电。ESP32 插在排母上，外设接到排针（male）上。所有连线都要用导线完成；电阻和部分电容焊在扩展板上，或焊在外壳的接插件上。

使用扩展板是一个不错的方案，而且相当便宜。但在最终组装时，强烈建议把所有导线都焊牢，而不是留在可拔插的连接上，否则故障会接二连三。

也有用于 ESP32_38 pin 开发板的扩展板，因此 38 针的 ESP32 也可以使用。

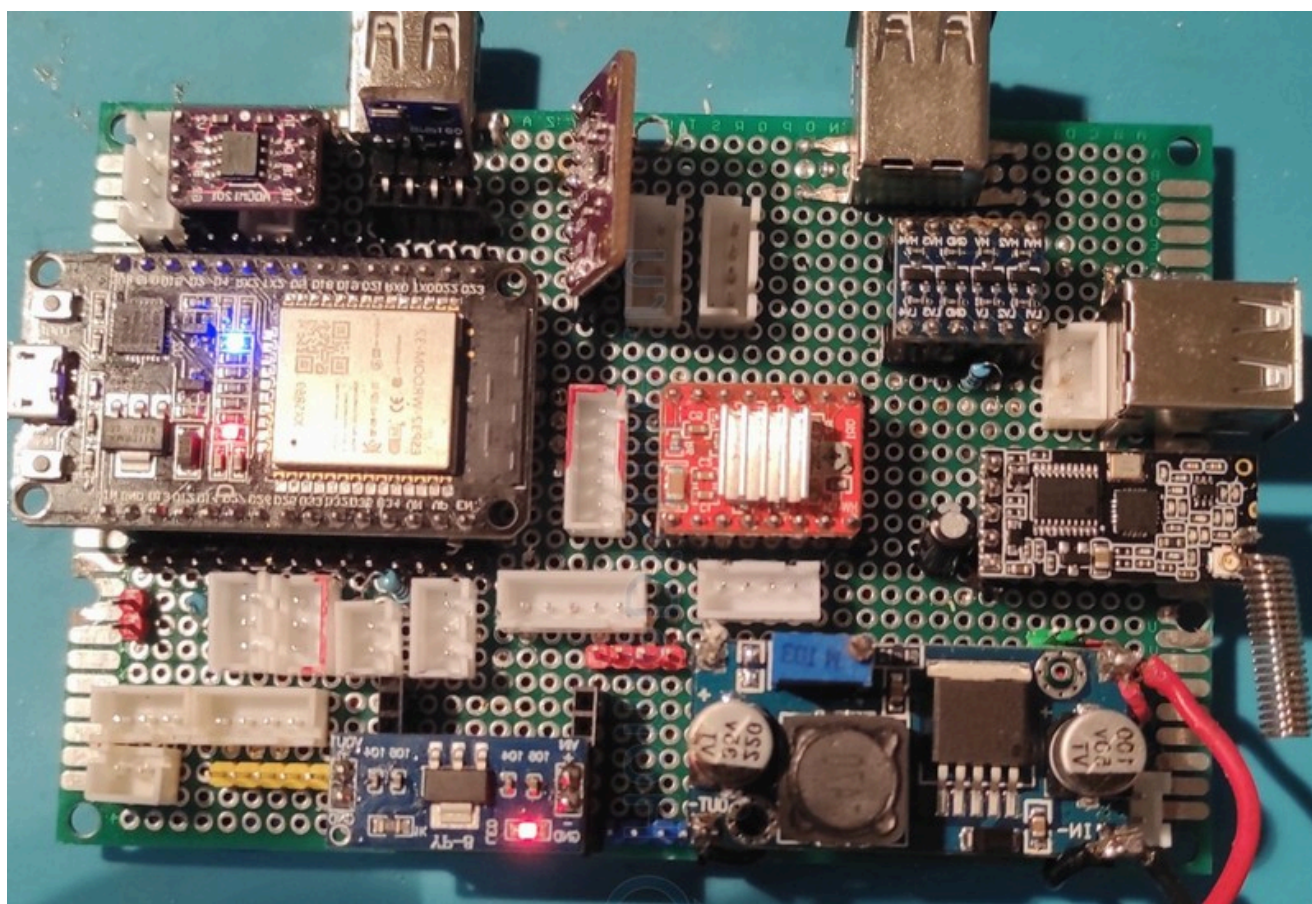
使用焊接用万能板。

可以使用专为焊接元件设计的印制原型板。（印制电路板原型、面包板、万能板）



所有元件都可以放在这样的板子上，甚至还能留出今后升级的空间。但使用这种板子需要一定的技能。提醒一句——在这种板子上组装 Samovar 并不轻松。尽管如此，项目仍然可以这样实现。

在这种万能板上组装 Samovar 的示例：



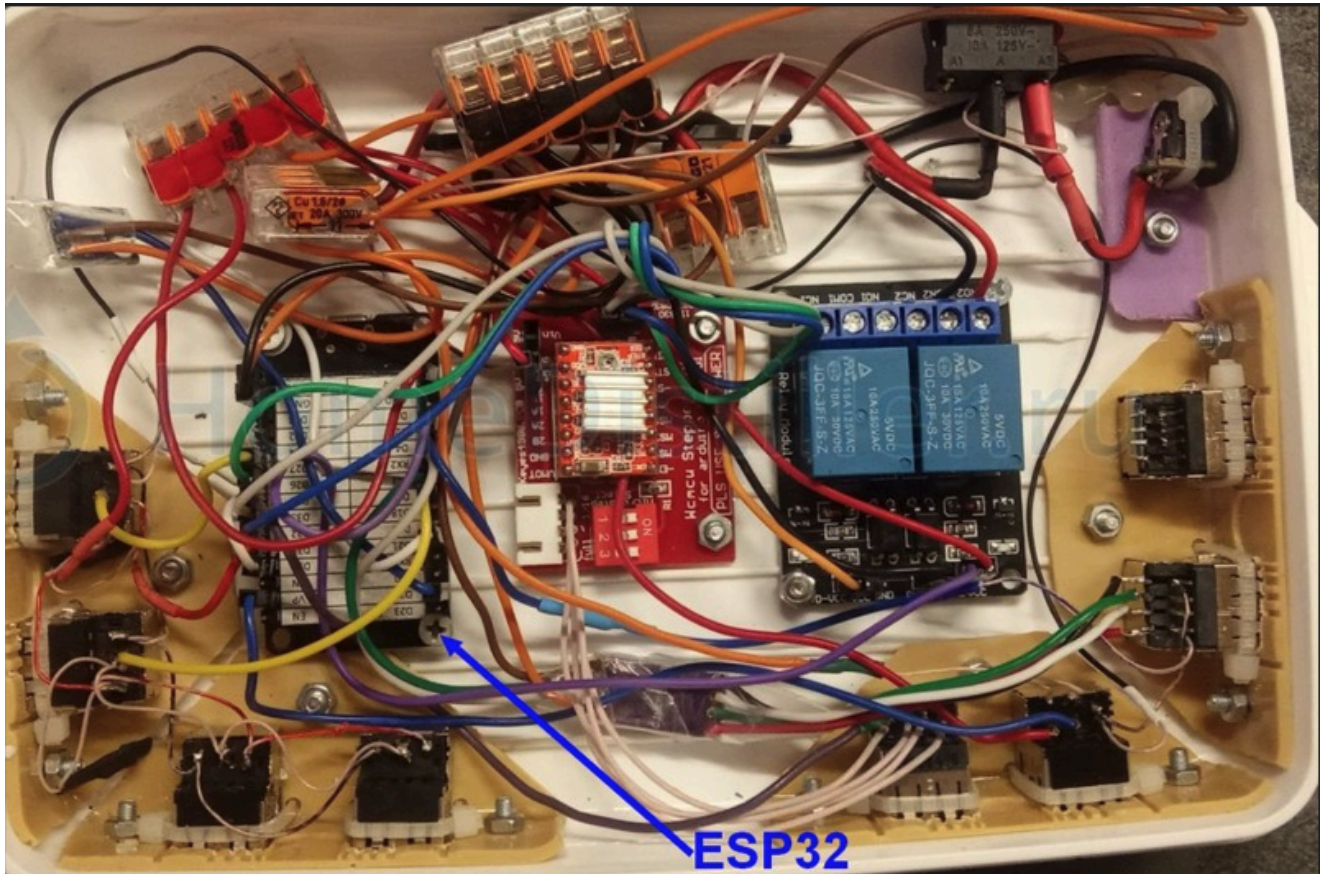
更多照片[见论坛帖子](#)。

不用 电路板组装

最后一种方案——完全不用印制电路板，点对点飞线连接。

把标有引脚名称的标签用打印机打印出来，贴在 ESP32 的背面，再把 ESP32 翻过来用铜柱固定在外壳上（便于散热）。导线用类似这样的接插件接到 ESP32 的引脚上：Dupont.

实现示例：



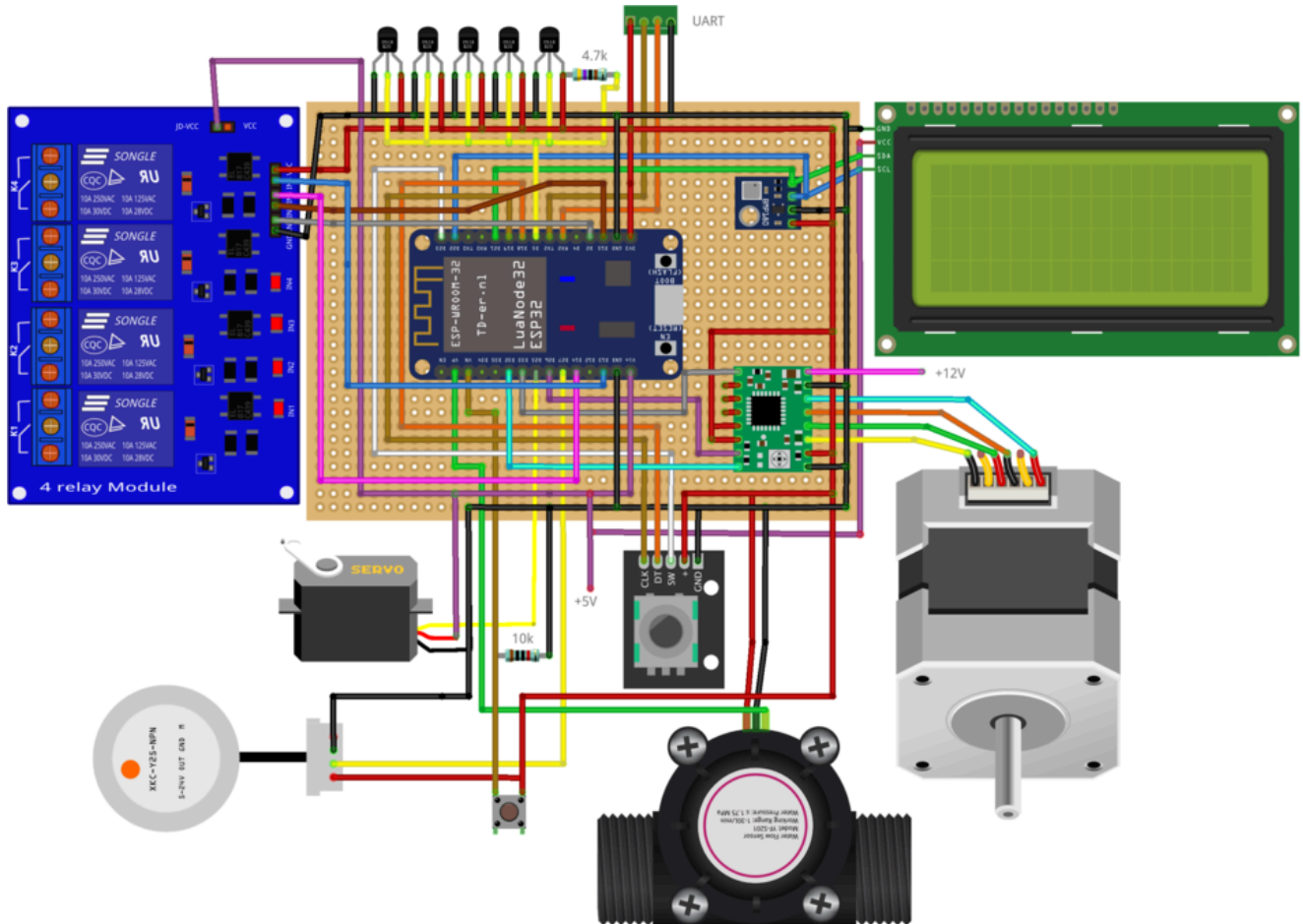
[在论坛上查看](#)

建议：

- 有印制电路板会让组装和之后的调试容易得多。但最重要的是，它能让您避开 ESP32 与外设之间因可能的接线错误和接触不良而引起的大部分问题。
- 如果想开始组装，但没有人分给您专用印制电路板，自己又不想下单生产——使用扩展板是最合适的方案。

最小配置

下面的最小配置用于监测，不对功率部分进行自动控制。



fritzing

更高分辨率的电路图在发布压缩包的 Fritzing 电路图文件夹中。

这里假定在印制电路板上安装 ESP32 时，已一并装好了[大气压力传感器](#)和[蜂鸣器](#)。

。接下来连接[温度传感器](#)。由于电脑通常在一处，塔在另一处，ESP32 可以用任意手机充电器供电。此时 Samovar 是您的观察员。它在显示器上显示消息，并发送到您的智能手机：升温结束、请开水、稳定结束、温度超限、暂停等等。您还能看到所有温度。蒸汽穿透时会弹出消息；有图表，可以分析塔的工作情况，等等。

但稳定和安全要靠您自己。它目前还不能控制任何东西，只负责通知。

仅用温度传感器和充电器供电来使用 Samovar，更像是一次测试。它可以让您确认 ESP32 和 WiFi 网络工作稳定。

继续组装时，需要把 Samovar 主板装进外壳，并使用更正规的电源。如果打算使用冷却水泵，需要一台 12 伏 80-100 瓦的电源。

也可以使用两台电源：40-50 瓦的给 Samovar 主板和步进电机供电，60-70 瓦的给冷却水泵供电。

下一步：

- 稳压型[调压器](#)可以让您几乎完全放心地把加热控制交给 Samovar：升温、稳定、保温，以及过程结束时关闭加热。
- [紧急断电](#)强烈建议在接好调压器后立即配置。
- [冷却设置](#)使出口处设定温度的保持完全自动化，开始时开水，运行结束时关水。

- [回流液位传感器](#)监测塔的液泛。与调压器配合，Samovar 会自动降低加热，防止液体从 TCA（通大气管）喷出。有了这个传感器，整个取酒过程中都可以让塔自动保持在预液泛状态。
- [取酒泵](#)
- [分馏分配](#)
- -

蜂鸣器

用于提示事件：信号为一串五声短促的蜂鸣。在网页界面中开启或关闭：“设置”（«Настройки»）→“主要”（«Основные»）选项卡，复选框“**使用蜂鸣器进行通知**”（«Использовать зуммер для нотификации»）（取消勾选时蜂鸣器完全停用）。同一选项卡中还有复选框“**切换程序时启用蜂鸣器**”（«Включать зуммер при смене программы»）。

它直接接到 ESP32 的引脚上（印制电路板上的 BUZ1 接口）：ESP32 DevKit 上为 D12，LILYGO T-Relay 4 上为 GPIO0，ESP32-S3 上为 GPIO39。



有源蜂鸣器 DC 3V（红色标签）；5V（蓝色标签）；12V（黑色标签）

请使用有源蜂鸣器，无源蜂鸣器无法工作。安装时注意极性。外壳上应有标记，或者长脚为正极。

如果手头有一个蜂鸣器，想确认它是不是有源的：

把电源调到 3 伏，接上负极，用正极碰几下另一个触点。如果没有声音，把电压升到 5 伏。

如果听到音调、蜂鸣声，就是有源的；如果正极接触时只听到咔哒声，就是无源的。12 伏的同理。

推荐使用 3 伏蜂鸣器，不过 5 伏的也有人用过，音量相当不错。

蜂鸣器没有音量调节。可以尝试在电源线上串一个电阻，或者干脆用胶带把发声孔稍微遮住一点。

大气压力传感器

Samovar 7.00: 传感器类型在配置器的“传感器” («Датчики») 选项卡上选择；只有非标准构建才需要手动编辑 `Samovar_ini.h`。

注意。

没有这个传感器 *Samovar* 也能工作，但最好还是装上，何况这个传感器非常便宜。

本项目可以使用 BMP180、BMP280、BME280 和 BME680 传感器。其中一些传感器不仅能测量大气压力和温度，还能测量湿度，甚至空气质量。

Samovar 只使用大气压力数据，所以用其中哪一种传感器都没有区别。

传感器与显示屏一起接在 I2C 总线上：ESP32 DevKit 为 SDA D21 和 SCL D22，LILYGO T-Relay 4 为 GPIO33/GPIO32，ESP32-S3 为 GPIO8/GPIO9。读数换算为 mm Hg，用于修正沸点温度。

印刷电路板上预留了 6 针传感器的焊盘，其中只连接了 4 个引脚。

因此，如果要购买传感器而不是用手头已有的，最好选简单可靠的 BMP180。关于这款传感器的更多信息见[这篇评测](#)。

该传感器无需额外设置。

警告

由于传感器通过 I2C 总线工作，某些 **BMP280** 传感器的地址可能不同。这种情况下，请在 `Samovar_ini.h` 文件中取消注释 `#define USE_BMP280_ALT` 这一行也就是应当改成这样：

...

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

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

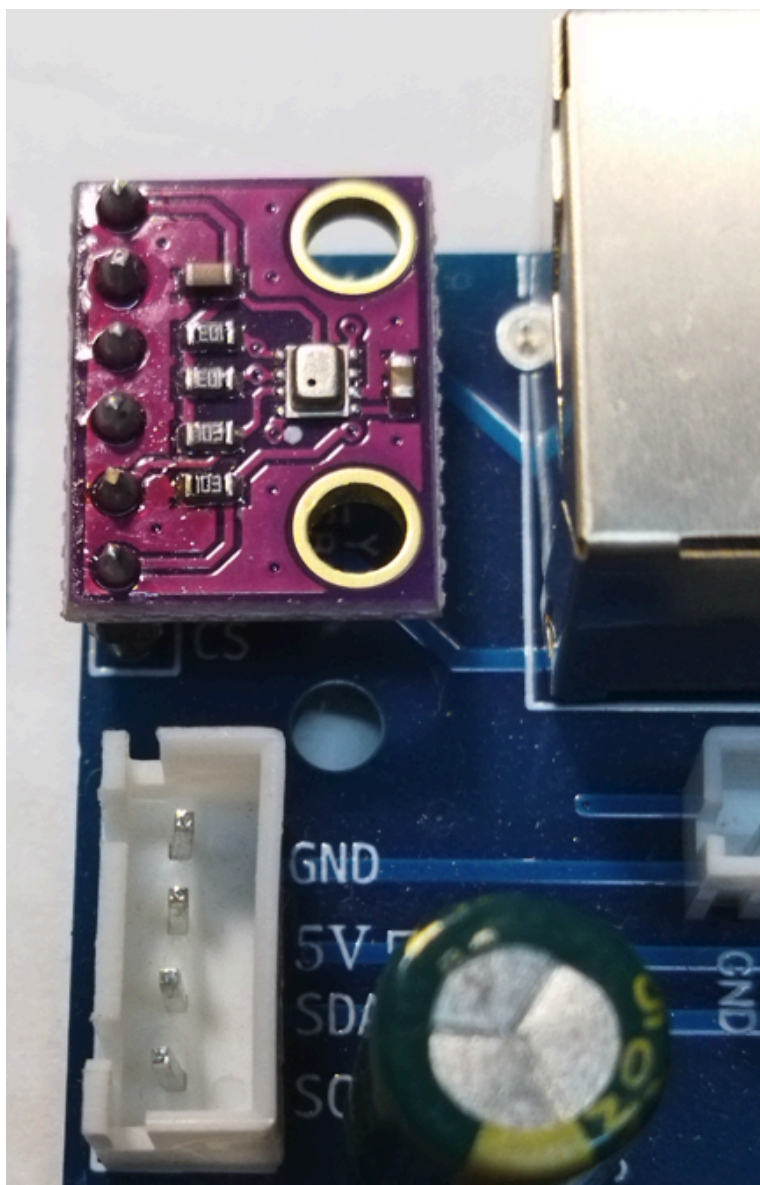
Samovar 7.00 的标准配置启用了 BME680。在配置器中，传感器在“传感器”

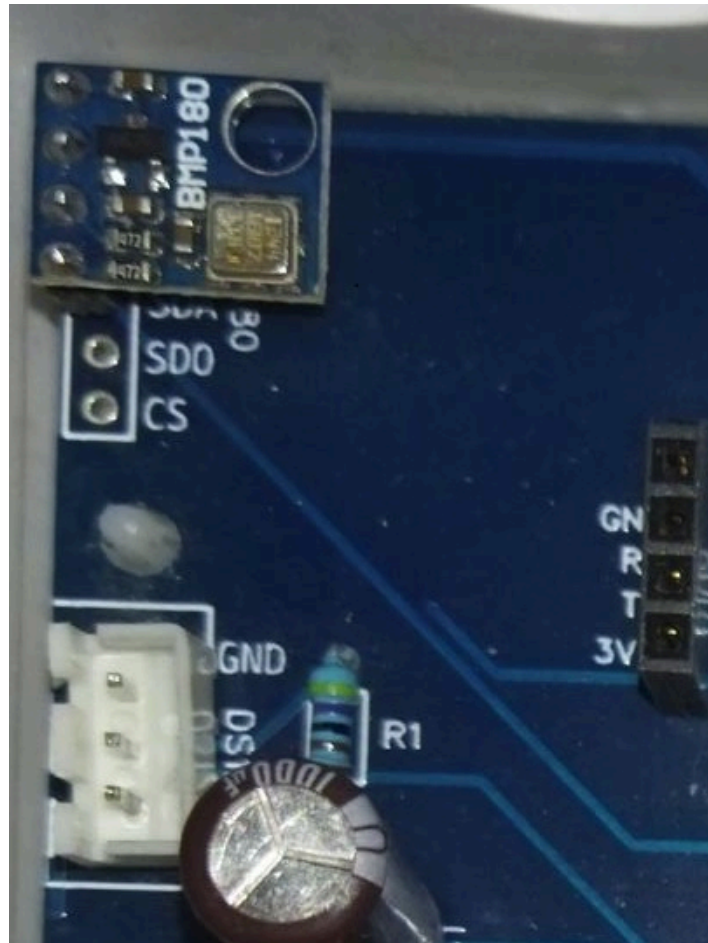
(«Датчики») 选项卡的“大气压力” («Атмосферное давление») 字段中选择：

“BMP180/BMP085”、“BMP280”、“BMP280，备用地址” («BMP280, альтернативный адрес»)、“BME280”、“BME680”或“不使用” («Не использовать»)。

在 Arduino IDE 中构建时，在 **Samovar_ini.h** 中只保留一条对应的指令处于启用状态；对于使用备用地址的 BMP280，请用 `USE_BMP280_ALT`。

最好把传感器模块直接焊在印刷电路板上。





温度传感器

Samovar 使用 **DS18B20** 数字传感器。用三根线连接：3.3 V 电源、GND 和一条公共数据线，数据线通过 4.7 kΩ 上拉电阻接到 3.3 V。

数据线：ESP32 DevKit 为 GPIO5，ESP32-S3 为 GPIO12，LILYGO T-Relay 4 为 GPIO27。所有传感器并联在同一条线上，固件通过各自唯一的地址区分它们。

支持五种用途：蒸汽温度、塔节、冷却水或回流液、蒸馏釜和 TCA（大气排气管）。安装后，在 Samovar 设置中把检测到的地址分配给相应的传感器（“设置”（«Настройки»）→ “温度”（«Температура»）选项卡，字段“分凝器前蒸汽传感器地址”（«Адрес датчика пара перед дефлегматором»）、“塔节传感器地址”（«Адрес датчика в царге»）、“水温传感器地址”（«Адрес датчика воды»）、“蒸馏釜传感器地址”（«Адрес датчика в кубе»）、“TCA 传感器地址”（«Адрес датчика в ТСА»）），并在开始加热前检查读数。传感器以 12 位分辨率轮询。如果某个传感器在过程开始时已连接、之后却消失，Samovar 会定期发出提示。

线路较长时，请使用质量好的电缆、可靠的连接，并为传感器单独供电。不建议使用两线寄生供电。

使用的调压器

Samovar 7.00: 调压器类型在配置器的“调压器” («Регулятор») 选项卡中选择：KVIC、RMVK、SEM_AVR 或“不使用” («Не использовать»)。

警告！

所有连接到 Samovar 的调压器都必须在电压上实现物理隔离。隔离器件可以是继电器、光耦、ADUM 等数据总线隔离器等。

绝对不要直接连接调压器，否则市电电压会传到您和 Samovar 身上。这种情况下最好的结果是 Samovar 烧毁。

Samovar 支持多种选项。在 Arduino IDE 中通过 **Samovar_ini.h** 中的指令选择。在 PlatformIO 中可以选择现成的环境：**Samovar** — KVIC，**Samovar_rmvk** — RMVK，**Samovar_sem** — SEM_AVR，**Samovar_no_power** — 不带 UART 调压器的版本。同一时间只能选择一个选项。

通过 UART 协议控制的调压器

使用 UART 的调压器的工作流程。

接通：Samovar 接通 1 号继电器，等待 **SAMOVAR_USE_POWER_START_TIME** 指定的时间（默认 2000 ms；在配置器中为“调压器”选项卡上的“调压器启动延时，ms” («Задержка запуска регулятора, мс»)；对于 SEM_AVR，在此之前还会再给调压器 3 s 的启动时间），然后向调压器发送工作命令。如果调压器没有响应，加热会关闭，并显示消息“调压器命令超时” («Таймаут команды регулятора»)。4 号继电器由所选模式的算法单独控制，不一定同时接通。

断开：1 号和 4 号继电器立即断开。700 ms 后向调压器发送 SLEEP 命令，再过 200 ms 停止流程结束。

1. 调压器 **KVIC**

与 Samovar 完全兼容。编码器调节步长 0.5 伏。稳压精度 0.1 伏。紧急情况下可由独立传感器关断。

2. 调压器 **RMVK** (只有 5 号和 6 号版本能配合 Samovar 工作)

所有 RMVK 型号的控制触点默认都有电源隔离，配合 Samovar 使用无需改装。

调压器上和从 Samovar 调节的步长均为 1 伏。稳压精度 1 伏。6 号版本紧急情况下可由独立传感器关断。通过 RMVK，Samovar 可以完全控制加热的升温、稳定和紧急关断。

警告！RMVK 只能处理整数值，即调节步长为 1 伏，这在 Samovar 以**预液泛**模式工作时有其特殊之处。

3. 调压器 **SEM_AVR**

与 Samovar 兼容。它有一个特点：在 Samovar 界面中，输出到负载的功率不是以伏特（像前两种调压器那样），而是以瓦特为单位设置和显示。

要与 Samovar 正确配合，需要使用 Samovar 自带的草图 (sketch) **Stab-avr.ino**，它位于目录 ...Samovar\Stab-avr 中

不通过协议进行外部控制的调压器。

例如：

[RM2、RM2M、RM2-Pro](#) (以下简称 RM)。

所有 RM 型号的控制触点默认都有电源隔离，配合 Samovar 使用无需改装。

对这类调压器的控制归结为两个参数：

- 接通/断开调压器本身。

- 启动/停止“升温” (Boost) 模式

该功能用两个继电器实现，由它们的触点控制调压器。收到 Samovar 的“开启加热”命令时，

1 号继电器的触点接通 RM。只有在所选模式的算法需要时，4 号继电器才接通“升温”模式。

4 号继电器断开触点，RM 关闭升温模式。运行结束时，Samovar 用 1 号继电器的触点关闭 RM。

注：

- 上面列出的所有调压器都可以独立工作，也可以不通过 UART 连接、按照 RM 调压器的原理和算法配合 Samovar 使用。

Samovar 和调压器显示的功率不一致，怎么办？

总的来说，功率的绝对值并没有那么重要，但让调压器和 Samovar 显示相同的功率是可以做到的，而且本质上只有一种方法。

所有调压器，包括 Samovar，都不测量功率。它们像计算器一样，根据两个值算出功率：当前加在加热管上的电压，以及输入的该加热管的电阻值。

输入的电阻是常数，电压是变化的。整个调整就归结为选取加热管的电阻值。

可以采用计算出的电阻：电压的平方除以功率。电压取加热管铭牌上的值。以 3 kW、220 伏的加热管为例： $220 \times 220 / 3000 = 16.133333333333 \text{ Ohm}$ 。

也可以用万用表测量。

关于 **RMVK** 调压器以及基于中国模块的简易调压器 (**Stab-avr**)。

警告！如果并不打算真正加热蒸馏釜，请断开加热管。

- 在 **RMVK** 中可以输入功率和电压值，电阻会自动算出；或者输入电阻，功率会自动算出。[设置方法视频](#)。看完照做，别忘了把调压器切换到以瓦特显示。

- 在 **Stab-avr** 中，用电位器设定当前实际的市电电压，并输入您的加热管功率值。

警告！Stab-avr 的加热管功率必须按 230 伏市电电压输入。如果加热器厂家在技术资料中给出的是 220 伏下的功率，需要换算成 230 伏下的功率。

启动 Samovar——加热。在升温模式下调整没有意义——电压一直在变，功率也随之变化。从 Samovar 发出命令，让调压器设定某个具体电压（比如 RMVK 设 170 伏，Stab-avr 设 2000 瓦）。调压器会转入稳压模式。

比较 Samovar 和调压器上显示的功率值。

现在有两种做法——要么修改 Samovar 中的电阻值（网页界面：“主页 - 设置”

(«ГЛАВНАЯ - Настройки») - “主要” («Основные») - “其他” («Прочие») 选项

卡)，这要方便得多；要么修改调压器中输入的功率值。目标是让调压器和 Samovar 上的数值尽可能接近。

Kvic 调压器。

它没有功率显示，所以只能调整 Samovar，调到您想看到的功率.....

这是第一种做法

第二种是使用这样的链路：独立的[电压电流表](#)— 调压器 — 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

如果不时更换不同的加热管，和/或切换加热管组件内各加热管的接法，这种方式会很方便。

电压电流表会真实测量所消耗的电流，并在屏幕上显示实际功率。所以如果希望 Samovar 显示这个功率，只需随时修改 Samovar 中的加热器电阻值即可。

另见：

- 选型帮助：[调压器概览](#)
- 除了控制调压器，Samovar 还可以[控制额外的辅助加热管](#)
- 如何不用导线、[通过无线信道将调压器连接到 Samovar](#)

调压器概览

本概览帮助您选择配合 Samovar 使用的调压器。

下面介绍的所有调压器都是输出电压稳压器。无论市电如何波动，它们都输出稳定的电压。它们不是升压设备，只有在输出电压低于输入电压时才能稳压。它们都可以作为独立设备工作。

注意。

本概览中调压器的优缺点仅从配合 Samovar 工作的角度来考虑。

KVIC 调压器

基于 Arduino 单片机的稳压器。如果有元件，并且具备组装和/或维修电子设备的技能，可以自己制作。手动控制 - 编码器。显示屏：0.96 英寸 LCD 也可以购买成品。外形、配置、保修和其他条件与制造者单独商定。源代码用汇编语言编写，开源，并且[公开提供](#)，您可以自行修改。

优点：

- 启动快，输出电压稳定，精度可达 0.1 伏
- 用带按钮的编码器手动控制，调节步长 0.5 伏
- 显示屏易于读数。
- 有稳压指示。
- 可以手动预设稳压电压，执行“停止升温”（«Стоп разгон»）命令后调压器会切换到该电压。
- 通过 UART 或 I2C 控制
- 可维修，新版本发布后可以更新程序。可以安装串口速率为 9600 波特的替代固件（默认 38400 波特）
- 通过单独的功率继电器控制辅助加热管。
- 有外部“停止升温”（«Стоп разгон»）命令接口
- 有外部“报警”（«Авария»）命令接口
- 启动时平滑地向负载施加电压，关断时平滑地撤除电压。

缺点：

- **调压器的所有元件**都处于高电压下。
- 更换程序时可能需要把单片机从调压器上拆下来。
- 预设的稳压电压在断电或重启时不会保存在设备存储器中。
- 维修和烧录需要专门技能。

“定制”购买或自己制作：[论坛支持帖](#)。

RMV-k 调压器



采用 2DIN 外壳、DIN 导轨安装的稳压器。手动控制和设置——用按键。信息显示：三位数码管。无法自己制作。

该稳压器在售有 6 个版本、5 种配置。（撰写本概览时）。
注意！只有 5 号和 6 号版本适合通过 UART 配合 Samovar 工作。

优点：

- 产品使用寿命 — 10 年。
- 6 个月保修
- 电压保持精度 1 伏。调节步长 1 伏
- 可存储 10 个预设值
- 加热管电阻保存在存储器中，用于计算功率
- 电压表校准
- 可以切换为以瓦特显示读数。
- 由 Samovar 通过 UART 控制（仅 5 号和 6 号版本）。
- 有外部“报警”（«Авария»）命令接口（3 号及以上版本有）。
- 有外部“停止升温”（«Стоп разгон»）命令接口（4 号及以上版本有）。

缺点：

- 维修时需要专门技能和设备。
- **调压器的所有元件**都处于高电压下。
- 无法更新固件
- 如果调压器作为独立设备使用，要使用“停止升温”（«Стоп разгон»）和“报警”（«Авария»）功能，需要额外的电源（一节 9 V “Krona”电池即可）
- 手动编程和向存储器输入数值比较复杂。

- 手动控制有些怪癖（例如，要降低电压，必须先把电压升高 1 伏，然后才能降低）。

可以通过[论坛支持帖](#)向开发者购买调压器（仅调压器，不含可控硅等）
在[网店](#)可以买到装在盒子里的完整成品，接上加热管即可工作（那里也可以买到自行组装用的元件套件）。

Stab-avr 调压器

用中国模块制作的简易加热管稳压功率调节器

基于 Arduino 单片机的稳压器。用按键控制。显示屏：0.96 英寸 LCD。只要有最基本的焊接技能就能自己制作。不出售。

源代码用 C++ 编写，开源，可以用 Arduino IDE 自行修改。

优点：

- 用现成模块组装，简单。
- 完全可维修。
- 由 Samovar 通过 UART 控制
- 可以更新程序，源代码开放。
- 除功率器件外，调压器的所有元件都处于安全电压下。
- 可以不经电压隔离直接接到 Samovar（Arduino NANO 和 ESP32 的电压需要匹配）。
- 外部“停止升温”（«Стоп разгон»）接口
- 外部“报警”（«Авария»）接口
- 电压表校准
- 可以预设并在存储器中保存输出功率值，执行“停止升温”（«Стоп разгон»）命令后调压器会切换到该功率。
- 加热管额定功率保存在存储器中。

缺点：

- 原版固件配合 Samovar 工作时启动时间较长。建议使用 Samovar 草图（sketch）附带的修改版固件。

自己制作的方法：[论坛支持帖](#)。

RM 调压器（RM2M、RM2M-Pro）



采用 3DIN 外壳、DIN 导轨安装的稳压器。手动控制和设置——用按键。信息显示：三位数码管。无法自己制作。

该稳压器由多家厂商销售，因此各家的配置和保修各不相同。可以只买稳压器（不含可控硅 等），也可以买可直接工作的整套设备。

优点：

- 设定电压保持精度：正负 1 伏。
- 调节步长 1 伏
- 显示负载消耗的功率：0 至 9.99 kW。
- 可存储 10 个电压预设值。
- 内置电压-时间定时器：0 至 999 分钟。
- 使用“停止升温”（«Стоп разгон»）和“报警”（«Авария»）功能时不需要额外电源

缺点：

- 没有控制通道。
- 编程和向存储器输入数值比较复杂。

结论。

配合 Samovar 使用时，推荐前三种调压器。只要采用 UART 控制，Samovar 就能完全控制调压器：从启动升温，到随时改变电压以及关闭加热。

可以选用由第三方设备紧急关断的功能。例如，Samovar 配合 PVK（蒸汽夹套锅）使用时，在蒸汽-水夹套中压力超限时关闭加热。可以用电接点压力表作为

传感器。

KVIC调压器。手动控制方便，0.5 伏的调节步长可以相当精确地把塔调到预液泛状态，并长时间保持。但只有经验丰富的电子工程师才能制作它。

RMV-k调压器。

工作稳定的设备，配置选择相当多。可靠，据了解几乎没有什么抱怨。通过 UART 连接 Samovar 时控制方便简单。可以买到连同“停止升温”（«Стоп Разгон»）和“报警”（«Авария»）传感器在内、可直接工作的成品盒。这些传感器可在调压器独立工作时使用，它们是 KSD9700 双金属温控开关。连接 Samovar 时不使用“停止升温”传感器，因为该接口被用作数据交换线路——相应地，Samovar 自己控制“停止升温”模式。“报警”传感器可以作为独立选项使用。这些传感器不适合在 Samovar 中使用，但在制作附加的[保护线路时可以用作元件](#)。

Stab-avr 调压器也毫不逊色。可以自己制作和维修，这一点非常有吸引力。这种调压器的特点是功率设定以瓦特显示（而不是像上面介绍的调压器那样以伏特显示）。

调节步长为加热管额定功率的 1/200。由于功率设定方式的原因，实际值可能与 Samovar 设定的值略有不同（例如 1803 W 而不是 1800）。这完全没有影响。

RM (RM2M、RM2M-Pro) 调压器的所有优点都被一个大缺点抵消了，那就是没有 UART 通信，因此无法“从外部”设置调压器的电压。稳压电压只能手动设置。

使用 RM 时，Samovar 只能接通/断开调压器以及开启/关闭升温。因为这种调压器不提供任何其他由第三方设备控制的方式。

建议。

只要有一点使用电烙铁的经验，就值得自己制作调压器 **Stab-avr**。

如果对自己的能力很没把握或者没有时间，就值得购买完整组装好的调压器 **RMV-k**。

如何将调压器连接到 Samovar

有线连接。

Kvic 和 RMVK 调压器

KVIC 调压器用 ADUM1201 实现电压隔离。Samovar 与它通信的速率为 38400 波特；启用

KVIC_USE_9600 时为 9600 波特。

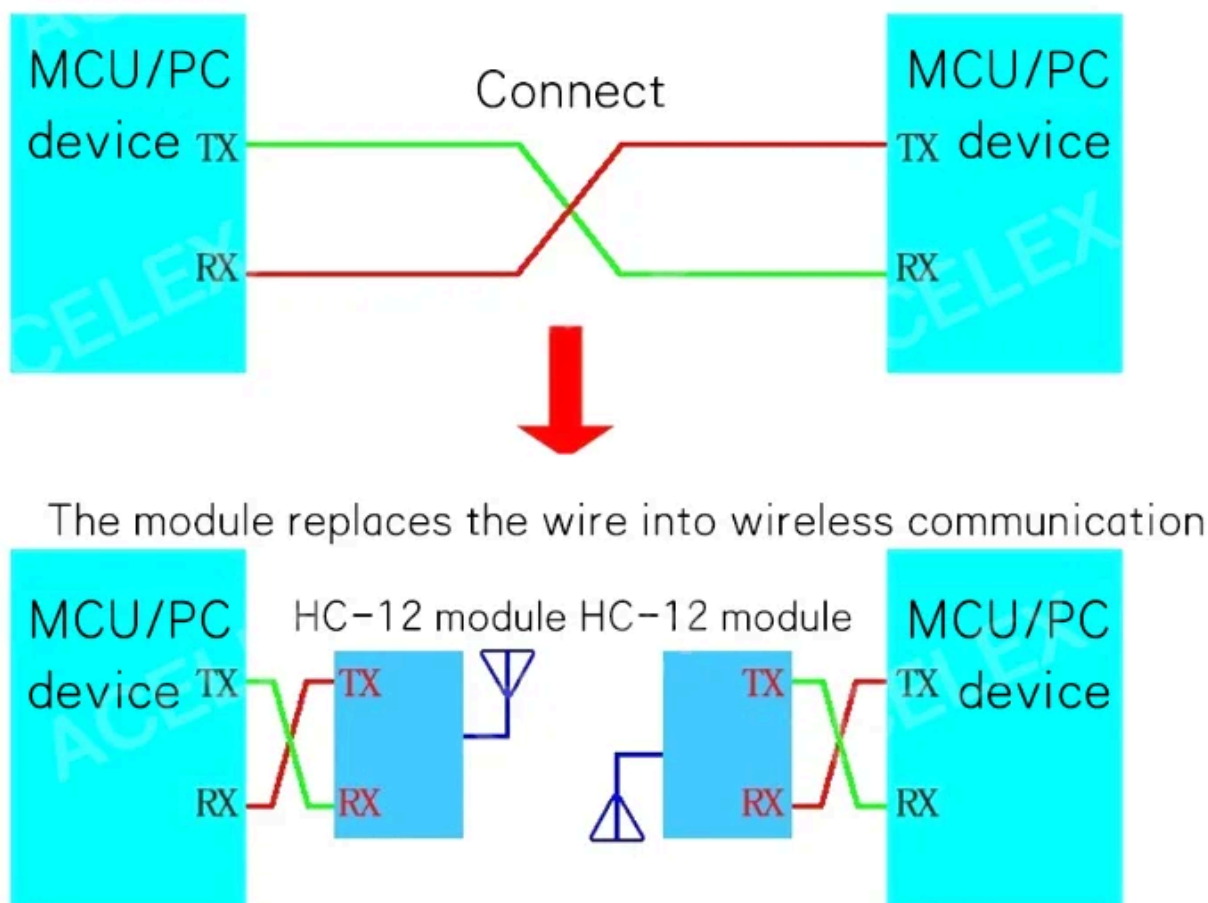
RMVK 调压器用 PC817 光耦实现隔离。串口速率为 9600 波特。

这些调压器与 Samovar 的连接方式相同。一台设备的 Rx 必须接到另一台设备的 Tx。

Samovar 一侧使用 UART2（印制电路板上的 UART 接口）：ESP32 DevKit 上为 RX D16、TX D17，LILYGO T-Relay 4 上为 RX GPIO25、TX GPIO4，ESP32-S3 上为 RX GPIO18、TX GPIO17。SEM_AVR 调压器（Stab-avr）也接在这个串口上，速率为 9600 波特。

如何无线连接调压器与 Samovar

使用通过 UART（COM 口）控制的调压器时，Samovar 可以不用电缆，而是借助两个 RF 模块（射频模块）HC-12来连接。方法非常简单，只需按图连接：



这里不需要 ADUM，因为这些模块设计工作电压为 3.1 V 至 5 V，所以既可以接到 ESP32（Samovar），也可以接到 Arduino Nano。

厂家标称该模块的传输距离可达 1 km（开阔地带并使用外置天线），对我们的用途来说绰绰有余。模块出厂默认速率为 9600 bps。如果使用通过 UART 控制的 STAB-AVR 或 RMVK 调压器，只需按图把一个模块

接到 Samovar，另一个接到调压器，就应该可以工作了。如果使用 Kvic 调压器，则必须对两个模块进行设置。这并不难，但需要把它们接到电脑上，修改模块之间交换数据的速率。

两个 HC-12 模块的速率必须与 Samovar 和调压器一致。RMVK 和 SEM_AVR 为 9600 波特。KVIC 使用 38400 波特；如果编译时启用了 **KVIC_USE_9600**，则为 9600 波特。

该模块有很多选项，但我们只关心一个：修改模块的串口速率。

要让模块进入设置模式，把 SET 引脚接到 GND。然后给模块上电，并通过 UART-USB 转换器（可以用 Arduino Nano）接到电脑。如果选择 38400 波特，向识别出的 COM 口发送命令 **AT+b38400**，等待回复 **OK+b38400**。使用 9600 波特时，不需要修改 HC-12 的出厂设置。

断开 SET 与 GND 的连接，断电，把模块接到 Samovar 和调压器上，就应该可以工作了。

警告！

Kvic 调压器与市电电压之间没有电气隔离，连 HC-12 的天线都会带市电电位！因此模块必须放在调压器外壳内部，或以其他方式可靠绝缘。

加热管

可以使用您的调压器能够控制的任何加热管。这里介绍最常见的一种加热管。

在俄语中，缩写 ТЭН（加热管）的意思是“管状电加热器”。因此，组件中有几根管就有几个加热器。用加热管的总功率除以管的数量，就得到一根管的功率。所有管的功率相同。



通常选择 3 kW 的型号，每根管功率为 1 kW。对于 1.5" 的塔，这是个不错的选择，因为 1.5 米高的塔在预液泛模式下运行需要 1.2-1.4 kW 的功率，这样还留有功率余量。

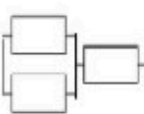
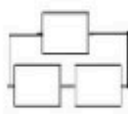
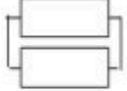
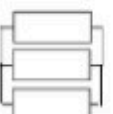
因此可以采用组合接法：

- 两根管经调压器连接，得到 2 kW。精馏时调压器把输出电压保持在 170-180 伏，即使市电电压“下沉”也完全够用。
- 剩下的一根管通过固态继电器连接，该固态继电器由 4 号继电器的触点控制。这根就是辅助加热管。

对于 2" 的塔，按这种接法 3 kW 的加热管不够用。4.5 kW 的正合适。

通过改变各管的接法，可以使用功率大得多的加热管，而且它们会工作在更温和的状态下。当然，要遵循够用即可的原则。

下表列出了组件中加热器的各种可能接法。

	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

如何看表（以 4.5 kW 加热管为例）

在左列选择功率 - 4.5 kW。顶行的编号表示加热器的接法。

例如，1 号是三个加热器全部串联。行与列交叉处是组件接 220 伏时输出的功率 = 0.5 kW，也就是 500 瓦。

6 号接法用 2 个加热器得到 3 kW，剩下一个加热器不用——它可以用作辅助加热管。

使用辅助加热管

Samovar 可以控制一根附加的辅助加热管，适用于“[精馏](#)”（«Ректификация»）、“[蒸馏](#)”（«Дистилляция»）和“[啤酒](#)”（«Пиво»）模式

要使用此功能，请在 4 号继电器的常开触点上接一个接触器或固态继电器，其负载电流须足以驱动第二个加热器。

“精馏”模式下辅助加热管的工作

使用附加辅助加热管无需修改草图中的任何内容。该功能默认启用。

下文提到的阈值温度在配置器的“温度”（«Температуры»）选项卡中设置（“退出升温温度，°C”（«Температура перехода из разгона, °C»）、“开启冷却温度，°C”（«Температура открытия охлаждения, °C»）、“辅助加热管阈值，°C”（«Порог разгонного ТЭНа, °C»）；在 Arduino IDE 中构建时，由 [Samovar_ini.h](#) 中相同的指令设置。

其逻辑与[PM](#)在“升温”（«Разгон»）模式下的逻辑类似：

收到“开启加热”（«Включить нагрев»）命令时，Samovar 除了开启主加热管外，还通过 4 号继电器接通接触器或固态继电器，由后者给辅助加热管供电。当蒸汽温度达到设置中的值（默认 39°）时：

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

4 号继电器断开触点，关闭辅助加热器。

使用 UART 控制的调压器（KVIC、RMV-K、SEM_AVR）时，还有两个条件：只有在“升温”（«Разгон»）状态有效、且蒸馏釜温度不高于 **OPEN_VALVE_TANK_TEMP**（默认 77°，“开启冷却温度”）时，4 号继电器才接通。蒸馏釜一旦升温超过该值，即使蒸汽尚未达到 39°，辅助加热管也会关闭。如果没有连接蒸馏釜传感器，Samovar 不控制辅助加热管。没有 UART 调压器时，4 号继电器在加热开始时与 1 号继电器一起接通，在升温结束时断开。

如果没有发出“开启加热”命令，或者传感器温度高于草图中设定的值，升温不会开启。

“蒸馏”模式下的工作。加热开始时辅助加热输出接通。开始沸腾后，如果设置中（“设置”（«Настройки»）→“啤酒”（«Пиво»）选项卡）取消勾选了“**沸腾时使用辅助加热管**”（«Использовать разгонный тэн при кипении»）复选框，输出就断开；如果勾选了该复选框，输出保持接通，直到加热正常停止。

“啤酒”模式下辅助加热管的工作

加热管的接法与上文相同。

但算法不同

糖化期间的辅助加热管由“辅助加热管阈值，°C”参数（配置器的“温度”选项卡）控制；在 [Samovar_ini.h](#) 中对应这一行：

```
#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

默认启用， 4 为摄氏度数。仅适用于 UART 控制的调压器。没有这种调压器时，只要温度低于目标值，4 号继电器就与主加热一起接通，并与主加热一起断开。

开启后，Samovar 命令调压器向主加热管输出全功率。如果蒸馏釜温度比糖化程序行中设定的温度低 4°，Samovar 接通 4 号继电器，由它命令辅助加热管开启。如果温差小于 4°，辅助加热管不开启，只由主加热器加热。

当前一个温度停留结束，或操作者发出“下一个程序”（«Следующая программа»）命令时，Samovar 把蒸馏釜温度与下一程序行中设定的温度比较。如果温差大于 4°，辅助加热管投入工作。

如果没有发出命令“开启加热”，或者温差小于设定值，升温不会开启。

麦汁煮沸期间，如果勾选了“沸腾时使用辅助加热管”复选框，辅助加热管接通，否则断开。

Samovar 7.00: 在“奶酪制作”（«Сыроварение»）模式下，4 号继电器用于盐水排放阀。请在它上面接奶酪阀门或用于其他模式的辅助加热管，但不能同时接这两种设备。“BK”（«БК»）、“NBK”（«НБК»）和“低温慢煮”（«Су-вид»）模式没有单独的辅助加热管逻辑：使用 UART 调压器时，这些模式下 4 号继电器不会接通。

接触器。漏电保护器。紧急断电

应对异常情况的附加保护。

建议制作。

这里介绍的保护装置独立工作，在严重异常情况下切断设备电源。

主要危险是酒精蒸汽穿透。如果由于某种原因发生以下任何一种情况：

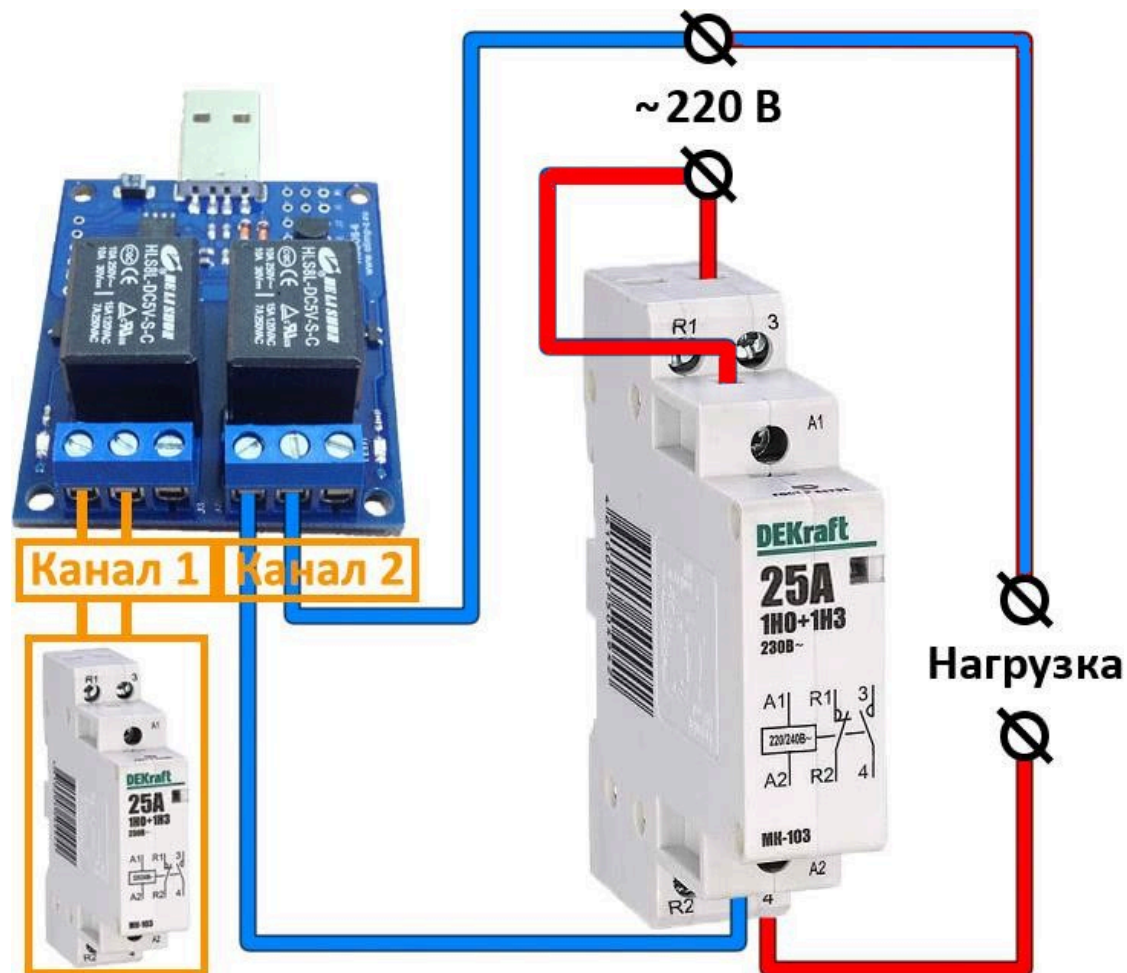
- 可控硅损坏（击穿短路）。
- 调压器死机或工作异常。
- 自动控制失灵。
- 12 V 电源损坏。
- 没有冷却。

这时必须关闭加热，阻止蒸汽外泄。监测点是 TCA（通大气管）。

方案 1

适用于 Samovar 配合接触器使用的情况。

将接触器连接到[继电器模块](#)



需要一个双金属温控开关 KSD9700 NC（常闭型），动作温度 55-70°C。最好选外壳绝缘的：陶瓷或塑料外壳。

制作方法：

12 V 电源的负极不直接接到 Samovar 主板，而是经过固定在 TCA 上的 KSD9700。最方便的做法是在 Samovar 外壳上装一个两针插头，再配一根所需长度的导线。

工作原理。

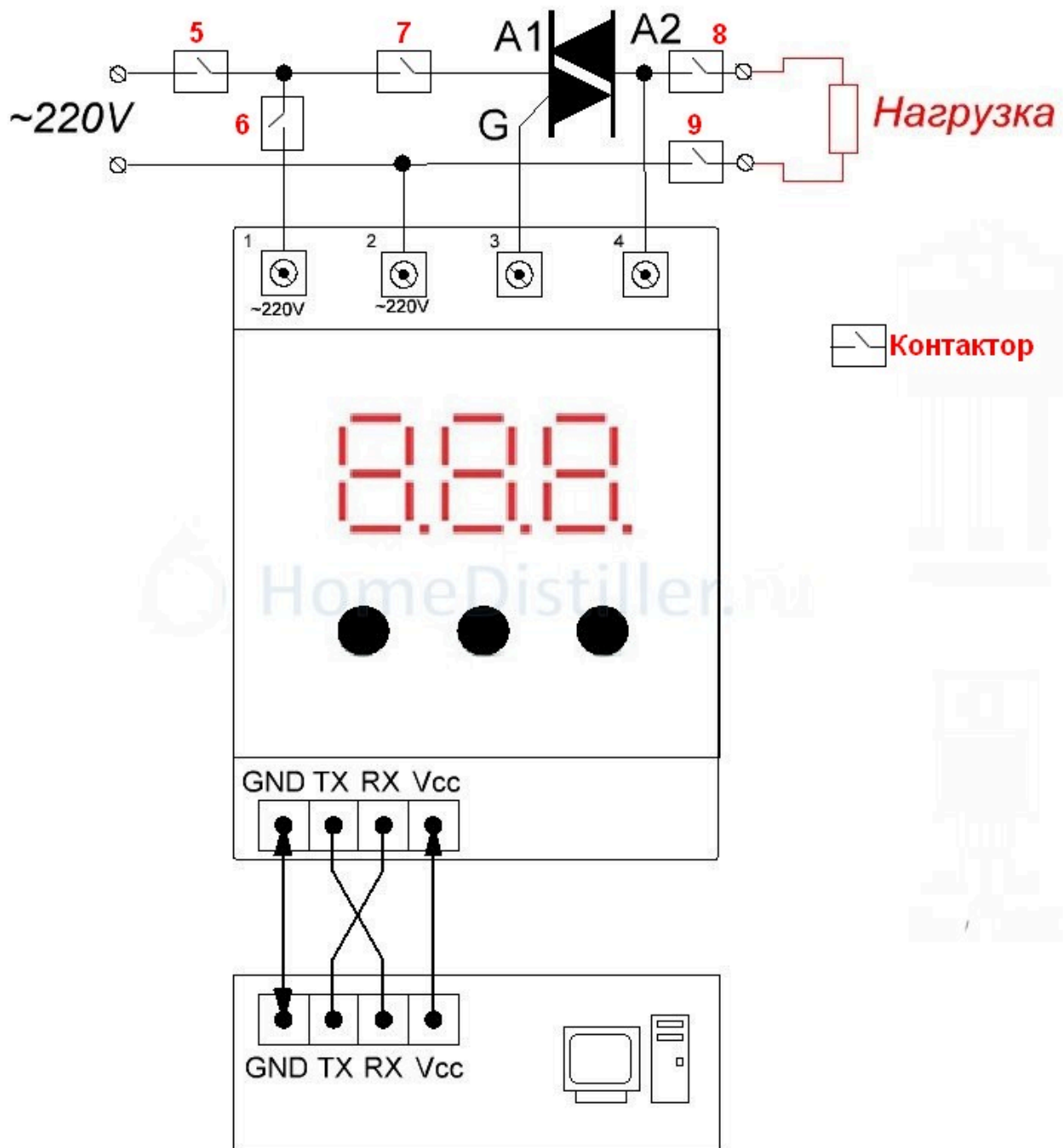
如果发生 Samovar 无法应对的紧急情况（与调压器失去通信、可控硅击穿等），蒸汽开始从 TCA 穿出。温控开关受热后断开电路。Samovar 断电，控制继电器释放接触器，接触器随即切断加热。蒸汽外泄停止。如果使用了水阀，水阀会关闭。

过一段时间 TCA 冷却，温控开关触点闭合，Samovar 重新上电。但其他所有设备没有您的命令不会接通。

当然，首先要查明断电的原因。

当然，把 Samovar 接到塔上时如果忘了插上插头，Samovar 也不会开机，因为没有电压，这相当于电源损坏。

接触器可能安装位置示意图



位置 5。接触器接通后给调压器供电，同时也给可控硅供电，可控硅按调压器的命令向负载输出电压。
接触器由 Samovar 的“开启加热” («Включить нагрев») 命令接通。因此，调压器需要一些时间启动，之后才会收到 Samovar 的“升温” («Разгон») 命令并向负载施加电压。这段延时由参数 **SAMOVAR_USE_POWER_START_TIME** 设置（默认 2000 ms；在配置器中为“调压器” («Регулятор») 选项卡上的“调压器启动延时，ms” («Задержка

запуска регулятора, мс»))，对于 SEM_AVR 还要再加 3 s。如果调压器在这段时间内没有响应，Samovar 会关闭加热，并显示消息“调压器命令超时”

(«Таймаут команды регулятора»)。

在此位置安装接触器之前，请权衡利弊。

位置 6. 强烈不建议。接触器装在这里实际上只是断开调压器，仅此而已。调压器本身几乎不耗电，如果只需要断开调压器，装一个小功率开关就够了。这样连接时，如果调压器死机、可控硅损坏等，加热管仍会继续加热。

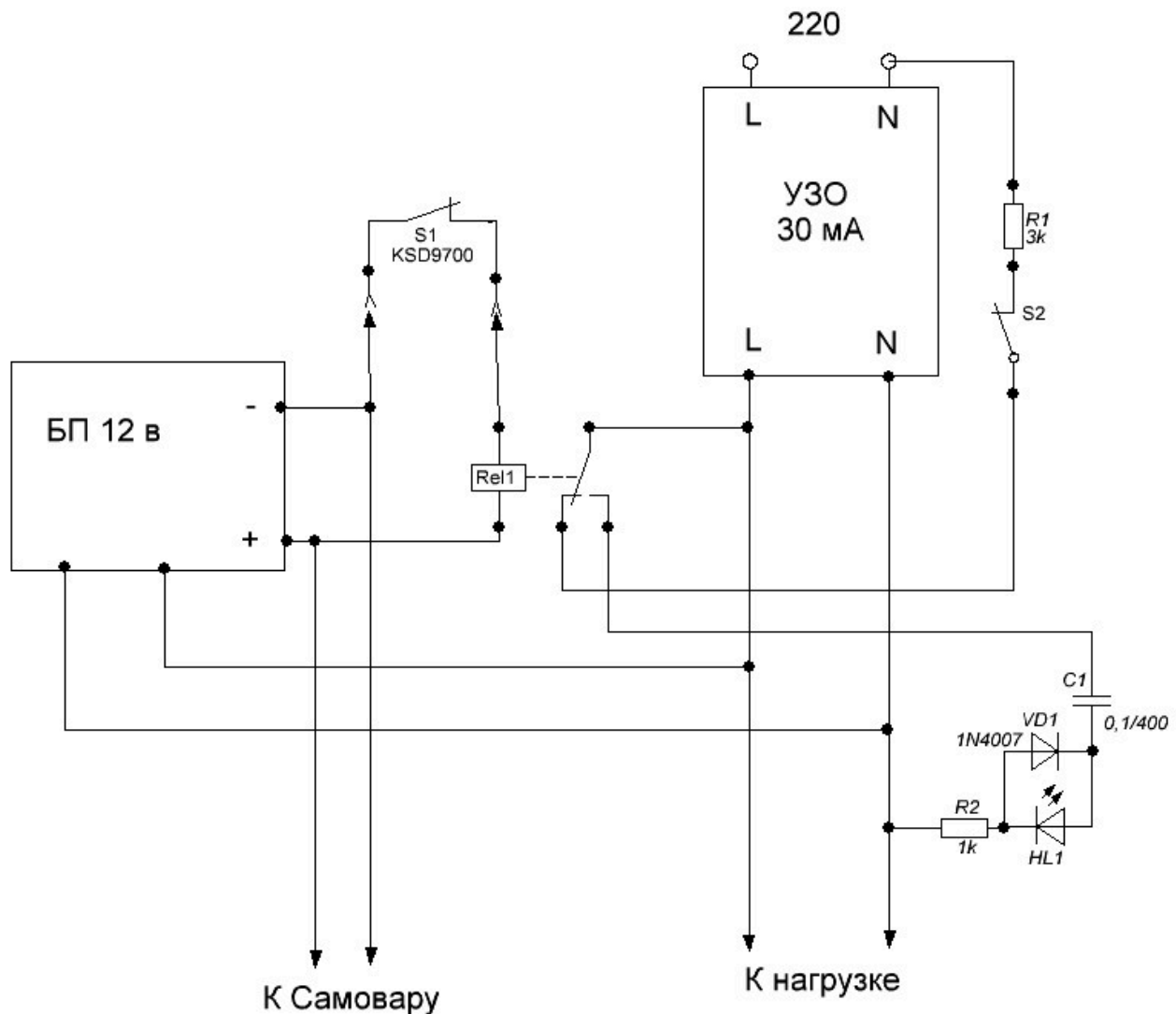
位置 7 和 8。接通主电源时，调压器就会开机，按自己所需的时间启动。可以对它进行调节、设置电压等。但负载上没有任何输出，因为接触器断开了通往可控硅的电源电路。只有当 Samovar 发出“开启加热” («Включить нагрев») 命令时，接触器才会闭合，电压经可控硅送到负载。

位置 9 - 推荐安装位置。接触器装在这里且可控硅关断时，加热管完全断电。

方案 2

可以用漏电保护器（剩余电流动作保护器）代替接触器，它还能增加防止设备外壳漏电的安全性。

传感器 S1（常闭型 KSD9700）安装在 TCA 上。



工作原理。

要接通包括 Samovar 在内的所有设备，请按往常闭按钮 S2。接通由漏电保护器本身完成。通电后 12 V 电源启动，给继电器 Rel1 的线圈供电。继电器触点 (NO) 闭合，LED HL1 点亮。此时可以松开按钮 S1。保护已就绪。如果 LED 没亮就松开按钮，漏电保护器会跳闸。

来看看可能出现的情况：

- 操作者忘了连接传感器 S1 – 继电器不吸合 – 漏电保护器跳闸
- 可控硅损坏 – 蒸汽从 TCA 穿出 – 传感器 S1 断开电路 – 继电器失电 – 漏电保护器跳闸
- 冷却不足 – 蒸汽从 TCA 穿出 – 传感器 S1 断开电路 – 继电器失电 – 漏电保护器跳闸
- 12 V 电源损坏 – 继电器失电 – 漏电保护器跳闸
- 自动控制和/或调压器死机 – 蒸汽从 TCA 穿出 – 传感器 S1 断开电路 – 继电器失电 – 漏电保护器跳闸。

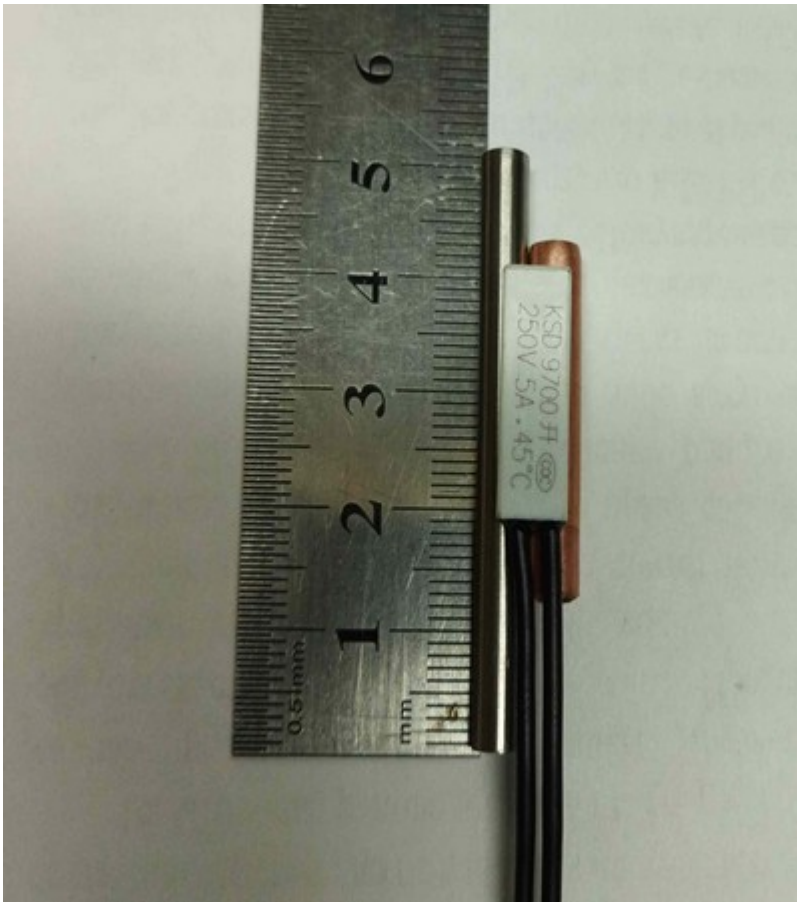
- 短时停电 – 来电时漏电保护器跳闸。

监测单元制作示例

TCA 管路监测单元。

材料：直径 6 mm 的不锈钢管，直径 6.1 mm 的铜管（假定 18B20 传感器不带套管；如果带套管，就不需要铜管），KSD9700 传感器，热缩管，硅胶管。

把不锈钢管和铜管（或温度传感器套管）并排放好。管子长度选得让不锈钢管长出几厘米。把 KSD9700 传感器放在上面，用热缩管把整个组件包起来。把在 Samovar 中指定为 TCA 传感器的 18B20 传感器插入铜管。以防万一，用任意密封胶封住铜管的一端，防止液体进入





精馏时，把不锈钢管串接在分凝器 TCA 到蒸汽冷凝液排放容器的管路中。经验表明，最好离 TCA 约 30 cm。

蒸馏发酵醪和运行醪塔时，把不锈钢管直接接到成品出口嘴上。也就是说，成品流经这根管子再流入接收容器。此外，您还能以 TCA 温度的形式看到出口处的成品温度。

由于可以实时调节冷凝器出水温度，就可以把水泵调到最佳出力。



也可以完全不用铜管，把不带套管的传感器直接压在不锈钢管上，但这样就得自己解决传感器的密封问题，而且 KSD9700 的导热接触面积也会更小。

为了正确工作，Samovar 根据 TCA 传感器关闭加热的温度（配置器“温度”（«Температуры»）选项卡上的“TCA 报警温度，°C”（«Аварийная температура TCA, °C»）；在 [Samovar ini.h](#) 中为这一行）

//关闭电源时的 TCA 最高温度

```
#define MAX_ACP_TEMP 75
```

必须设得比 KSD9700 的动作温度低 5-10°。默认值 75° 高于温控开关 55-70° 的阈值，因此必须调低。

这样蒸汽穿透时，Samovar 会先动作并发出关断命令；如果命令没有执行，接触器或漏电保护器就会动作。

冷却

Samovar 7.00: 冷却开启温度和水泵参数在配置器的“温度” («Температуры») 和“水泵” («Насосы») 选项卡中设置。

在精馏和蒸馏模式下，Samovar 保持设定的冷却水温度。

为组织冷却，Samovar 使用：

1. 3 号继电器输出：ESP32 DevKit 上为 GPIO14，ESP32-S3 上为 GPIO41，它控制从自来水管供水的阀门。也可以是任何其他需要在“请打开供水” («Откройте подачу воды») 命令下接通的设备。
2. 水泵控制输出：ESP32 DevKit 上为 GPIO4，ESP32-S3 上为 GPIO11；该输出上运行 PWM 调制器，自动改变供水泵的转速。在不使用水泵的情况下，这个输出也可以控制一个加大水流的阀门。

工作原理（以精馏模式和原版草图为例）。

蒸馏釜温度达到 **OPEN_VALVE_TANK_TEMP**（默认 77 °C）时，阀门打开，水泵启动。在最初十个调节周期内，水泵以启动值 **PWM_START_VALUE**（默认 40%）运行，之后调节器保持设定的水温，且不低于 **PWM_LOW_VALUE**。

冷却在两处设置：

1. 在配置器中（“温度” («Температуры») 和“水泵” («Насосы») 选项卡）或在 **Samovar_ini.h** 文件中——两者是同一组参数，配置器会把它们写入该文件。相关各行：

//蒸馏釜传感器达到此温度时，打开供水阀并启动供水泵

#define OPEN_VALVE_TANK_TEMP 77

蒸馏釜温度达到 77° 时，发出打开供水阀的命令，同时发送启动水泵的 PWM 控制信号，并向操作者发送需要开水的消息。无论是否使用水泵或阀门，该消息都会发送。

#define DELTA_T_CLOSE_VALVE 20

//在蒸馏和精馏模式下，蒸馏过程结束后，水泵一直运行到水温比设定的冷却水温度低这个差值为止。单位为摄氏度 C°，可以修改。强烈不建议关闭此功能。（详见[这里](#)）

//水泵 PWM 调节器工作的下限值（百分比）。需选取能保证有水流的值

#define PWM_LOW_VALUE 10

众所周知，电机启动所需的电压高于维持运转所需的电压。此行设置电机能让供水泵不停转的最低值，单位为 %。

//水泵 PWM 调节器开始工作时的值（百分比）。需选取能保证电机启动的值

#define PWM_START_VALUE 40

水泵电机能够可靠启动时的电压占额定电压的百分比。

也就是说，如果您的水泵（已装在系统中且其中有液体）在 4 伏时能可靠启动、低于 4 伏则不能启动，那末在 12 伏额定电压下这就是额定值的 30%。因此可以把 40 改为 30。

2. 在设置的“温度” («Температура») 选项卡中——水温传感器的设定值。过程中水泵会保持这个温度。

停止加热后，当水温冷却到设定值减去 **DELTA_T_CLOSE_VALVE** 并且至少已过去三分钟时，冷却关闭。另一种情况是之前已加热的蒸馏釜冷却到 70 °C 以下。在带 TCA 传感器的模式中，还会检查 TCA 的冷却情况。

在**醪塔**模式下，在网页界面中打开“**自动**” («АВТОМАТ») 按钮后可以使用自动冷却控制。在**啤酒**和**奶酪制作**模式中，水泵和阀门按程序在冷却和转移阶段开启（见[奶酪制作模式](#)）。阀门只在固件请求冷却的模式和阶段中使用。

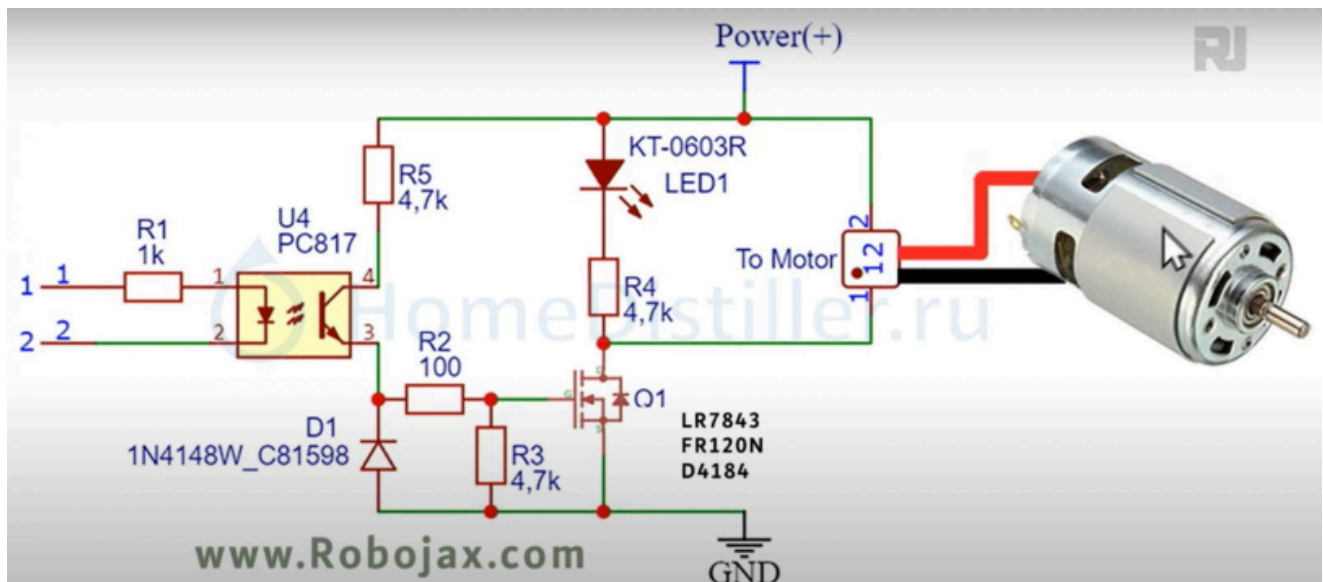
实现方式：

控制水泵（或流量阀）需要一个中间器件：固态继电器或功率板。

注意。

有些固态继电器直接接到 ESP32 上可能不动作——电压和电流都不够。

建议使用带场效应管（MOSFET）的板子[LR7843](#)



这个 MOSFET 在负极（低侧）控制电机。
虽然据称 MOSFET 几乎不发热，但最好把板子放在外壳内有空气流动的位置。

下一步：[隔膜泵](#)，[离心泵](#)和[无水泵冷却](#)

使用隔膜泵

不建议把泵直接接到有压力的自来水管上。强烈建议加一个带浮球阀的中间水箱。浮球阀简单可靠，可以多年无故障工作。为防止漏水，可以加一条应急溢流管。

隔膜泵大多能自吸，但最好还是通过接头从水箱底部取水，使泵始终有水供应。

推荐的泵：12 V 供电，功率不超过 60 W，流量 2-3.5 L/min。

注意

在 2" 塔上精馏时，耗水量约为 40-50 L/h。用 3 kW 加热管蒸馏时，耗水量更大。

最常用的就是这类泵。



虽然功率偏大，但这类型号表现良好。注意左侧的螺钉和盖子。这是压力超限时自动断电的装置，可能可调，也可能不可调，有些型号没有这个装置。

如果打算使用这种泵，Samovar 的 12 V 电源必须够用，例如 12 V 100 W。这种泵噪音较大，选择安装位置时要考虑到这一点。*

如果只打算在精馏时使用水泵，用小一些的泵就够了。它体积小，可以直接装在水箱或塔上，噪音也更低。



工作电压：12 V DC，
工作电流：0.5-0.7 A（功率须大于 6 W）
流量：1.5-2 L/min（左右），
最大吸程：78.7 英寸（2 m）

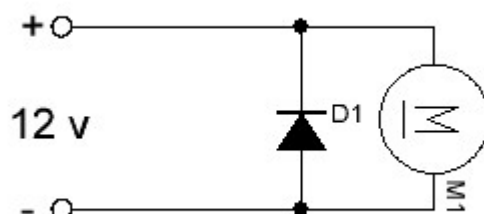
原则上，可以使用任何参数合适的 12 V 隔膜泵。

* 有一种方法可以降低噪音，让泵用起来更舒心：修改 PWM 频率。默认值设在文件 **pumppwm.h** 中；为避免改动该文件，可在 **user_config_override.h** 中添加自己的一行：

```
#define PUMP_PWM_FREQ 15
```

15 是冷却水泵默认的 PWM 频率，单位为赫兹，最高可设为 450。该参数在配置器中没有。

还需要在感性负载两端加一个续流二极管。听起来吓人，其实很简单：在尽量靠近电机的地方装一个二极管。



任何额定电流不小于电机电流的二极管都可以。快速肖特基二极管表现良好，例如 **1N5844** 或类似型号。

更多细节见[论坛帖子](#)

使用离心泵

泵的种类很多：隔膜泵、柱塞泵、齿轮泵、离心泵等等。用于我们这类用途的泵可以分为 2 类：一类在关闭时任何情况下液体都无法通过，另一类则可能让液体靠重力自流通通过。

第二类包括各种带叶轮的离心泵。

怎么区分？往进水管里吹气；如果空气能自由通过，就是这一类。

关于使用这类泵。

使用中哪些因素影响较大。

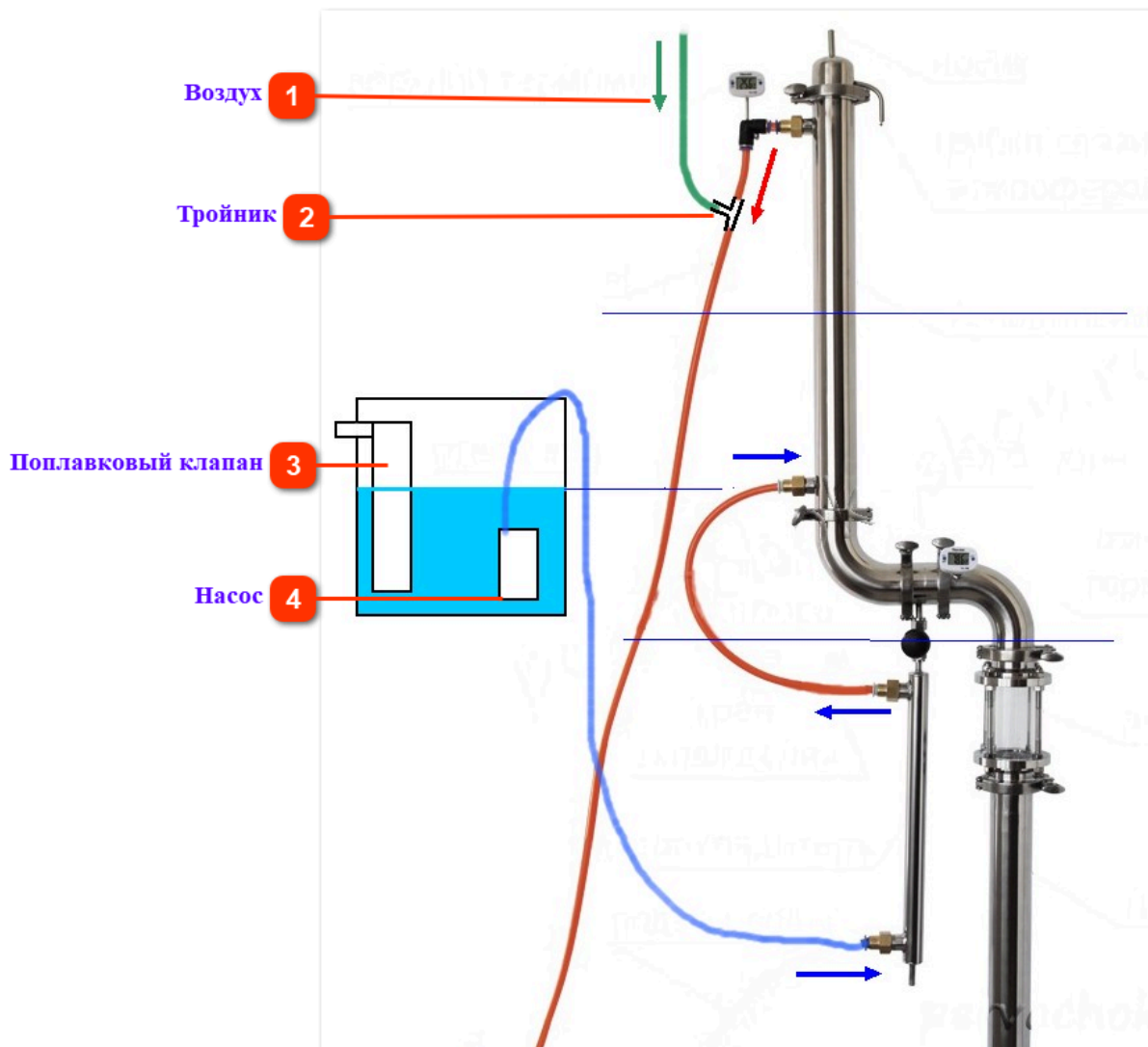
- 第一个因素是性能。如果制造商标明泵的流量为 600 l/h、扬程为 5 米，这并不意味着泵每小时能把 600 升水送到 5 米高。要么是在取水口高度上的 600 升，要么是 5 米扬程加上它能挤出来的那点水（如果还能挤出来的话）。
- 第二个因素是 PWM。它不仅决定泵输送的水量，还决定水柱能被抬升的高度。
- 第三个因素是寄生因素，即系统中可能出现的不受控的“重力自流”。

下面以一位同好使用的具体装置为例，看看如何用这类泵组织冷却。该装置在 Samovar 控制下已经运行了一年多。

出口保持温度 60°C 时，温度波动为 1-2 度。

把一个带浮球阀和潜水式鱼缸水泵的容器（3-5 升，只要能装下接头和水泵即可）抬高到分凝器的高度。

最好让浮球维持的容器水位与分凝器下接管齐平。上下偏差 10-15 cm 并不要紧。



- 1** **空气**
空气经这根管子进入，以消除不受控的“重力自流”。实际上它就是水路的 TCA（大气排气管）
- 2** **三通**
实现“断流”
- 3** **浮球阀**
使容器中的水位保持恒定。
- 4** **水泵**

使用管壳式分凝器时。

水泵把水送入后冷却器的下进口，水从后冷却器出口进入分凝器的下进口。热水从分凝器的上出口流出。

在分凝器出口装一个温度传感器，并做一个“断流”。

做法是：在排水管上装一个三通，从三通引一根管子向上。关键是三通要高于水泵所在容器的液面。

这样布置能达到什么效果。

水泵不必“知道”水要送多高；它开始提供的正是水流，而不是为了建立把水抬上去的压力，用叶轮在泵体里空搅。

PWM 真正控制了水流量。

“断流”完全消除了不受控的重力自流。

注意：

“断流”非常重要。测试中，没有断流且水泵关闭时（也就是只靠流经水泵的重力自流冷却），在 1800 瓦的加热功率下，出口水温都没有超过 40°。

用迪姆罗特冷凝器作分凝器时。

水泵把水送入后冷却器的下进口，水从后冷却器出口进入分凝器的上进口。热水从分凝器的下出口流出。

也就是说，水流与蒸汽逆向。

在分凝器的下出口装一个温度传感器，热水排水管最好先向上引到分凝器上进口的高度，再向下通往排水口。排水管向下之后，同样要做一个“断流”。做法是：在排水管上装一个三通，从三通把“水路 TCA”管子竖直向上引出。确保三通不低于水泵所在容器的液面。

离心泵遇到的这些问题，隔膜泵的用户几乎不会碰到。

唯一遇到的问题是水很冷、电机又不太听话。设定功率为 15% 时水泵停转，而设为 20% 时出口温度又太低。

顺便说一句，如果使用额定流量更大的水泵，也可能出现这种情况。

解决办法是在水泵出口直接做一个回流泄放。在水泵出水管上装一个三通，通过喷嘴（标定过的小孔）向同一个容器做固定泄放。这样可以提高最小功率值，而实际送去冷却的冷却液会更少。

受控泄放的原理适用于任何水泵。

不用水泵

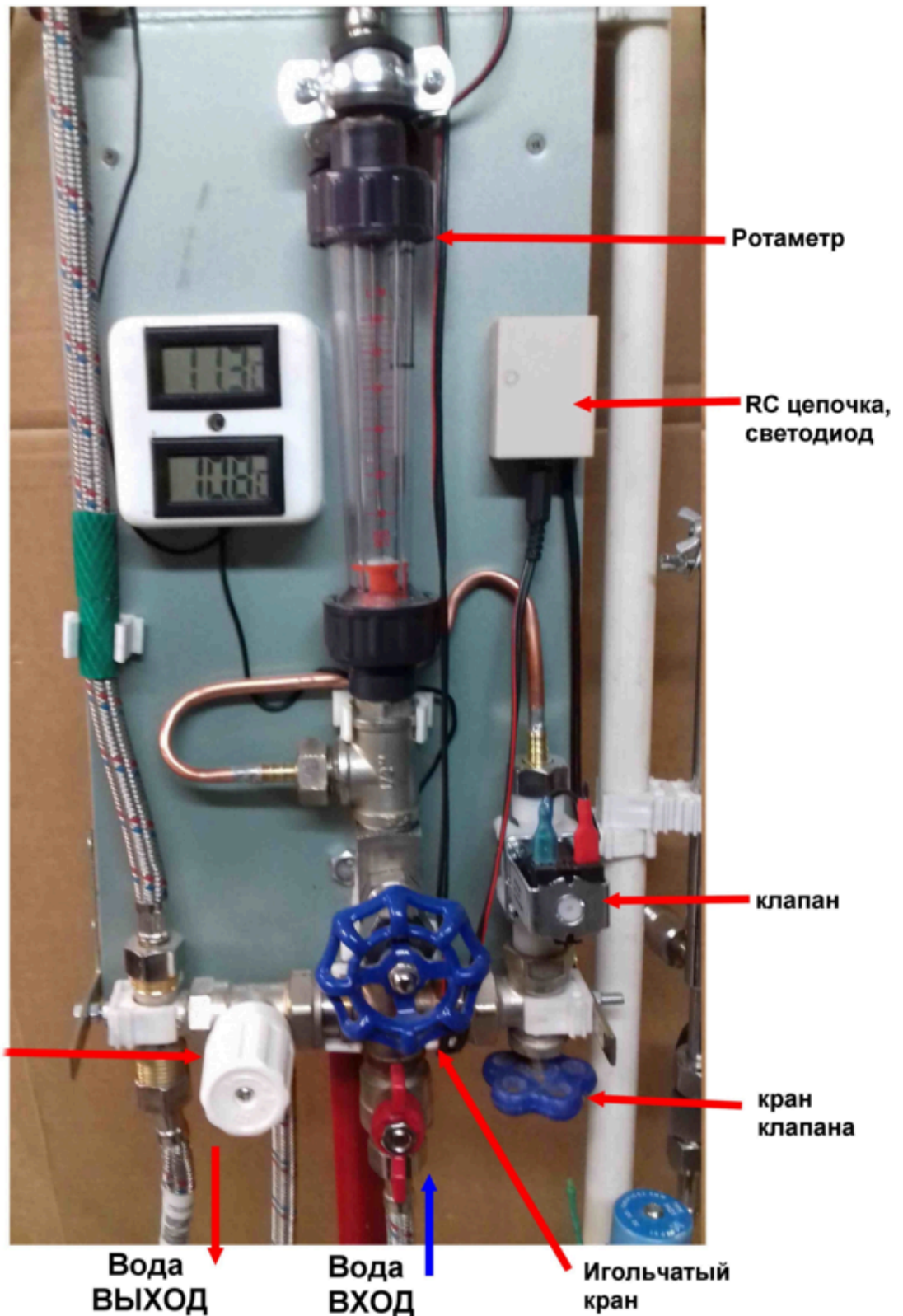
Samovar 7.00: 水泵和水阀在配置器的“泵” («Насосы») 选项卡上启用；下面的指令只在手动配置时需要。

半自动模式

这种方法的思路是：它更像一个看门狗，而且效果相当不错。

水来自自来水管网（或放在塔上方的压力水箱）。

水先进入针阀，再进入转子流量计，流量计用于调节和监视。针阀旁边做一条旁路，装另一个水龙头和一个电磁阀。



按以下步骤调整：

打开水。把针阀完全关闭。强制给电磁阀通电使其打开，用电磁阀前的水龙头设定流量，例如每小时 10 升。断电——电磁阀关闭。

旁路上的电磁阀代替水泵接到 Samovar——接到水泵控制输出（ESP32 DevKit 为 GPIO4，ESP32-S3 为 GPIO11），中间经过晶体管开关或继电器，做法见[冷却](#)页面。

在配置器的“泵” («Насосы») 选项卡上，取消勾选“控制水泵或搅拌器” («Управлять насосом воды или мешалкой») 复选框，并在“水阀控制” («Управление клапаном воды») 字段中填入 **LOW** (如果您的开关是高电平导通，则填 HIGH)。配置器把这些写入 Samovar_ini.h 文件，在那里也可以手动编辑同样的行：

```
##define USE_WATER_PUMP //использовать управление ШИМ для  
насоса для воды  
必须注释掉
```

而这一行：

```
#define USE_WATER_VALVE LOW //использовать управление  
клапаном для регулировки напора воды  
必须取消注释。只要启用了 USE_WATER_PUMP，USE_WATER_VALVE 指令就不起作用——在固件中  
二者互斥。
```

然后启动塔，用针阀调节水流量，使分凝器出口温度为比如 50°。让塔进入工作状态，期间用针阀稍微调整流量。 就这样——塔得到了冷却。

如果水温高于水温传感器的设定值，旁路上的电磁阀打开，额外每小时 10 升的水用于冷却；温度一降下来，电磁阀就关闭。

水温比设定值高出 1 °C 以上时电磁阀打开，比设定值低 1 °C 以上时关闭。精馏和蒸馏模式就是这样工作的。在啤酒模式下不使用这个输出——只打开 3 号继电器的阀门；在奶酪制作模式下，只要产品温度比目标值高出 1 °C 以上，旁路阀就保持打开。

在这种模式下系统可以运行相当长的时间。 但不应长时间无人看管，还需要定期用针阀调整流量。

回流液位传感器

Samovar 7.00: 传感器的启用和极性在配置器的“传感器” («Датчики») 选项卡上设置，并保存到 user_config_override.h。

XKC-Y25-V 非接触式液位传感器以电容式感应元件为核心，对介电常数的变化作出反应。这一特性使液位检测器能够隔着各种非金属壁感知液体：塑料、玻璃、陶瓷等。这种液体传感器甚至能感知到手的触碰。壁的最大厚度、也就是传感器的感应距离为 12 mm。隔着金属壁传感器无法工作。

更多信息见：[XKC-Y25-V 非接触式液位传感器](#)

实际上，Samovar 可以配合任何传感器工作（激光式、电感式、接触式 等），只要它能在 3.3 伏（或 5 伏，前提是使用逻辑电平转换器）下工作，并在触发时输出逻辑 1 或逻辑 0。
使用哪种传感器可以在 Samovar_ini.h 文件中选择

使用触发时输出逻辑 1 的回流液位传感器。

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

使用触发时输出逻辑 0 的回流液位传感器

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

(下面所有示例均基于使用 XKC-Y25-V P-N-P 传感器)

选择输出信号模式。

某些型号的传感器有 4 个引脚，并内置输出信号反相电路，由 MODE 引脚的状态控制。

正常模式

MODE 引脚接电源正极或悬空：OUT 信号不反相。

- 如果液位处于传感器的感应范围内：LED 亮，传感器输出为高电平。
- 如果液位处于传感器感应范围之外：LED 灭，传感器输出为低电平。

反相模式

MODE 引脚接地：OUT 信号反相。

- 如果水位处于传感器的感应范围内：LED 亮，传感器输出为低电平。
- 如果水位处于传感器感应范围之外：LED 灭，传感器输出为高电平。

装在哪里、如何固定，

传感器装在视镜上，因为它隔着金属无法工作。

第 1 阶段——临时兼长期方案。

需要准备：

- 一张塑料卡片
- 两个办公用长尾夹
- 传感器
- 一支记号笔
- 一个阶梯钻头
- 一把小刀



拿起卡片，用夹子把它夹在视镜的立柱上（照片不太清楚），标出第二根立柱的位置。剪下来，用阶梯钻头钻一个比所需略小的孔。再用小刀修整，使传感器能紧紧卡住。



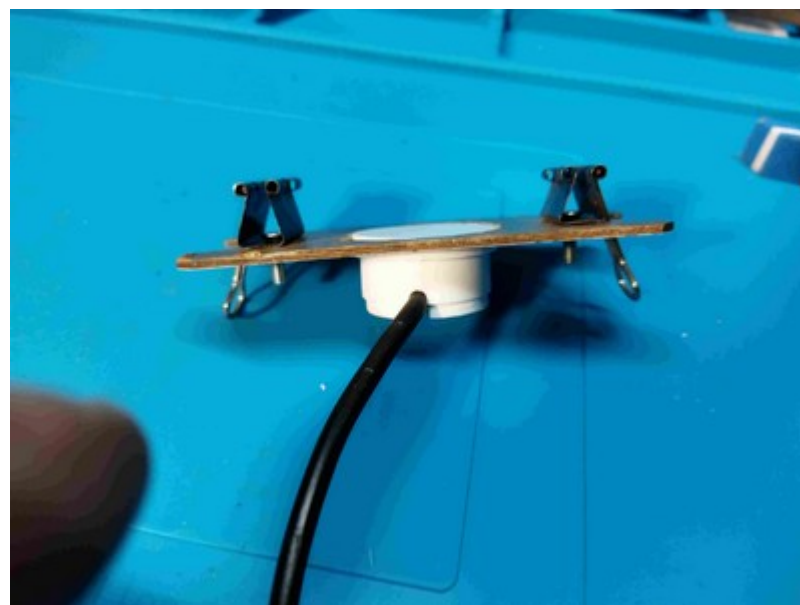
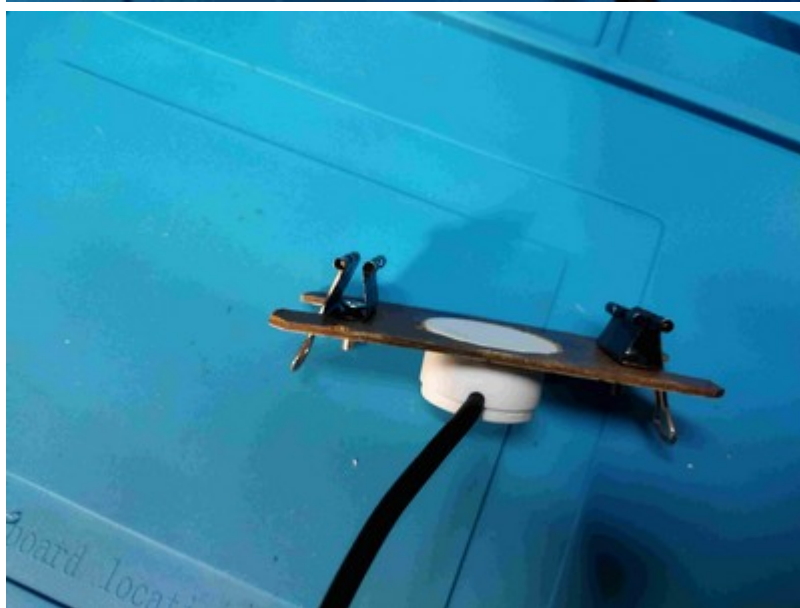
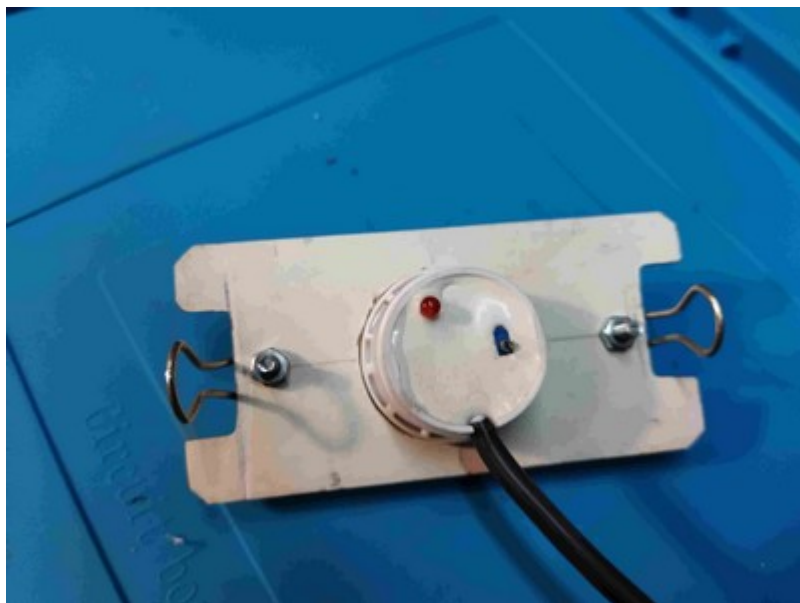
画上 + (加) 和 - (减) , 这是灵敏度调节方向; 取下传感器的盖子, 把传感器插进孔里, 用两个夹子固定在视镜上。



完成！先别装回盖子——还需要 转动微调电位器来找到触发点。

长期方案。

取一块硬塑料，用同样的方法开孔，但这次在夹子上钻孔，用 M3 沉头螺钉固定。



拆下一根弹簧，用同样的方法装到视镜上。



这样做的好处：第一，结构更牢固；第二——可以装得尽量低，甚至可以斜着装。

这是电容式传感器。调节时请用带长绝缘柄的螺丝刀，最好是全塑料或大部分为塑料的。

把传感器接到印刷电路板上的 LEVEL 接口。检查很简单——接好后把手靠近，LED 就会亮。

传感器 LED 连续亮 3 秒，Samovar 才认为回流液位传感器触发。如果 LED 闪烁的间隔短于 3 秒，不算触发。

如果 Samovar 判定传感器已触发，就命令调压器降低电压：KVIC 降 1 V，RMVK 降 2 V，SEM_AVR 降当前功率的 3%。下一次降压至少要在 40 秒之后——这个过程反应较慢。在“预液泛”（«Предзахлѐб»）程序中，传感器触发是正常现象，不会发送液泛消息。

升温后，当塔进入稳定状态、视镜也已热透时，需要设定触发阈值。

先降低传感器灵敏度，让塔进入工作状态，开始取酒头。用螺丝刀转动微调电位器的螺钉，直到传感器触发，然后再往回拧一点。

取酒头期间，可以通过提高功率让塔液泛，以确认传感器能正确触发并进行微调。这一小插曲不会影响最终产品的质量。

调得越灵敏，传感器对视镜中出现回流液的反应就越早，但也可能出现误触发。如果传感器拆下过，或塔长时间闲置，需要重新调节。

可能出现的问题。

- 传感器没有连接，但 Samovar 报告传感器已触发。

配置草图时，可以在配置器的“传感器” («Датчики») 选项卡上取消勾选“使用回流液位传感器” («Использовать датчик уровня флегмы») 复选框，或者在

Samovar_ini.h 文件中注释掉这一行：

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

或者另一种方法：

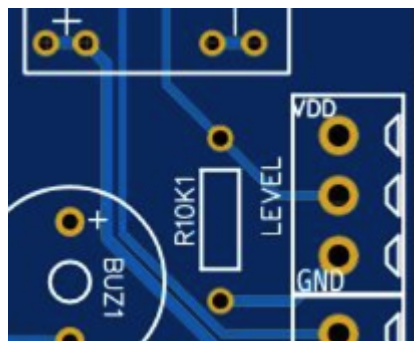
在 Samovar_ini.h 文件中注释掉这一行

```
##define IGNORE_HEAD_LEVEL_SENSOR_SETTING //Если датчик флегмы  
используется, то игнорировать команду на отключение датчика в  
WEB-интерфейсе. 之后，界面的设置 («Настройки») 选项卡上会出现一个复选
```

框，可以用它启用或停用回流传感器。

- 传感器已连接，没有触发，但 Samovar 报告传感器已触发。

使用传感器时，必须装上 R1-10k 电阻（位于 LEVEL 接口旁边）。



如果传感器触发时输出逻辑 1，液位输入端必须下拉到 GND：ESP32 DevKit 为 GPIO27，ESP32-S3 为 GPIO19。标准 DevKit 印刷电路板上已经做好了这个下拉。

如果传感器触发时输出逻辑 0，液位输入端必须上拉到 +3.3 V：ESP32 DevKit 为 GPIO27，ESP32-S3 为 GPIO19。

驱动器。步进电机。

Samovar 7.00: 电机方向和平滑加速在配置器的“步进电机” («Шаговый двигатель») 选项卡上设置。

本项目使用 A4988 或 DRV8825 步进电机驱动器（下文中步进电机简称电机）

。选哪种驱动器？读一读本文，对这个问题有个概念；两种驱动器的主要区别列在文末。

驱动器最好和 ESP32 一样装在排针座上。

设置驱动器之前，ESP32 必须已装在电路板上，并已上传 Samovar 草图。

不要接电源和电机。

拿起驱动器，对照驱动器板和印刷电路板上的引脚标记，把它插进排针座。

用万用表的通断档检查。GND 引脚必须接地，VMOT 必须接到 VIN 电源接口的 +12 伏，VDD 必须接到 ESP32 第一个引脚的 +3.3。

给 VIN 接口接上 12 伏电源。可以再检查一遍各电压是否到位。

印刷电路板上供驱动器使用的开关：

- H1 - 三个引脚全部接通

- SW1 - 如果有拨码开关，全部拨到 OFF 位置。如果没有开关——先不要装跳线帽。

现在需要设置驱动器。

不要接电机。

驱动器是个娇气的东西，甚至有人建议断电后再调节。

也就是说：把微调电阻转一点，通电，测电压，断电——再转。如此反复，直到调到需要的值。

还有一种更快的方法。

准备一把带绝缘柄的螺丝刀。把万用表的一支表笔接到印刷电路板上任意接地点（GND），另一支表笔要么小心地碰着微调电阻、同时用螺丝刀转动滑片，要么在表笔上套一个“鳄鱼夹”，夹在螺丝刀头上，隔着夹子转动螺丝刀。

调节步骤见以下链接。

[如何设置 DRV8825 步进电机驱动器](#)

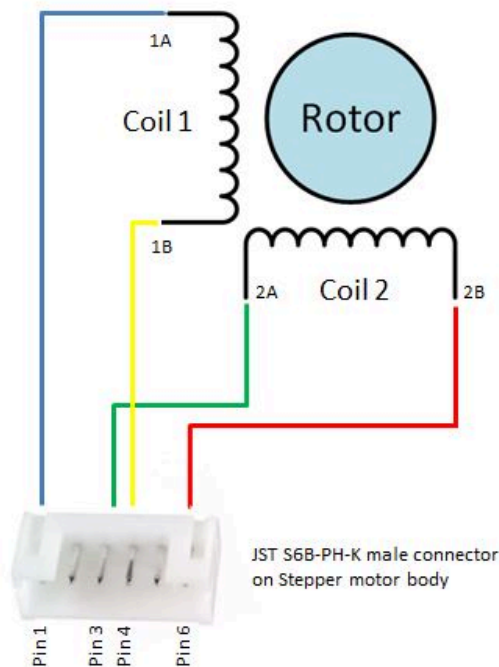
[如何设置 A4988 步进电机驱动器](#)

连接步进电机。

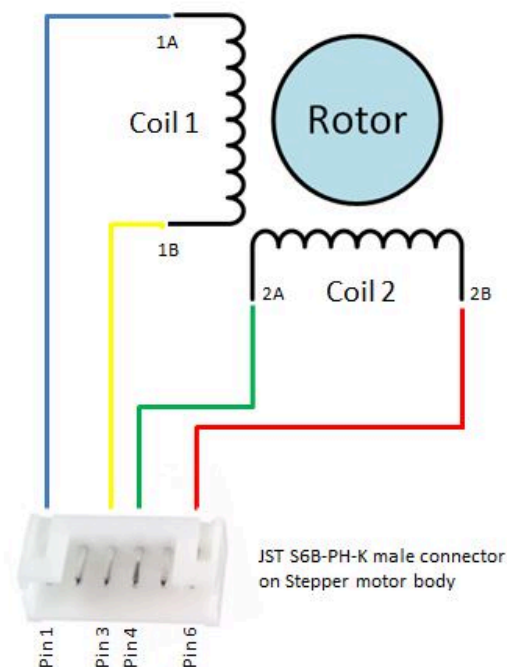
本项目使用双极 NEMA17 步进电机。先检查电机是否符合本项目要求。双极电机有两组独立的绕组。不同厂家电机的引脚排列可能不同。

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

Crossover



Straight



把套件附带的带线插头接到电机上，任取一根线，用万用表通断档 找出与它配对的线。这就是一组绕组。把这两根线做上标记。再检查第二对。蜂鸣器响，就没问题。

现在从一对中任取一根线，再从第二对中任取一根，测一下。它们之间不应该响。如果响了，说明这不是双极电机。

现在把电机接到印刷电路板上的接口。 按照上图，1A-1B 是一组绕组，2A-2B 是另一组。"A" 是绕组的始端，"B" 是末端。当然可以严格按标记连接，但没有必要。

关键是把一组绕组接到驱动器的 1A-1B 引脚，另一组接到 2A-2B。哪组接哪边无所谓，绕组的始端 和末端也无所谓。

接好了。通电。电机不转。打开网页界面，发出取酒命令。

电机轴应该开始转动。

电机嗡嗡响却不转；电机转动时一顿一顿

- 检查电机与驱动器的连接是否正确，以及是否接上了。是接到驱动器本身，而不只是印刷电路板。
- 检查驱动器的 RST 和 SLP 引脚上是否有 +3.3 伏。
- 在网页界面的设置页面“泵” («Насос») 选项卡上，点击“泵校准” («Калибровка насоса») 。如果“步进电机速度” («Скорость шагового двигателя») 字段为 8000 (或类似数值)，改为 800
([论坛帖子](#) :我在调试 Samovar 时也遇到过类似情况，区别是用的 TMC2208 驱动器。最后通过选择步距细分解决了问题：泵只有在 1/8 时工作稳定，其他模式下要么嗡嗡响，要么一顿一顿地转、严重丢步 (不过在另一台设备上，这个驱动器和泵在所有细分下都能工作，唯一的区别是控制信号为 5 V 而不是 3.3 V) 。也许这与 STEP 脉冲的频率有关；那就试着在“步进电机校准” («Калибровка шагового двигателя») 窗口中调整“步进电机速度” («скорость шагового двигателя») 参数 (比如设为 4000 而不是 8000) ，然后按 100 ml 校准。
记住：选择电机工作模式时，Samovar 印刷电路板必须完全断电。)

如果电机轴转向不对，在接到电路板的地方把任意一组绕组的两根线对调。

注意。

文件 `Samovar_ini.h` 中有一行可以用软件改变转向：

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

在配置器中，这是“步进电机” («Шаговый двигатель») 选项卡上的“步进电机反向” («Обратное направление шагового двигателя») 复选框；同一选项卡上还有“步进电机平滑加速” («Плавный разгон шагового двигателя») (`USE_STEPPER_ACCELERATION`) 。不过一次性对调接线，比每次更新时都留意这个参数更方便。

通过取酒程序让电机以不同速度运行。把取酒速度设为 70-150 ml/h。一只手握住电机外壳，另一只手试着让轴停下；如果轴很容易被停住，就回到驱动器设置，把电压稍微调高一点，真的只要 0.02-0.04 伏。

再检查一下轴是否容易被停住。

把套件附带的散热片粘到驱动器芯片上。

现在可以装泵头了。先用中性硅脂润滑滚轮和软管。以后需要定期润滑 (每 5-7 次蒸馏一次)，这样能延长滚轮和软管的寿命，电机也更轻松。

带着泵运行一下，看看有没有停顿、电机听起来是否过载、电机和驱动器芯片散热片有多烫.....

如有需要，现在可以进行精细调整。

一点理论。

与传统的直流和交流电机不同，步进电机以离散的步或增量运动，因此得名“步进”。

步进电机的每一步对应一个精确的转角。步进电机的步距角通常为 1.8° 或 0.9°，分别对应一整圈分为 200 步或 400 步。

标准 NEMA17 电机每步转角为 1.8°（每整圈 200 步）

电机可以工作在几种模式下——整步和微步。

整步模式是指电机每 1 步使轴转动 1.8°

整步模式下电机的扭矩在所有模式中最大。

微步 是把标准步分成若干个微步。标准步可以分为 2、4、8、16 和 32 份
步距细分可以用 SW1 开关设置

MS1, MS2, MS3——微步分辨率选择引脚。用它们可以选择五种模式之一：

对于 A4988 驱动器			
MS1	MS2	MS2	步距
off	off	off	整步
on	off	off	1/2
off	on	off	1/4
on	on	off	1/8
on	on	on	1/16

对于 DRV8825 驱动器			
MS1	MS2	MS2	步距
off	off	off	整步
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

说明。

假设选择了 1/4 模式。那么每 1 个微步电机轴转动 $1.8^{\circ}/4 = 0.45^{\circ}$ ，相当于轴每转 1 整圈需要 $200 \times 4 = 800$ （或者换个算法， $360^{\circ}/0.45^{\circ} = 800$ ）个微步。

微步模式下电机扭矩降低，但平稳性和精度提高（这对本项目并不特别重要）

步进电机在整步模式下振动和噪音较大。

微步能显著降低振动和噪音，使电机运转更平稳、更安静。

总之，观察并倾听您的电机如何工作。可以试试不同模式并加以比较。

注意。

选择电机工作模式时，Samovar 印刷电路板必须完全断电。

确定电机工作模式、把拨码开关拨到相应位置（或最终焊好跳线）之后，需要校准泵。每次改变电机工作模式后都必须重新校准。

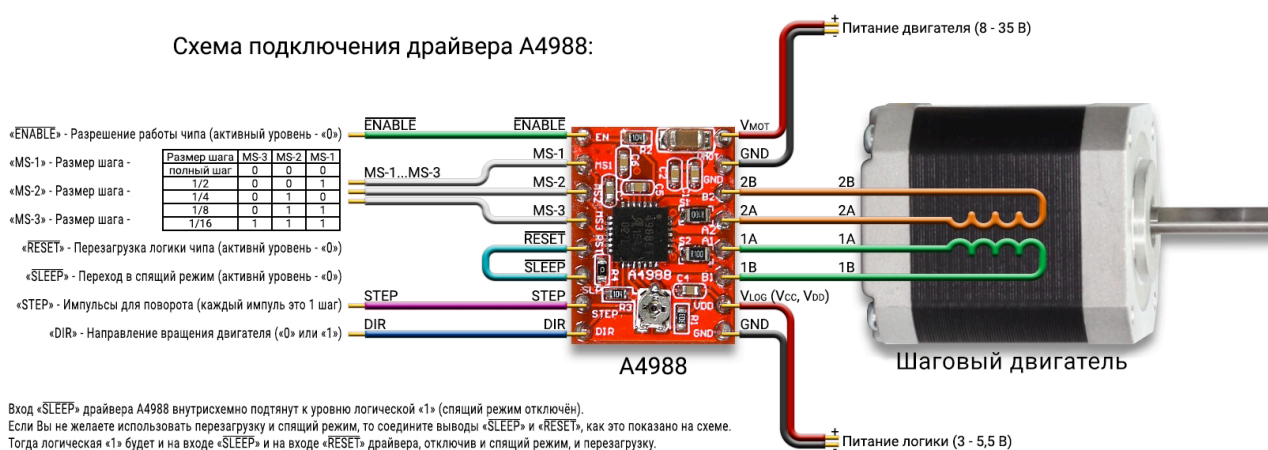
取酒精度还受泵润滑是否规律的影响。润滑越少，实际取酒速度越低。

DRV8825 与 A4988 电机驱动器的区别 (简单说，不涉及具体细节)

- 设置限流的电位器位置不同。
- DRV8825 驱动器在微步模式下支持 1/32 步。
- A4988 驱动器最多支持 1/16 步。
- DRV8825 驱动器不加散热片时每相最高 1.5 A，
- A4988 驱动器不加散热片时每相 1 A。
- A4988 驱动器比 DRV8825 驱动器便宜。

参考资料。

Схема подключения драйвера A4988:



分馏分配

Samovar 7.00: 舵机 11 个位置的修正值在配置器“设备” («Оборудование») 选项卡上的“舵机修正值 (11 个数)” («Поправки сервопривода (11 чисел)») 字段中设置；配置器会把它们写入 Samovar_ini.h 文件中的 servoDelta 数组。

Samovar 可以按照您编写的取酒程序，把整个运行过程分成 11 份。

注意。

如果不需要分份，或者您自己更换容器，就不需要这个装置。没有它 Samovar 也能工作。

制作时需要一个 Arduino 项目中常用的那种舵机。

舵机 [SG90](#)

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



警告！

Samovar 使用**旋转角度为 180° 的舵机**，其他舵机都无法正确工作（舵机的旋转角度有 90°、180°、360° 或连续旋转几种）。

购买前请仔细挑选舵机。如有疑问，最好放弃，改从明确标注旋转角度的其他卖家购买。

导线定义：红色 — +5 V，棕色 — GND，黄色或橙色 — 控制信号。信号输入接到 ESP32 DevKit 的 GPIO25 或 ESP32-S3 的 GPIO10。

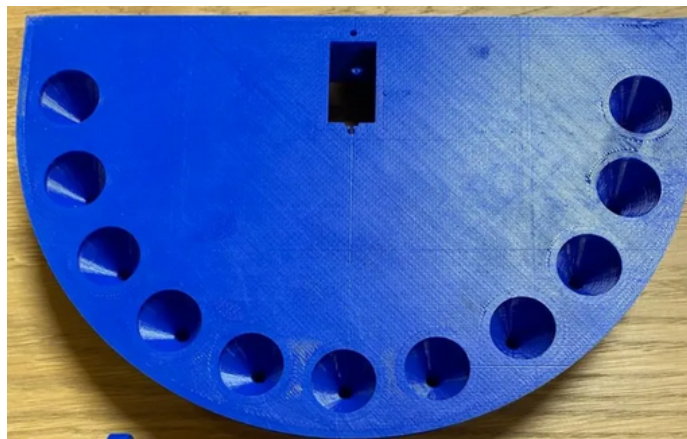
检查舵机旋转角度可以使用一个专门的[草图 \(sketch\)](#)。把该草图放在与 Samovar 文件相同的目录中 (... Documents\Arduino\...)，并代替 Samovar 安装到 ESP32 上。安装草图的方法见[用户手册](#)。

草图安装到 ESP32 后，舵机应开始从初始位置转到极限位置再转回来，循环往复。检查完舵机后，需要把 Samovar 草图重新上传到 ESP32。

为 SG90 舵机设计了一个可 3D 打印的收集转台模型。模型文件 在 3D 目录中，每个 Samovar 发布压缩包
里都有。

这些文件是 **Samovar_plate.stl** 和 **Samovar_horn.stl**。

。请用耐丙酮和酒精的塑料打印，例如 PLA。



SG90 是最便宜、最常见的舵机之一。有些用户遇到过它工作不稳定的问题。

表现为舵机很难到达所需位置，或者某个时刻摇臂开始“抖动”。经验表明，这与成品输送管的位置不合适、尺寸过大、过硬或者相反过软有关。

SG90 可以换成：

舵机 [MG996R](#)，MG996、MG995。



这种舵机比前一种贵，但功率也更大（因此耗电更多），减速齿轮可以是塑料的也可以是金属的（对我们的用途没有区别），附带的配件也更多。

Samovar 使用旋转角度为 180° 的舵机，其他舵机都无法正确工作。

导线定义：红色 — +5 V，棕色、灰色或黑色 — GND，黄色或橙色 — 控制信号。信号输入接到 ESP32 DevKit 的 GPIO25 或 ESP32-S3 的 GPIO10。

也为这些舵机设计了可 3D 打印的收集转台模型。这些文件是 **plate_996_servo.stl** 和 **horn_996_servo.stl**。模型文件 在 3D 目录中，该目录在 Samovar 发布压缩包里。请用耐丙酮和酒精的塑料打印。

收集转台的制作和改装示例：



见论坛：[帖子 1](#), [帖子 2](#)

位置修正

每个舵机都有自己的误差，需要针对转台位置单独 调整。

如果收集转台是打印的或按模板制作的，漏斗之间距离相等，就需要修正旋转角度。

舵机旋转角度设置 在文件 Samovar_ini.h 中 (见[上传和更新草图](#)和[草图设置](#)),
相关各行为:

//舵机旋转角度修正值。

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

花括号中的值是每个位置的角度修正值,共 11 个。接收漏斗从零号位置开始计数。修正值以度为单位。每个位置单独修正,与相邻位置的修正无关。

修正时,为“精馏”(«Ректификация»)模式编写一个测试用的取酒程序

```
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
```

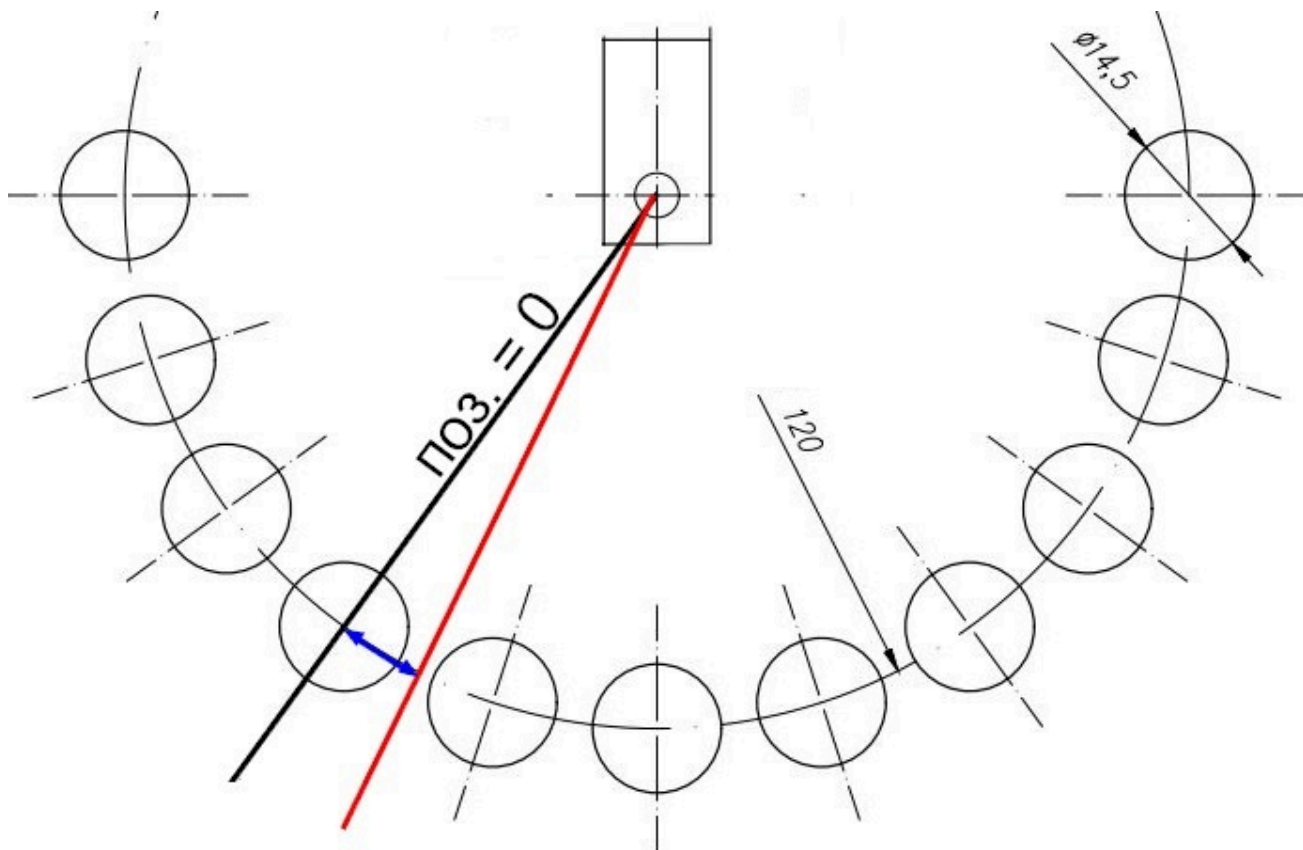
复制这段文字,粘贴到 Samovar 网页界面主页“程序”(«Программа»)选项卡上的取酒[程序文本框](#)中,然后保存。

可以先不在舵机上装摇臂,或者装成便于取下的方式。连接舵机并开启 Samovar。启动时 Samovar 会把舵机转到起始位置。安装摇臂,使其指向零号漏斗。

从 Samovar 发出“开启加热”(«Включить нагрев»)命令(加热器必须断开),然后发出“开始取酒”(«Начать отбор»)命令。摇臂停在零号漏斗处。

“下一个程序”(«Следующая программа»)命令把摇臂移到下一个位置——1 号漏斗。如果摇臂位置与穿过舵机转轴中心和接收漏斗中心的假想线重合,那就好,这就是零,不需要修正。再次发出“下一个程序”命令,摇臂移到下一个位置。

如果摇臂没有到达所需位置或者转过了头,就需要修正。以 3 号漏斗为例(记住漏斗编号从零开始)。原版草图中这个漏斗的修正值为 4 (负 4)



假设摇臂“转过了头”，停在红线上。测量零位与实际位置之间的弧长（蓝色箭头）。

比如说是 10 mm。

现在要求出旋转 1 度对应的弧长。在网上找一个 [计算器](#)。输入半径值（这里是 120），角度设为 1。

得出旋转 1° 时弧长为 2 mm。我们的摇臂离所需位置 10 mm。

$10/2=5$ 度。

摇臂转过了头，需要往回转，所以这是负值。草图中已经有 -4（负 4）

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

因此 3 号漏斗（即 4 号位置）的旋转角度修正值为负 9°

这适用于特定的舵机和转台，漏斗分布在半径为 120 mm 的圆上。

用同样的方法检查摇臂的所有位置，并修正这一行花括号中的值

`int8_t servoDelta[11] = {0, -2, -3, -9, -3, -2, 0, 0, 0, 0, -2};`

修正后，把草图重新上传到 Samovar，设置测试取酒程序，检查摇臂在各位置之间的移动。

自己制作收集转台

漏斗位置固定的收集转台。见[论坛上的帖子](#)。制作模板也可以在那里下载。

几乎不需要修正舵机旋转角度值的收集转台。



要点在于，钻孔的中心是按照摇臂的实际位置来标记的。
制作说明[见论坛](#)。

漏斗



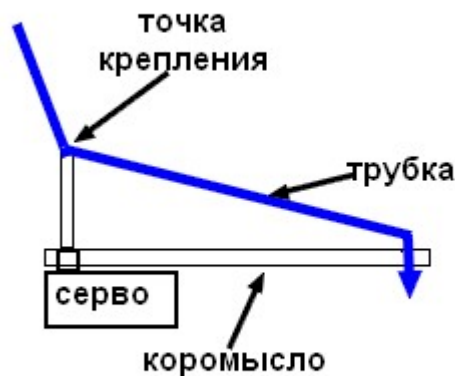
- 购买：
[带可拆卸滤网的金属厨房迷你漏斗，5 个装。](#)
[SUEF 小号塑料油漏斗，2/5/10/20 个装，液体油漏斗。](#)
[带可拆卸滤网的不锈钢厨房迷你漏斗，5 个装](#)
-
- 用 3D 打印机制作——漏斗文件在 3D 目录中。

- 可以使用**药瓶盖**或医用**注射器**。不建议使用小于 5 ml 的注射器——可能会溢出；10 或 20 ml 的更好。如果接收漏斗溢出，最简单的办法是把接收容器尽量放低，把转台抬高。从漏斗到容器的管中的液柱会给液流增加额外的速度。

排查可能出现的问题。

假设舵机是好的

- 舵机工作时 Samovar 重启 - 可能是 12-5 伏转换器功率不够，或者舵机电机电流太大。可以这样做：
 - 作为临时措施，加大电容 C4 的容量。愿意的话 C3 也可以加大。
 - 增加第二个 12-5 伏转换器。连通地线，用它给舵机供 +5（所有 5 伏用电设备都可以由这个转换器供电）
- 即使没有接管子，摇臂在漏斗上方也“抖动” - 摇臂的水平刚度不足。
- 接上供液管后，摇臂在漏斗上方“摆动”。
 - 管子太硬，一直把摇臂往回“拉”，舵机则不断试图把它转回原位。
 - 管子在转台上方“晃动”，把摇臂一会儿拉向这边，一会儿拉向那边。可以选择在舵机减速器转轴的中心处再把管子固定一下。



紧急按钮。附加紧急传感器

已按 Samovar 7.00 核对。

把按钮或传感器接在紧急输入端与 GND 之间。低电平触发：ESP32 DevKit 为 GPIO35，ESP32-S3 为 GPIO48。

触发后，Samovar 关闭加热、供水、取酒和搅拌器，并在开发板重启之前禁止再次启动。多个常开按钮和传感器可以并联连接。对于 ESP32 DevKit，请使用 PlatformIO 环境 **Samovar_alarm_button**，或在 **user_config_override.h** 中添加 **#define USE_ALARM_BTN**。在 **Samovar_s3** 环境中，GPIO48 紧急输入已默认启用。

酒精泄漏传感器。 [起始](#)。 [实现](#)。

ESP32 DevKit 的紧急输入为 GPIO35，ESP32-S3 为 GPIO48。两种情况下，输入端对 GND 短接时触发。

经典 ESP32 的 GPIO35 需要一个外部 10 kOhm 上拉电阻接到 +3.3 V：GPIO34–39 引脚没有内部上拉。ESP32-S3 的 GPIO48 由固件启用内部上拉。

在 Samovar 1.3.2 板上，紧急按钮接口标记为 ALRM_BT（较早的资料中也称为 ALARM_BTN）。它连接到 GPIO35，必须有一个 10 kOhm 上拉电阻接到 +3.3 V。

帖子链接：[链接 1](#)，以及 ais77 的 [链接 2](#)、[链接 3](#)。

推荐配置

推荐配置在最小配置的基础上增加了自动运行所需的执行机构：

- 用于控制加热的功率调节器和接触器；
- 供水阀，必要时再加冷却水泵；
- 用于取酒的步进电机及其驱动器；
- 蒸馏釜、蒸汽、塔节和水的温度传感器；
- 大气压力传感器；
- 回流液位传感器或您使用的其他报警传感器。

LCD 显示屏、编码器和独立按钮便于本地控制，但不是必需的：主要设置和控制都可以通过网页界面完成。

LCD 显示屏、编码器、按钮

Samovar 7.00: 控制按钮在配置器的“设备” («Оборудование») 选项卡中启用。

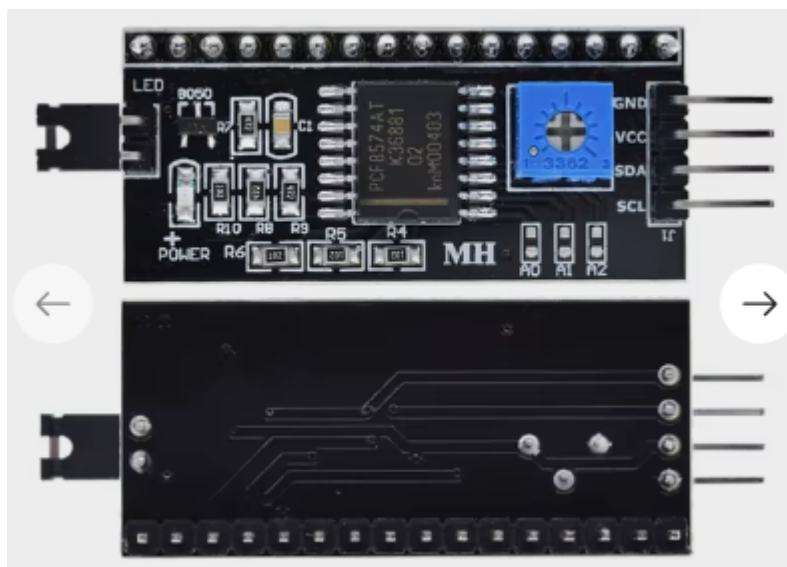
注意。

Samovar 没有显示屏、编码器和通用按钮也能工作。所有信息都可以在网页界面中查看，如果启用了集成，也可以在 Blynk 中查看。在设备旁边时显示屏很方便；没有显示屏时通常用不到编码器，而单独的按钮可能会有用。



项目使用四行显示屏：20X4 LCD2004，带I2C模块。没有该模块就无法把显示屏接到 Samovar。

这是 I2C 模块



这是焊好I2C模块的 20x4 显示屏。



可以购买已焊好模块的显示屏，也可以分别购买成套件自己焊接。
显示屏背光可以是蓝色、绿色、黄色或白色（这会影响价格）。如果不需要背光或背光碍事，可以在调好显示屏后拔掉 I2C 模块上的 LED 跳线帽将其关闭，文字仍然可以看清。
大多数显示屏只支持拉丁字体，部分支持西里尔字母。Samovar 使用拉丁字体。

连接。

显示屏用 I2C 模块引出的 4 根线，按模块和主板的引脚定义接到 Samovar 主板上。
通常第一次上电时背光亮了，但没有文字。旋转 I2C 模块上的微调电阻，文字就会出现。把对比度调到最高。
显示屏的设置就完成了。

注意。

如果显示屏上某些行或字符显示不正确，请仔细重焊模块与显示屏之间的连接，并清除助焊剂。

编码器

项目使用带“转接板”的 EC11 编码器。它的样子如下。



它用 5 根线按引脚定义接到 Samovar 主板上。它的主要作用是在 LCD2004 显示屏上“滚动”各行。

如果编码器滚动显示行的方向反了，请对调 S1 和 S2 触点上的导线。

按钮。

Samovar 对按下按钮的反应取决于所选的工作模式（见[LCD 显示屏和通用按钮](#)）。

普通的常开轻触按钮即可。把它接在输入端与 +3.3 V 之间：ESP32 DevKit 上为 GPIO39，ESP32-S3 上为 GPIO47。输入端需要一个外接的 10 kOhm 下拉电阻接到 GND。

如果不使用按钮，ESP32 DevKit 上不需要这个电阻，但必须在配置器的“设备”（«Оборудование»）选项卡中取消勾选“使用控制按钮”（«Использовать кнопку управления»），或者在 Samovar_ini.h 文件中注释掉按钮：

```
///define USE_BTN //使用按钮
```

在 ESP32-S3 上，按钮输入 GPIO47 无论此设置如何始终启用，因此即使没有按钮也需要接 10 kOhm 电阻到 GND，否则可能出现误触发。

警告！

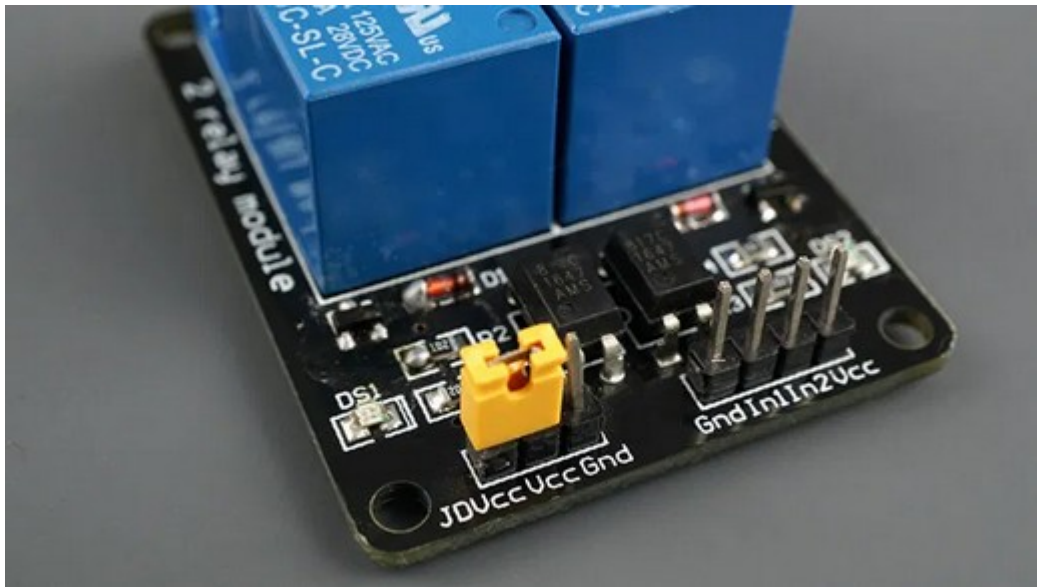
启动时按住单独的按钮约两秒钟会启动 Wi-Fi 接入点。按住编码器按钮约两秒钟只会删除保存的 Wi-Fi 设置，其他硬件设置不会被重置

如何正确连接继电器模块

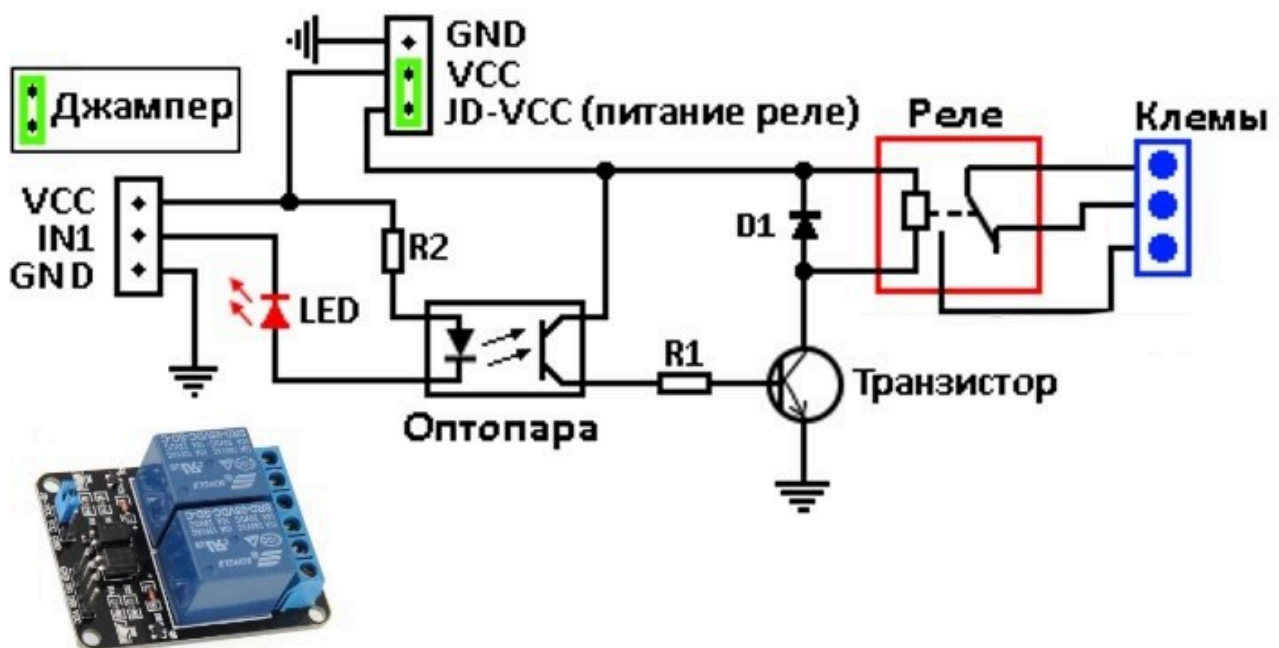
ESP32 可以连接 5 伏或 12 伏的继电器模块。
模块上必须有 JDVCC - VCC 跳线帽。

警告：

如果没有 JDVCC-VCC 跳线帽，说明模块的供电没有电气隔离。不建议使用这类模块。



电路图（以一个通道为例）：



与 ESP32 的连接：

图中上方的接口：

必须！拔掉跳线帽！

- GND 引脚接继电器电源的负极（-5 或 -12 V，取决于所用继电器）。
- VCC 引脚不使用（即不接任何东西）。
- JDVCC 引脚接继电器电源的正极（+5 或 +12 V）。

第二个接口：

- VCC 接 ESP32 的 +3.3 伏
- IN1、IN2 接 ESP32 相应的输出。为实现完全的电气隔离，这个接口的 GND 引脚可以不接 ESP32 的 GND。

这样整个模块（包括继电器）由单独的电源（5 或 12 伏）供电，Samovar 只提供 3.3 伏：供给电阻 R2、光耦的 LED 和指示 LED。

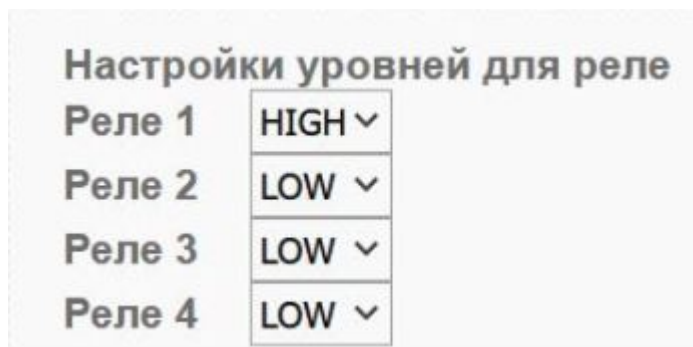
IN1 引脚为逻辑 0 时继电器吸合；为逻辑 1（+3.3 伏）时继电器释放

注意。

大多数继电器模块属于低电平触发型（Low Level Trigger，LLT，默认启用）。也就是说，要让继电器吸合，继电器控制输入（IN1）必须为逻辑“0”。输入（IN1）加逻辑“1”时，继电器释放。当单片机要控制很多设备时会使用这种继电器，以减少单片机的发热和电流消耗。在要求上电时或“默认”就让继电器吸合的系统中也会使用它们。

要用逻辑“1”而不是逻辑“0”来控制模块：

- 在 Samovar 网页界面的 [“设置 – 其他”](#)（«Настройки – Прочие»）选项卡中，选择模块所需的触发电平。



字段 **rele1...rele4** 设置继电器吸合时的电平。对于本文所述的逻辑 0 吸合的模块，将电平设为 **LOW**。

警告。在 ESP32 DevKit 上，2 号继电器（搅拌机）接在 GPIO15 上。这是一个启动配置引脚（strapping pin），开发板复位时硬件会把它拉高。如果在这个通道上使用逻辑 1 吸合的模块，每次 Samovar 启动时它都会短暂吸合。2 号继电器请使用低电平（LOW）吸合的模块。

扩展配置

扩展配置以推荐配置为基础，增加了用于特定模式和场景的硬件。

可以连接 [带编码器的 LCD 显示屏](#)、[用于分馏分配的舵机](#)、一个或两个用于搅拌器和水泵的 [I2CStepper](#) 模块、[紧急按钮](#)、附加传感器、[外部测量仪](#)以及 [NBK](#)（连续醪塔）硬件。只连接在所选编译环境中已启用、并在 `Samovar_ini.h` 或 `user_config_override.h` 中已配置的部分（最简单的方法是通过配置器，它由 `flash_windows.bat` 或 `flash_macos_linux.sh`）。

奶酪制作硬件

Samovar 7.00: [奶酪制作](#)模式使用开发板的标准输出，不需要单独的接口：

pH 传感器为 PH-4502C 模拟模块。其输出接到模拟输入 `LUA_PIN`：ESP32 DevKit 上为 `GPIO34`，ESP32-S3 上为 `GPIO4`。MPX5010DP 压力传感器和 Lua 的 `analogRead()` 函数使用同一个输入，因此它们不能与 pH 传感器同时工作。读数按校准换算为 pH，见 [pH 校准](#)。

搅拌器为 2 号继电器（与啤酒模式中的搅拌器相同），或地址为 1 的 `I2CStepper` 模块的搅拌器；具体用哪个在程序行中设置。

排液阀为 4 号继电器。在其他模式中，此输出控制辅助加热管，因此只能在它上面接阀门或加热管其中之一。

冷却由水泵或阀门完成，与其他模式相同；如果没有 `USE_WATER_PUMP` 或 `USE_WATER_VALVE`，奶酪制作程序中的冷却阶段将无法通过校验。

测量仪

dranek 制作的测量仪是一台独立的外部设备，用于附加测量。Samovar 的基本运行不需要它，它也不属于 Samovar 固件：Samovar 不读取它的数值。

测量仪的源代码位于 Samovar 仓库的 **pro_mini_ntc** 目录中（一个 ESP8266 草图 (sketch)、包含电路图和 Gerber 文件的“Схема и плата”（“电路图和电路板”）文件夹，以及包含说明的“Справочная информация”（“参考资料”）文件夹）。该目录不参与 Samovar 固件的编译。

[论坛上关于测量仪的说明、电路图和讨论](#)。制作时请使用作者原帖中的电路。

I2CStepper

I2CStepper 是一个基于 Arduino Nano 的独立模块，自带步进电机、四个继电器、一个外部传感器、一块 16×2 LCD 和一个编码器。当前固件使用 I2C v3 协议。

该模块可以作为独立设备使用：既可以单独通过 16x2 LCD 屏幕上的菜单和编码器控制，也可以接到 Samovar 上控制。

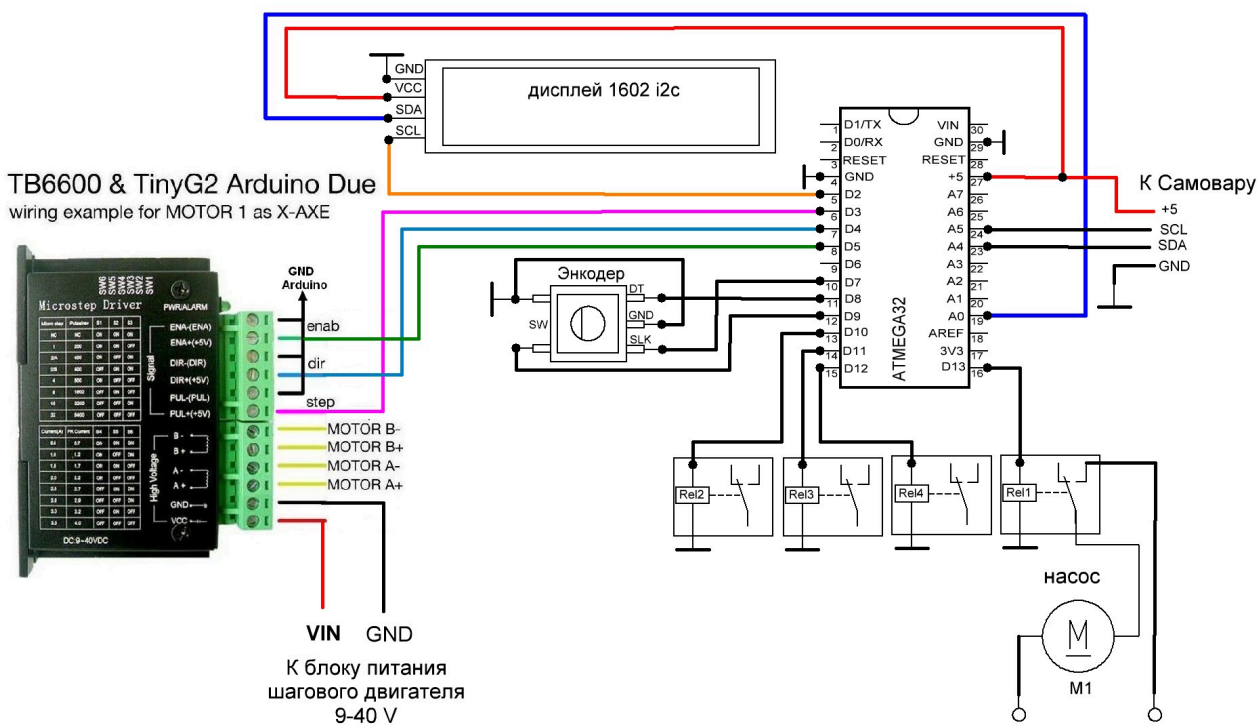
扩展器通过 I2C 总线用四根线接到 Samovar（与 LCD 屏幕相同）。Samovar 会识别出已连接的扩展器，并可以控制它。

Samovar 最多支持十块地址为 1–10 的板：奇数地址为搅拌器，偶数地址为泵/灌装板。每种类型中使用地址最小的那块板。搅拌器按转速、运行时间和暂停工作；泵按 ml/h 工作；灌装模式按设定的体积和速度工作。

如果设备报告支持继电器，可以在 I2CStepper 页面上切换第一个继电器，在 Lua 中可以切换全部四个。继电器的用途由用户自行决定：Samovar 固件不做规定。

Samovar 启动时在总线上搜索模块。如果找到模块，在蒸馏 («Дистилляция»)、醪塔 («Бражная колонна») 和啤酒 («Пиво») 模式的页面上会出现“I2CStepper”选项卡；它打开该板的手动控制页面：速度、步数、启动和停止、继电器 1。板自身的设置 (地址、模式、速度、搅拌机运行时间和暂停、泵的进料量、灌装体积和速度、每 1 ml 的步数) 在设置 («Настройки») → “I2CStepper”选项卡中编辑，并保存在 Nano 的存储器中。外部泵校准通过 I2CStepper 页面上的“外部泵校准” («Калибровка внешнего насоса») 按钮启动；结果写入该板的“步/毫升” («Шагов/мл») 字段。

接线图



外设连接引脚：

```

// I2C Master 引脚——连接屏幕以及可能的其他设备
// SDA_PIN 0 //A0
// SCL_PIN 2 //D2

// I2C Slave 引脚——与 Samovar 通信
// SDA_PIN A4
// SCL_PIN A5

// 步进电机引脚
#define STEPPER_STEP 3
#define STEPPER_DIR 4
#define STEPPER_EN 5

//继电器引脚
#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

// 编码器引脚
#define ENC_CLK 7 //S2
#define ENC_DT 8 //S1
#define ENC_SW 9 //KEY

```

提示。

如果打算把扩展器作为单独的设备装在自己的外壳里，用 USB A-B 电缆（电脑—打印机用的那种）把它接到 Samovar 比较合理。模块的 +/- 5 伏电源和两路 I2C 总线信号直接取自 Samovar 的显示屏接口。Samovar 外壳上装一个 USB-B 插座。

扩展器外壳上装一个 USB-A 插座。

这样接到 Samovar 时，Arduino Nano 同时获得电源和 I2C 总线。作为独立设备使用时，用单独的电源代替 Samovar，例如手机充电器。

如果使用内置电源，在电源输出端装一个 USB-B 插座，用同样的 USB A-B 电缆供电，这样就不会误把 +5 伏送进 Samovar。

外设。

继电器模块使用 5 伏的，最好带光耦控制。也可以使用 12 伏继电器，[接法](#)与 Samovar 中的原理相同

可以使用任何泵：直流电机，或交流单相、三相电机均可。因为泵电机是先经继电器、再经任意功率开关（接触器或固态继电器）启停的。小容量可以考虑 15RM 磁力驱动泵：不锈钢、安静、出力好。

搅拌器或 NBK（连续醪塔）可以使用任何双极步进电机，只要功率和扭矩足够。推荐 Nema 23 配合适的驱动器。

当前的 Arduino Nano 固件和 v3 协议说明见[I2CStepper 仓库](#)。改用 v3 协议后，请检查板地址、模式和“步/毫升”（«Шагов/мл»）校准：版本变更时，EEPROM 结构会重新初始化。

当前模式：

- Mixer：搅拌器，每分钟转数、运行时间、暂停、暂停后反转和软启动。
- Pump：以 ml/h 连续进料，可选暂停。
- Fill：按设定体积（ml）定量灌装。

外部传感器可以停止泵或使其暂停。如果 I2C 卡死，模块会自行停止步进电机并关闭继电器；恢复后需要用编码器确认。

警告。

Lua 函数 **set_stepper_target(sp, direction, target)** 只控制 I2CStepper 泵（工作中的偶数地址板）：速度、方向和步数。如果泵在总线上没有响应，函数返回 0；它不控制 Samovar 内置的步进电机。

--set_stepper_target(sp, direction, target) //sp - 速度，单位为步/秒，
direction - 正向或反向，
target - 步数
sp = 0 时步进电机停止。

示例

```
set_stepper_target(950, 0, 1000)
```

该示例设定速度 950、方向 0，移动 1000 步。

pH 酸度传感器及其连接

在奶酪制作模式下，pH 传感器让 Samovar 能在“等待 pH” («Ожидание pH») 阶段等到设定的酸度。测量套件由一支浸入式电极和一块 PH-4502C 板组成。电极接到该板的 BNC 接口；板的模拟输出 PO 把电压送给 Samovar。不要把电极本身直接接到 ESP32 或 ADS1115：它需要一块信号处理板。

电极必须适合被测产品及其温度。为水设计的传感器在牛奶中可能被蛋白质和脂肪污染；能否长时间浸泡，以电极的技术说明书为准。校准不能消除污染和老化。

通过 ADS1115 连接

ADS1115 是外置 ADC，即把电压转换为数字的转换器。连接需要一个 ADS1115 模块、两个精度 1% 的 10 kΩ 电阻和一个 100 nF 电容。这些数值针对的是 PH-4502C 板的输出，而不是电极电缆。

- PH-4502C 用 5 V 供电，ADS1115 用 3.3 V 供电。把两块板和 Samovar 的 GND 连在一起。连接前请核对您手中板子的引脚标记：不同模块的引脚顺序不同。
- PH-4502C 的 PO 经第一个 10 kΩ 电阻接到 ADS1115 的 AIN0。第二个 10 kΩ 电阻接在 AIN0 与 GND 之间。100 nF 电容与第二个电阻并联。分压器把电压减半：PO 上的 5 V 在 AIN0 上变为 2.5 V。DO 输出不用于 pH 测量。
- ADS1115 的 SDA 和 SCL 接到 Samovar I2C 的同名线上——I2C 是与各模块通信的公共两线制线路。使用您的 ESP32 开发板原理图中给出的引脚；GPIO 编号取决于具体构建。SDA/SCL 的上拉必须接到 3.3 V。
- ADDR 接 GND：ADS1115 的地址将为 0x48。如果该地址已被其他设备占用，请选一个空闲的 ADS1115 地址，并在构建设置中指定。同一条线上不能有两个地址相同的设备。
- 启用带地址值的构建设置 [USE ADS1115](#)，例如 [#define USE ADS1115 0x48](#)，然后上传固件。pH 会自动从 AIN0 读取，LUA_PIN 保持空闲。如果 ADS1115 不可用，将无法启动奶酪制作，也无法保存 pH 校准。

接到 ESP32 之前，用万用表检查 AIN0 与 GND 之间的电压。固件中 ADS1115 的测量范围为 ± 4.096 V，但不允许在输入端加高于芯片电源的电压。修改电路或改用 ADS1115 之后，请重新校准。

不用 ADS1115 的连接

如果禁用了 USE_ADS1115，同一个分压器的输出接到 LUA_PIN 而不是 AIN0。PH-4502C 用 5 V 供电，其 GND 与 Samovar 的 GND 相连。不能把 5 V 直接加到 ESP32 的输入端。LUA_PIN 也被 MPX5010DP 压力传感器使用：只能在上面接一个设备。奶酪制作期间，该输入被 pH 测量占用，无法从 Lua 读取。

开启设备时读数跳变怎么办

先比较在单独塑料杯缓冲液中和在奶槽中的读数，同时逐一开启泵、搅拌器和加热。如果跳变与设备开启有关，很可能是电气干扰。若设备关闭时缓冲液中的读数缓慢漂移，则需要检查电极和 PH-4502C；隔离无法修复它们的故障。

为消除经公共电源和地线传入的干扰，可以把测量部分隔离出来：**电极 → PH-4502C → 1:2 分压器 → ADS1115 → ADuM1250 → ESP32**。ADuM1250 是 I2C 电气隔离模块：它传递数据，但两侧之间没有电气连接。

- PH-4502C、ADS1115 以及 ADuM1250 的测量侧由单独的 USB 移动电源供电，移动电源须有 5 V 输出并支持小负载。在这个电路中 ADS1115 也用 5 V 供电；其输入端的分压器保留。测量期间，移动电源不得在充电，也不得为 Samovar 的其他部分供电。
 - ADuM1250 朝向 ESP32 的一侧由 Samovar 的 3.3 V 供电。ADS1115 的 SDA/SCL 接到 5 V 一侧，Samovar 的 SDA/SCL 接到 3.3 V 一侧。每一侧的上拉电阻都必须接到该侧自己的电源。
 - 不要把两侧的地线连在一起。**如果用导线、USB 电缆或经由其他设备把它们连起来，隔离就被破坏了。Samovar 的其他 I2C 设备留在 ESP32 一侧。
 - 把 PH-4502C 和 ADS1115 放在一起，远离加热调压器。PO 导线尽量短，电极电缆远离电源线。ADS1115 地址和 USE_ADS1115 设置保持不变。
- 这是一种抗电气干扰的连接方式，并不保证电极的精度。校准后，在开启设备的情况下再检查一遍；开启设备时电路不应产生可重复出现的 pH 偏移。

软件滤波与限制

在带中值滤波的固件中，pH 读数由间隔一秒的五次连续测量计算得出。中值是排序后位于中间的数：序列 5.42；5.41；3.10；5.43；5.42 得出 5.42。滤波可以去掉短暂的尖峰，但不能修正恒定误差或读数的缓慢漂移。读数平稳变化时，延迟约为两秒。

五个样本都填满后才能得到 pH。出现读取错误、无效读数、两次测量间隔超过五秒、校准系数变更或过程重置后，窗口会重新填充。旧结果不会被当作新的有效测量。在“等待 pH” («Ожидание pH») 阶段，到达阈值的确认保持 30 秒。

滤波对两种连接方式都适用——经 ADS1115 和经 LUA_PIN。校准页面上的 ADC 字段显示的是不经此滤波的最新原始样本；记录校准点前请等它稳定。Samovar 在 pH 换算中不考虑温度：请在相同温度下比较结果和对照样品。网站上的文档更新不会更新您设备上的固件。

校准与保养

用两种新鲜的缓冲液（约 pH 7 和 pH 4）校准整套组装好的套件。按溶液标签填入测量温度下的数值。用按钮操作的步骤见 [pH 校准](#) 一节。

用于牛奶之后，按制造商说明，用适合清除蛋白质和脂肪沉积的清洁剂清洗电极。传感端应存放在推荐的电极保存液中，不要干放，也不要泡在蒸馏水里。不要超过电极允许的温度。清洗后，在一份新鲜缓冲液中检查读数。

参考资料：[ADS1115](#)、[ADuM1250](#)、[电极保养](#)、[pH 测量中电气干扰的成因](#)。

附加功能

附加功能可以借助 Lua 以及通过 I2C 连接的 PCF8575/PCF8591 扩展器来实现。启用 Lua 的方法有：PlatformIO 中单独的 **Samovar_lua** 构建环境、配置器“设备”（«Оборудование»）选项卡上的“使用 Lua”（«Использовать Lua»）复选框，或 **user_config_override.h** 中的 **#define USE_LUA** 指令；扩展器也在同一处用 **USE_EXPANDER** 和 **USE_ANALOG_EXPANDER** 指令启用（在配置器中为“PCF8575 扩展器地址”（«Адрес расширителя PCF8575»）和“PCF8591 扩展器地址”（«Адрес расширителя PCF8591»）字段）。

附加功能的编程基于 [脚本语言 Lua](#)。脚本编辑器位于“设置 – 编辑器”页面（“设置”（«Настройки»）–“编辑器”（«Редактор»））。

端口扩展器上可以连接任意执行机构或按钮（以按钮方式工作的传感器），并由您自己决定在 Samovar 运行的某一时刻应当发生什么。

Lua 初始化时运行 **init.lua** 文件。Samovar 工作期间，通用的 **script.lua** 和设置中选定的模式脚本大约每秒运行一次。可以在网页界面的文件编辑器中修改它们。

可以在 Samovar 内存中保存变量、运行最多 10 个定时器、获取 Samovar 的运行状态等等。也就是说，可以连接附加设备，并在 Samovar 工作时自动控制它们

可以检查安全传感器是否触发，如果触发，直接从脚本关闭 Samovar，或者加大、停止供水。可能的方案很多，取决于具体的设备。

Samovar 的基本功能不会改变。

脚本可以使用开发板的一个模拟输入——**LUA_PIN**（ESP32 DevKit 为 GPIO34，ESP32-S3 为 GPIO4），通过 `analogRead()` 函数读取。**Samovar 7.00**：在 [奶酪制作](#) 模式下，该输入由 pH 传感器占用，从脚本访问它会报错。继电器、泵和按钮的输出不能从脚本切换——只能通过端口扩展器或 Samovar 的标准功能。

蓝牙串口监视器

Samovar 固件不使用 ESP32 模块自带的蓝牙。可以用外接的 HC-05 或 HC-06 模块搭建一个远程串口监视器：它们只是把 Samovar 输出到主诊断串口的内容（也就是在 Arduino IDE 或 PlatformIO 串口监视器中看到的内容）通过无线信道转发出去。串口速率为 **115200 波特**。

将蓝牙模块的 RX 输入接到主串口的 TX（开发板的 TX0 引脚），连通两边的 GND，并把模块设置为 115200 波特。供电电压按您所用模块板上的标注选择；ESP32 的信号线为 3.3 V。如果只想查看日志，模块的 TX 输出不必接到 Samovar 的 RX。带 RXD2/TXD2 引脚的 UART2（ESP32 DevKit 上为 GPIO16/GPIO17，ESP32-S3 上为 GPIO18/GPIO17）留给功率调节器使用——那是另一个串口。

Lua 脚本中 **print()** 函数的输出也发往同一个串口。项目源代码的 **Lua_UART** 文件夹中有一个独立 Arduino 接收端的示例：它通过 UART 接收一行字符串，按逗号拆成 8 段（每段最多 20 个字符），并显示在显示屏上。不要把模块直接接到市电路或调压器输出端。