

## Install

Disclaimer: You could damage your raspberry pi if you do not insert a voltage divider between the echo pin on the sensor and the GPIO pin on the Raspberry Pi. If you choose to do this you do it at your own risk.

Installation instructions assume Python3 on Raspberry Pi OS 11 (Bullseye) or Raspbian version 10 (Buster).

## Supported OS Versions

Raspberry Pi OS 11 (Bullseye)

Raspbian OS 10 (Buster)

Raspbian OS 9 (Stretch) - Support ended on June 30, 2022. Upgrade to Bullseye.

## Install Dependencies

**Note:** Your account must have sudo access to install Raspi-Sump.

Login to your Raspberry Pi running Raspberry Pi OS.

Check that your user account is a member of the gpio group. This is needed for accessing the gpio pins.

**groups**

You should see all groups your account belongs to. If gpio is not listed run the following command (where 'username' is your account name);

```
sudo usermod -aG gpio username
```

Logout and log back into your account for the groups to take effect.

Install Pip, RPi.GPIO and Matplotlib

```
sudo apt update && sudo apt -y upgrade
```

```
sudo apt install python3-pip python3-rpi.gpio python3-matplotlib
```

RPi.GPIO is the library that controls the sensor.

Matplotlib is used to generate charts.

The Pip package manager is required to install Raspi-Sump in the next step.

## Install Raspi-Sump

The following will automatically install hcsr04sensor if it is not already installed on your Pi.

```
sudo pip3 install --no-binary :all: raspisump
```

Navigate to `/home/username/raspi-sump/` and move the sample config file to this directory.

```
cd /home/username/raspi-sump
mv sample_config/raspisump.conf .
```

The `/home/username/raspi-sump` folder is setup as follows on install;

- `raspi-sump/sample_config/raspisump.conf` (all configurations for raspisump).
- `raspi-sump/csv` (location of waterlevel readings to csv file)
- `raspi-sump/charts` (location of charts if using `rsumpchart.py`)
- `raspi-sump/logs` (location of `rsumpmonitor.py` logs if using raspisump as a continuous process)
- `raspi-sump/web` (all files needed for the optional pi webserver install)
- `raspi-sump/cron` (example crontab for scheduling readings)

**\*\*Note** take care with your `raspisump.conf` file if you are using Gmail or any other mail system that requires authentication. Your username and password will be viewable in the file. You should have a strong password on your account.. The installer also tightens file security on the file automatically.

## Edit raspisump.conf

All configurations are recorded in `/home/username/raspi-sump/raspisump.conf`

See the configuration file for explanations of variables. You can choose to take imperial (inches) or metric (centimetres) water level readings.

## Hardware

Setup hardware (Please make sure you understand GPIO information on your pi).

You must use two resistors to create a voltage divider from the Sensor to the Pi. There are various combinations of resistors that you can use, a google search for Voltage Divider Calculator will allow you to calculate which combination you can use to bring the voltage down from the echo pin to 3.3V. I used a 470 Ohm and 1K Ohm resistor to bring the voltage down on the GPIO pin to 3.4 which is within a tolerable 5% level. I could have also used a 1K and 2K resistor to give me 3.33V.

Four wires connected as follows from the sensor to the pi (note, this will require some soldering). A floppy disk power connector fits nicely on the sensor. If you are just testing then a breadboard works great for quick and easy connections.

1-VCC pin to 5V pin on Pi (pin 2)

2-Ground pin to Ground on Pi (pin 6)

3-Trig pin to GPIO

4-Echo pin to GPIO (need 470R resistor and 1K resistor to create a voltage divider.) In short, the 470 Ohm and 1K Ohm resistor are connected to one another with the Echo wire soldered between both of them to the GPIO pin. The other end of the 1K resistor is then soldered to the Ground wire.

see <https://www.linuxnorth.org/raspi-sump/> for information on pins I used.

Google soldering resistors for good information on how to do this if you have never done it.

## Starting Raspi-Sump

To start raspi-sump manually and take your first waterlevel reading issue the command;

```
rsump.py
```

To run raspisump at 1 minute intervals enter the following line in crontab as follows;

1 - crontab -e

2 - enter line at the end of the crontab as follows;

```
*/1 * * * * /usr/local/bin/rsump.py &> /dev/null
```

3 - Save crontab

(See cron documentation for questions on configuring crontab)

1) To monitor the log file in the csv folder while raspi-sump is running;

e.g. `tail -f 'waterlevel-20230523.csv'`

## If running as a continuous process

There may be times where you want to run Raspi-Sump more than once every minute. The default setting is 0 which will run rsump.py for one reading and then exit. This allows you to use the linux Cron scheduler to run at a specific interval. Unfortunately cron allows one minute as its minimum interval.

To take readings at shorter intervals you can specify the amount of seconds between readings in the raspisump.conf file.

1) set `reading_interval` in `raspisump.conf` to desired interval in seconds (e.g. `reading_interval = 30`).

2) Add `rsumpmonitor.py` to crontab (see next section)

- 3) To start Raspi-Sump on bootup add the following line at the end of /etc/rc.local just before the line 'exit 0'
- ```
/usr/local/bin/rsump.py &
```
- 4) Reboot your Raspberry Pi or run the following command. Your pi will run Raspi-Sump on boot from now on.
- ```
rsump.py &
```

Note\*\*\* Do not forget the ampersand '&' as this will run the script as a background process.

- 6) To stop Raspi-Sump:
 

```
sudo killall 09 rsump.py
```
- 7) To monitor the log file in the csv folder while raspi-sump is running;
 

```
e.g. tail -f 'waterlevel-20230523.csv'
```

## Health check with rsumpmonitor.py. If checking level more than once per minute only.

To check for the health of the rsump.py process run the rsumpmonitor.py script as root. Add to pi user crontab as follows;

1 - crontab -e

2 - enter line at the end of the crontab as follows;

```
* /5 * * * * /usr/local/bin/rsumpmonitor.py &> /dev/null
```

3 - Save crontab

This will check the rsump.py process every 5 minutes and restart it if it is stopped.

## Making Line Charts of Sump Activity

You can make a daily chart of sump pump activity by using rsumpchart.py.

1 - From the command line run;

```
rsumpchart.py
```

This will create a line chart of sump pump activity. You can easily modify the file to save to a different location with another name. Combined with a scheduled cron job it is an easy way to see the latest activity graphically.

**\*\*Note** that this requires matplotlib on your RaspberryPi which can be installed with the apt command. See the Install Dependencies section at the beginning of this file.

You can also use the move\_file.sh script provided as an example of how you transfer files offsite to a webserver or save historical chart information.

## Test Email Alerts

### On Demand Email Test

To test that emails are working run the command 'emailtest';

```
emailtest
```

### Heartbeat Alerts

Raspi-Sump can send email tests at predefined intervals. See the raspisump.conf file option 'heartbeat' and 'heartbeat\_interval'.

In /home/username/raspi-sump/raspisump.conf, this section configures the email heartbeat once per week.

```
# Set a heartbeat sms or email interval in order to regularly test that your
# notifications are working as intended.
# 0 = No notifications
# 1 = Send notifications
heartbeat = 1

# Set the frequency of the sms/email heartbeat notifications.
# Values can be set to any number and are in minutes.
# For reference;
# daily    = 1439 minutes
# weekly   = 10079 minutes
# Monthly  = 43199 minutes
heartbeat_interval = 10079
```

## Optional - Setting Up a Local Web Server for easy Charts Viewing

Setting Up The Local Webserver on the Pi

### Purpose

The following instructions allow you to configure your raspberry pi to view graphs of sump pit activity through your web browser. This is accomplished by

configuring a local webserver on your pi.

Once complete you will be able to view sump pump activity by connecting to `http://ip_address_of_your_pi`

## Preparation

If you have not done so in a while run the following command to update your Pi. This command updates repository information and then upgrades packages that are installed on your Pi. If you did this already earlier in the instructions then it is not necessary to do again.

```
sudo apt update && sudo apt -y upgrade
```

## Getting Started

These instructions will do the following - install the Lighttpd webserver on your Pi - create your webcharts folder structure - configure cron to run the script to create for graphs of sump pump activity

Install the Lighttpd webserver on your Raspberry Pi as follows.

```
sudo apt install lighttpd
```

Create your first webcharts. This will create the needed folder under `/home/username/raspi-sump/charts/`

```
rsumpwebchart.py
```

Change to the web server root folder at `/var/www/html`

```
cd /var/www/html
```

Create the symlinks for your folders to be viewable with the web server. Replace 'username' with your account name.

```
sudo ln -s /home/username/raspi-sump/web/index.html index.html
```

```
sudo ln -s /home/username/raspi-sump/web/css css
```

```
sudo ln -s /home/username/raspi-sump/web/images images
```

```
sudo ln -s /home/username/raspi-sump/charts charts
```

Enable directory listing for historical charts

```
sudo lighttpd-enable-mod dir-listing
```

Restart the web server

```
sudo systemctl restart lighttpd
```

Create a cron job to generate an hourly graph of your sump pit activity for viewing on your pi webserver.

1 - `crontab -e`

2 - enter line at the end of the crontab as follows;

```
59 * * * * /usr/local/bin/rsumpwebchart.py &> /dev/null
```

3 - Save crontab

Open a web browser to `http://ip_of_your_pi`. At the 59th minute of every hour you will create a chart of sump pit activity for the day which will be viewable on this page. It will also copy historical information that you can access from the link in the web page.

## Support

For support open an issue on the Github Issue Tracker or consider joining our discord server.

For Discord simply send an email to `alaudet@linuxnorth.org` and request an invite link.