Geekbrains

Комплекс фиксации, хранения и визуализации параметров окружающей среды с доступом через интернет

Программа:

Инженер умных устройств

Группа: 5471

Ромашин Ростислав Геннадьевич

Москва

2024

# Введение

Человек является сложным биологическим видом, для его нормального функционирования необходимы определенные внешние условия, при несоблюдении которых будут происходить сбои или невозможность жизнедеятельности. Внешние факторы называются экологическими от греческого слова *экос*, т.е. «местопребывание», а наука, изучающая взаимодействие живых организмов со средой обитания называется «экология».

Основными экологическими метриками являются:

1. температура окружающей среды

2. относительная влажность

3. атмосферное давление

4. чистота воздуха

При нарушении хотя бы одной из вышеперечисленных метрик, велика вероятность гибели живого организма.

### 1. Температура окружающей среды

Гипотермия - это состояние, при котором температура тела снижается ниже нормы, что может привести к серьезным последствиям, таким как обморожение и смерть от переохлаждения. При гипотермии организм не способен поддерживать нормальную температуру, что приводит к замедлению обмена веществ и нарушению работы органов. Важно предпринимать меры для нагревания тела и предотвращения серьезных осложнений.

Гипертермия - это повышение температуры тела, что может привести к тепловому удару, ожогам и другим тепловым повреждениям. При гипертермии организм перегревается, что может вызвать нарушение работы органов, дезориентацию и даже смерть. Необходимо немедленно принимать меры для снижения температуры тела и предотвращения серьезных осложнений.

### 2. Относительная влажность

Высокий уровень влажности способствует развитию плесени, бактерий и вирусов, что увеличивает риск респираторных заболеваний и аллергических реакций. Постоянное воздействие высокой влажности может привести к ухудшению качества воздуха в помещении и ухудшению здоровья.

Низкий уровень влажности может вызвать обезвоживание организма, раздражение слизистых оболочек и проблемы с дыханием. Сухой воздух может способствовать появлению сухости кожи, раздражения глаз и дыхательных проблем, особенно у людей с респираторными заболеваниями.

### 3. Атмосферное давление

Изменения в атмосферном давлении могут вызвать баротравму, что проявляется в боли в ушах, давлении в груди и других серьезных проблемах. Баротравма может возникнуть при изменениях высоты, полетах на самолете или погружении под воду. Важно соблюдать правила безопасности и предотвращать возможные осложнения.

### 4. Чистота воздуха

Высокий уровень загрязнения воздуха токсичными веществами, такими как тяжелые металлы и химические соединения, может привести к респираторным заболеваниям, раку и другим серьезным заболеваниям. Загрязнение воздуха может вызвать проблемы с дыханием, аллергические реакции и ухудшение общего состояния здоровья. Необходимо принимать меры для снижения уровня загрязнения и защиты от вредных веществ в воздухе.

1. BME680 - Датчик газа, давления, температуры и влажности с низким энергопотреблением

- BSEC - проприетарное ПО

- BME68x-Sensor-API - открытое ПО

2. Odroid-X2 - Клиентский контроллер на базе CPU ARMv7

- ArchLinux

- Клиент mqtt на языке С

- Sense - приложение сбора метрик с датчика BME680 на языке С с помощью открытых API

- Web-сервер на языке С

- cronie - ctrontab для archlinux

3. IBM PC - Серверная часть на базе CPU x86

- Node-Red

- Mosquitto

- Telegraf

- InfluxDB

- Grafana

4. ЛВС - с возможностью организации интернет доступа ко всем вышеперечисленным компонентам

- OpenWRT router + ddns

- Nginx

## 1. BME680

Датчик газа, давления, температуры и влажности с низким энергопотреблением от компании BOSH.

BME680 — это цифровой 4-в-1 датчик, измеряющий загазованность, влажность, давление и температуру на основе проприетарных принципов измерения. Модуль датчика помещен в крайне компактный корпус с металлической крышкой LGA, имеющий размеры всего 3,0 × 3,0 мм² и максимальную высоту 1,00 мм (0,93 ± 0,07 мм). Его небольшие размеры и низкое энергопотребление позволяют интегрировать его в устройства, работающие от батареи или в устройства с частотной связью, такие как мобильные телефоны или носимые устройства.

Данный датчик 4-в-1 куплен на [алиэкспресс](https://aliexpress.ru/item/0\_4001042311625.html) за 353 рубля. Что, примерно в десять раз дороже [классических](https://aliexpress.ru/item/32228095913.html) "одномерных" датчиков. В чем же преимущества и недостатки?

### Программное обеспечение BSEC

Датчик BME680 предназначен для использования вместе с решением Bosch Software Environmental Cluster (BSEC) и API датчика BME6xy, чтобы раскрыть его полный потенциал. Программное обеспечение BSEC включает интеллектуальные алгоритмы, которые позволяют использовать такие сценарии, как мониторинг качества воздуха по различным индексам.

Датчик оснащен специальной пластиной, нагревающейся до 320 и более градусов по цельсию. Данная пластина замеряет сопротивление воздуха. В интернете можно встретить много отрицательных отзывов про точность данных датчиков. Но, просмотрев даташиты производителя понимаешь неоднозначность плохих выводов. С одной стороны допустимая погрешность измеряемой температуры +-1 градус по Цельсию не позволит применять его в больницах. Но, прибор не позиционируется как высокоточный и медицинский.

Некоторые этот датчик "по-старинке" подсоединяют к системам наподобие arduino или esp, и наблюдают большие разночтений с рядом стоящими бытовыми приборами. Например, температура воздуха вопреки погрешности может быть больше чем на 5 градусов от бытового термометра. Дело тут не в оригинальности продукта, хоть и существует большая доля вероятности купить брак. Дело именно в расчетах. Датчик по своей сути является микроконтроллером со своими регистрами и своей проприетарной логикой работы. Для расчета показателей мощностей ардуино не достаточно. Можно запустить датчик в форсированном режиме и он будет сбрасывать постоянно неточную информацию. Весь смысл заключается именно в закрытом проприетарном ПО. Именно оно участвует в сложнейших расчетах и управлении режимов датчика. Во первых, для того чтобы снять показания сопротивления воздуха, необходимо нагреть пластину до 320 градусов. Этот процесс кратковременный, но все же нагрев отражается на термодатчике. Следовательно нужно замеры температуры делать в промежутках между нагревами. При расчете температуры, проприетарным ПО учитываются остальные параметры, такие как влажность и давление. О сложности ПО говорит и тот факт, что драйвера для датчика BME680 и BME688 используются одни и те же. А вот датчик BME688 позиционируется уже как AI (ИИ).

Программное обеспечение BSEC от Bosch Sensortec доступно в виде закрытого двоичного файла, который будет предоставлен через Лицензионное Соглашение на Программное Обеспечение (SLA) на сайте Bosch Sensortec (https://www.bosch-sensortec.com/bst/products/all\_products/BSEC). Забегая вперед замечу, что пока мне с проприетарным ПО разобраться не удалось, из-за его сложности.

Ключевые особенности системы аппаратного и программного обеспечения:

Расчет температуры окружающего воздуха вне устройства (например, телефона)

Расчет относительной влажности окружающего воздуха вне устройства

Расчет индекса качества воздуха (IAQ) вне устройства

Кроме того, программные алгоритмы обрабатывают компенсацию влажности, базовую линию, а также коррекцию долгосрочного дрейфа сигнала газового датчика.

### BME68x-Sensor-API

API датчика охватывает базовую коммуникацию с датчиком и функции компенсации данных, и доступен как открытый исходный код на Github (https://github.com/BoschSensortec/BME68x-Sensor-API). Т.е. функционал открытого ПО ограничен. Бош бережно хранит алгоритм расчетов в секрете. Более того, они удалили свой старый открытый репозиторий, который содержал более "простой" для понимания драйвер. В новом репозитории, упомянутом выше, лежит другой драйвер, который требует наличия "тяжелого" COINES SDK.

[Cтарый](https://github.com/BoschSensortec/BME680\_driver/releases) репозиторий с открытым драйвером датчика BME680 до сих пор красуется в последнем даташите.

Но, нашел старый открытый драйвер 3.5.9 на стороннем репозитории https://github.com/wintersteiger/BME680\_driver.

Драйвер содержит открытие API. Для того чтобы начать получать или записывать информацию на датчик, необходимо самостоятельно написать основную программу для конкретной платформы и операционной системы, которая с помощью апи будет обращаться через драйвер к датчику.

Т.к. готовых примеров не нашел, то за основу взял решение для Raspberry Pi, но для более старого датчика BME280 https://www.waveshare.com/w/upload/b/ba/BME280-Environmental-Sensor-Demo-Code.7z. Код от BME280 частично совместим и позволяет отображать только температуру. "Подсмотрев" основную логику организации взаимодействия, удалось заставить шайтан-машину работать).

## Odroid-X2

Данный малино-подобный мини-ПК использовал в своем прошлом ДЗ для [веб-сервера и чатов](https://github.com/allseenn/ciot/tree/main/03.Tasks).

<img src="https://github.com/lucabelluccini/hardkernel-odroid-x-stuff/blob/master/201209072304519403.jpg?raw=true">

Распиновка в соответствии с картинкой выше (внизу правее на гребенке идет первый пин)

```

GND 50 49 ADC\_AIN3

VDD\_IO 48 47 ADC\_AIN2

SYS (5V) 46 45 VD16

VD13 44 43 VD22

VD4 42 41 VD1

VD23 40 39 VD9

VD17 38 37 VD14

VD10 36 35 VD5

VD12 34 33 VD3

GND 32 31 HSYNC

VD20 30 29 VDEN

VD6 28 27 VSYNC

VD11 26 25 VD18

VD7 24 23 VCLK

VD0 22 21 VD15

VD8 20 19 VD2

VD21 18 17 VD19

TXD 16 15 SPI\_1\_MISO

RXD 14 13 PWM\_BRT

VDDQ\_LCD 12 11 SPI\_1\_MOSI

SPI\_1\_CSN 10 09 XE\_INT12

T\_SDA 08 07 T\_SCL

SPI\_1\_CLK 06 05 SCL

T\_RST 04 03 SDA

BL\_EN 02 01 T\_INT

```

Схема подключения датчика BME680 к I2C Odroid-X2

```

BME680 Odroid-X2

+-----+-----------------+

GND GND (50 pin)

VDD VDD (48 pin) 1.8V

SDA SDA (03 pin)

SCL SCL (05 pin)

```

Датчик подключается без подтягивающих резисторов.

Основной проблемой [Odroid-X2](https://www.hardkernel.com/shop/odroid-x2/) является его "древность". Снят с поддержки лет 10 назад. Но, по мощности не уступает многим Raspberry даже сегодня. Официальной последней версией была Ubuntu 14.04. Умельцы сделали неофициальный образ на базе Ubuntu 16.04. Для языка СИ это не проблема, но хотелось, что-то посвежее.

### ArchLinux

И чисто случайно обнаружил [образ](https://archlinuxarm.org/platforms/armv7/samsung/odroid-x2) для odroid в официальном хранилище ArchLinux. Данная система позиционируется как более профессиональная версия Linux и одной из особенностей являются Rolling обновления. Т.е. как таковых версий у нее нету, переодически накатываются самые последние обновления и все. Таким образом получаем самое свежее ядро, библиотеки и ПО:

```

$ uname -a

Linux odroid 6.9.6-3-ARCH #1 SMP PREEMPT Thu Jun 27 07:33:29 MDT 2024 armv7l GNU/Linux

$ cat /etc/os-release

NAME="Arch Linux ARM"

PRETTY\_NAME="Arch Linux ARM"

ID=archarm

ID\_LIKE=arch

BUILD\_ID=rolling

ANSI\_COLOR="38;2;23;147;209"

HOME\_URL="https://archlinuxarm.org/"

DOCUMENTATION\_URL="https://archlinuxarm.org/wiki"

SUPPORT\_URL="https://archlinuxarm.org/forum"

BUG\_REPORT\_URL="https://github.com/archlinuxarm/PKGBUILDs/issues"

LOGO=archlinux-logo

```

### Клиент mqtt

Основной проблемой является, то что пакета с библиотекой paho-mqtt-c нет в офрепозиториях ArchLinux.

Пришлось собирать ее самостоятельно из исходников скачанных с [официального репозитория](https://github.com/eclipse/paho.mqtt.c).

```bash

make

make install

```

Написал программу для [mqtt-клиента](https://github.com/allseenn/ciot/blob/main/04.Tasks/mqtt/mqtt.c).

Програма позволяет с помощью ключей командной строки задать следующие параметры:

- -i - ip-адрес и порт mqtt-сервера

- -u - имя пользователя

- -p - пароль

Программа позволяет также принимать информацию со стандартного потока ввода, до 4-x аргументов в следующем порядке:

- температура

- давление

- влажность

- газ

### Sense

Используя полученный и раннее описанный, опыт инжиниринга драйвера от BME280, написал программу [sense](https://github.com/allseenn/ciot/tree/main/04.Tasks/sense) для получения информации с датчиков.

Учел, на будущее, особенности интеллектуальной работы датчика, и зашил в программу возможность получать информацию только об одном или нескольких конкретных показателей с помощью следующих ключей командной строки:

- -t - температура

- -p - давление

- -m - влажность

- -g - газ

- -h - справка

### Web-server

Веб сервер был основан на [раней версии](https://github.com/allseenn/ciot/blob/main/03.Tasks/03/server.c)

Добавлена авторизация с помощью логина и пароля.

<img src=img/01.jpg>

Веб-сервер использует консольную программу sense, описанную ранее и ее значения после парсинга заносятся в соответствующие массивы.

Далее информация в виде html-страницы отправляется авторизованному пользователю в виде фронтэнд-интерфейса.

<img src=img/02.jpg>

Интерфейс простой, он выводит в таблице названия измеряемых величин, их значения и размерности в единицах измерения СИ.

### Cronie

Cronie эта стандартная служба планировщик для системы archlinux. Благодаря тому что ранее написанные программы адаптированны для работы в консоли можно организовать пайплан по получению показаний датчика и отправки его на mqtt-сервер по расписанию: один раз в минуту:

```

$ crontab -l

# send temerature, pressure, humidity and gas to mqtt broker

\* \* \* \* \* /usr/local/sbin/sense -t -p -m -g | /usr/local/sbin/mqtt -i 192.168.1.8:1883 -u admin -p students

```

## IBM PC

В качестве сервера используется IBM PC совместимый компьютер с установленной на него Ubuntu 24.04.

На сервере установлен docker и docker-compose.

С помощью [bash-скрипта](https://github.com/allseenn/mesh/blob/main/03.Tasks/install.sh), созданного на прошлом курсе по распределенным сетям, генерируется docker-compose файл, устанавливающий следующие системы:

- Mosquitto

- Nod-Red

- Telegraf

- Influx-DB

- Grafana

### Mosquitto

Mosquitto является сервером mqtt, он не нуждается в особой настройке, кроме как задании логина и пароля. Я же его еще настроил для работы с веб-сокетами. Это нужно для организации работы через сеть интернет, посредствам веб-сервера nginx.

Роль mqtt-сервера принимать сообщения от паблешеров и отдавать их подписчикам. Поэтому большинство программ настраивается на mosquitto, а сам он не настраивается на них

### Nod-Red

<img src=img/01.png>

В качестве и подписчика и паблишера может выступать система node-red.

Добавим несколько подписчиков в его дашборд:

- temperature

- pressure

- humidity

- gas

<img src=img/02.png>

Для подключения к брокеру необходимо задать параметры такие как логин и пароль

<img src=img/03.png>

Таже необходима задать топик, т.е. тему с которой будет связан информация

<img src=img/04.png>

К каждому подписчику подключить отладчик. Таким образом мы сможем видеть что посылается на наш брокер mqtt по конкретной теме

<img src=img/05.png>

Для вывода информации необходимо выбрать маленького паучка справка на верху. Это часто забывается )

<img src=img/06.png>

Как только нажали на паука, поступающая информацию начнет поступать в окно. Помимо самих данных у нас отображается дата и время получения сообщения. По этой причине, отправка времени с программы sense (считывание датчика) является излишней. Т.к. далее в базе данных временных рядов, дата и время являются основополагающими для InfluxDB и нет необходимости задавать отдельное поле времени для этого.

<img src=img/07.png>

### Telegraf

Если node-red нам нужен только для отладки. И в дальнейшей схеме может быть исключен. То telegraf является связующим звеном между брокером и базой данных.

Он подписывается на все топики брокера используя заданные в настройках авторизационные данные и передает их дальше, в нашем примере InfluxDB

Пример конфига telegraf:

```

[agent]

interval = "3s"

round\_interval = true

metric\_batch\_size = 1000

metric\_buffer\_limit = 10000

collection\_jitter = "0s"

flush\_interval = "3s"

flush\_jitter = "0s"

precision = ""

hostname = ""

omit\_hostname = false

[[outputs.influxdb\_v2]]

urls = ["http://$LOC\_IP:8086"]

token = "$INFLUXDB\_TOKEN"

organization = "$ORG"

bucket = "$BUCKET"

[[inputs.mqtt\_consumer]]

servers = ["tcp://$LOC\_IP:1883"]

topics = ["#"]

username = "$USERNAME"

password = "$PASSWORD"

data\_format = "value"

data\_type = "float"

[[inputs.docker]]

endpoint = "unix:///var/run/docker.sock"

```

### InfluxDB

Ключевым игроком в нашей связке выступает СУБД Influx-DB, это не просто система управления базой данных, но и визуализация, которая может заменить в простых случаях Grafana.

Для начала в веб-интерфейсе InfluxDB необходимо выбрать бакет (корзину) в которую скидывает информация с помощью Telegraf.

В данном бакете можно выбрать те параметры и топики, которые мы хотим отслеживать

<img src=img/08.png>

СУБД поддерживает несколько языков запросов, один из которых SQL и еще один FLUX

<img src=img/09.png>

Пример запроса на языке FLUX

```

from(bucket: "IoT")

|> range(start: v.timeRangeStart, stop: v.timeRangeStop)

|> filter(fn: (r) => r["\_measurement"] == "mqtt\_consumer")

|> filter(fn: (r) => r["\_field"] == "value")

|> filter(fn: (r) => r["host"] == "e81d2a776ed2")

|> filter(fn: (r) => r["topic"] == "gas" or r["topic"] == "humidity" or r["topic"] == "pressure" or r["topic"] == "temperature")

|> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)

|> yield(name: "mean")

```

Данный запрос можно скопировать и использовать его в дальнейшем для построения визуализации в системе Grafana

### Grafana

Графана - это система визуализации и анализа данных. Она поддерживает огромное количество типов данных и хранилищ, как встроенными средствами, так и с помощью сторонних плагинов.

Необходимо с помощью авторизационных данных (token) подсоединиться к InfluxDB.

И в разделе Dashboards создать новый.

<img src=img/10.png>

Можно создавать множество дашбордов на каждый случай жизни, причем можно использовать всевозможные комбинации одних и тех же показателей с разными системами представления.

<img src=img/11.png>

При создании нового дашборда, отобразится список имеющихся соединений с хранилищами, в нашем случае это InfluxDB

<img src=img/12.png>

Ранее созданный в InfluxDB запрос можно использовать в дашборде

<img src=img/13.png>

Обязательно необходимо нажать на кнопку "Save" и сохранить дашборд, если нажать "Apply", то информация не сохранится.

<img src=img/14.png>

На дашборде выведена информация о четырех показателях. Спидометры которые показывают текущее значение и график за час. Временные рамки можно гибко настраивать.

<img src=img/15.png>

## Доступ в интернет

С помощью веб-сервера Nginx организован доступ ко всем компонентам системы. Без порт-маппинга и файрвола. Только средствами веб-сервера. Включая доступ к mosquitto по веб-сокету

Пример конфига nginx.conf

```

server {

listen 443 ssl;

server\_name mosquitto-rrg-5471.gb-iot.ru;

location / {

proxy\_http\_version 1.1;

proxy\_pass http://192.168.1.8:8081;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

proxy\_set\_header Host $host;

}

ssl\_certificate /etc/letsencrypt/live/gb-iot.ru/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/gb-iot.ru/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

}

server {

listen 443 ssl;

server\_name influxdb-rrg-5471.gb-iot.ru;

ssl\_certificate /etc/letsencrypt/live/gb-iot.ru/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/gb-iot.ru/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

location / {

proxy\_set\_header X-Forwarded-For $remote\_addr;

proxy\_pass http://192.168.1.8:8086;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

}

}

server {

listen 443 ssl;

server\_name grafana-rrg-5471.gb-iot.ru;

ssl\_certificate /etc/letsencrypt/live/gb-iot.ru/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/gb-iot.ru/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

location / {

proxy\_set\_header X-Forwarded-For $remote\_addr;

proxy\_pass http://192.168.1.8:3000;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

}

}

server {

listen 443 ssl;

server\_name nodered-rrg-5471.gb-iot.ru;

ssl\_certificate /etc/letsencrypt/live/gb-iot.ru/fullchain.pem;

ssl\_certificate\_key /etc/letsencrypt/live/gb-iot.ru/privkey.pem;

include /etc/letsencrypt/options-ssl-nginx.conf;

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem;

location / {

proxy\_pass http://192.168.1.8:1880;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

}

}

```

Прошивка OpenWRT с установленным пакетом DDNS позволяет постоянно обновлять доменные зоны. И легко находить доступ по имени.

## Список литературы:

1. [BME680 – Datasheet 1.9 02-2024](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme680-ds001.pdf) - 53 страниц

2. [BME680 – Datasheet 1.3 07-2019](https://github.com/allseenn/dipiot/blob/main/docs/CZ\_BME680\_BOSCH\_0001.pdf) - 54 страниц

3. [BSEC Integration Guideline](https://github.com/allseenn/dipiot/blob/main/docs/BST-BME680-Integration-Guide-AN008-47.pdf) - 56 страниц

4. [BSEC Binary Size Information](https://github.com/allseenn/dipiot/blob/main/docs/BSEC2%20Binary%20Size%20Information.pdf) -3 страницы

5. [BME68x - Shipment packaging details](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme68x-sp000.pdf) - 9 страниц

6. [BME680 – Application Note](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme680-an014.pdf) - 13 страниц

7. [BME680 Integrated Environmental Unit](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme680-fl000.pdf) - 2 страниц

8. [BME680: Handling, Soldering and Mounting Instructions](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme680-hs000.pdf) - 16 страниц

9. [BME680 Shuttle board 3.0 flyer](https://github.com/allseenn/dipiot/blob/main/docs/bst-bme680-sf000.pdf) - 2 страницы

10. [Release Notes 2.5.0.2](https://github.com/allseenn/dipiot/blob/main/docs/Release%20Notes\_BSEC\_2.5.0.2\_Website\_Release\_11012024.pdf)

\*Итого, около 200 страниц технической информации. Т.к. информации не для публикации, то ссылки на данные материалы ведут в закрытый репозиторий.