

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ
И ИНФОРМАТИКИ

Кафедра технологий программирования

ЛАБОРАТОРНАЯ РАБОТА 11
ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ МОБИЛЬНЫХ И
ВСТРАИВАЕМЫХ СИСТЕМ»

Проектирование, моделирование и реализация приложения для
встраиваемых систем

Методические указания по выполнению лабораторной работы

Минск 2024

СОДЕРЖАНИЕ

Введение	3
1 Методические рекомендации.....	4
1.1 Основной учебный материал	4
1.2 Введение в Raspberry Pi и Raspberry Pi ОС.....	4
1.2.1 Операционные системы для Raspberry Pi.....	6
1.2.2 Пакет симуляции платы Sense HAT.....	7
1.3 Node-RED	8
1.4 Дополнительные средства для проектирования встраиваемых систем.....	10
2 Методические указания и задания.....	11
2.1 Методические указания.....	11
2.1.1 Критерии оценивания	11
2.1.2 Отчет по лабораторной работе	11
2.1.2.1 Требования к репозиторию.....	12
2.1.2.2 Защита отчета по лабораторной работе	13
2.2 Задания.....	14
2.2.1 Задание 1.....	14
2.2.2 Задание 2.....	14
2.2.3 Задание 3.....	14
2.2.4 Контрольные вопросы.....	15

ВВЕДЕНИЕ

Целью работы является получение проектированию приложений для встраиваемых систем, моделированию сигналов датчиков и их обработка. Для достижения поставленной цели необходимо решить следующие задачи:

- развернуть виртуальную машину и установить операционную систему Raspberry Pi;
- установить в виртуальной машине программное средство Node-RED;
- установить пакет моделирования платы Sense HAT;
- создать учетную запись на сервисе Pusher.com для настройки системы обработки уведомлений;
- изучить примеры и реализовать их с целью моделирования системы обработки данных датчиков температуры в режиме реального времени.





1 Методические рекомендации

В данном разделе представлены рекомендации по выполнению лабораторной работы.








1.1 Основной учебный материал

Учебный материал для выполнения лабораторной работы изложен в источниках:

а) Полезные ресурсы по ОС Raspberry Pi и Node-RED:

- 1)  Raspberry Pi:Настройка/Введение в использование Node-RED вместе с Raspberry Pi;
- 2)  Знакомство с Node-RED и потоковое программирование в Yandex IoT Core
- 3)  Make a Simple Raspberry Pi Weather Dashboard Using IBM Cloud
- 4)  Node-RED IoT Workshop using Raspberry Pi and Sense HAT sensor

б) Учебные материалы по разработке приложений с моделированием обработки данных датчиков с применением Raspberry Pi и Node-RED:

-  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p.
-  Sense HAT Emulator
-  node-red-node-pi-sense-hat-simulator
-  Raspberry Pi:Операционная система
-  Raspberry Pi OS
-  3 Ways to Run Raspberry Pi Desktop on a Virtual Machine (Virtualbox, QEMU, VMware Workstation)
-  Raspberry Pi: Введение

1.2 Введение в Raspberry Pi и Raspberry Pi ОС


Raspberry Pi — одноплатный компьютер размером небольшого размера (см. рис. 1.1), изначально разработанный как бюджетная система для обучения информатике, но позже получивший более широкое применение и известность. Разрабатывается британской компанией  Raspberry Pi Foundation.



Рисунок 1.1 — Одноплатный компьютер Raspberry Pi

Несмотря на свои физические размеры, Raspberry Pi представляет собою полноценный одноплатный мини-компьютер, что значительно расширяет его сферы применения. В частности, может быть использован в качестве игровой приставки до умного дома, небольшого сервера или даже суперкомпьютера. На рисунке 1.2 представлены основные компоненты одноплатного компьютера.

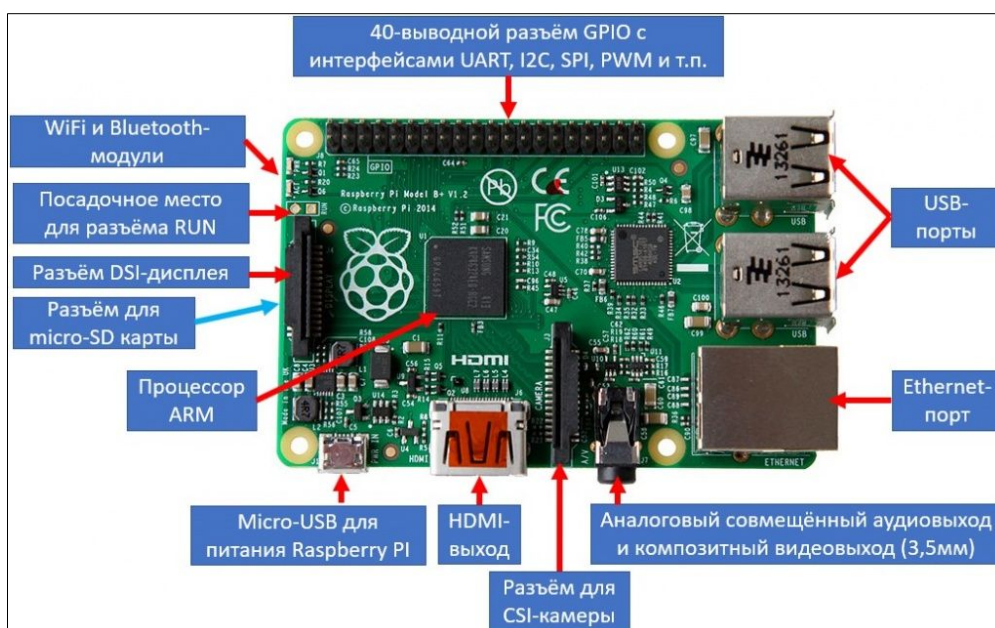


Рисунок 1.2 — Обзор составных элементов Raspberry Pi

Мини-компьютер Raspberry Pi выпускался или выпускается в следующих версиях: «А», «А+», «В», «В+», «2В», «Zero», «Zero W», «3В», «3В+», «3А+» и «4В». Первые четыре модели «А», «А+», «В» и «В+» оснащены ARM11-процессором Broadcom BCM2835 с тактовой частотой 700 МГц и модулем оперативной памяти на 256 МБ («А», «А+») и 512 МБ («В», «В+»). Модель «2В» оснащается процессором с 4 ядрами Cortex-A7 с частотой 900 МГц и оперативной памятью размером 1 ГБ. Модели Zero и Zero W выпустились тоже на базе ARM11, но уже с частотой 1 ГГц с объемом памяти в 512 МБ. Версии мини-компьютера «3В», «3В+» и «3А+» оснастили 4-ядерным процессором Cortex-A53 (ARM v8) с частотой процессора 1,2 ГГц у «3В» и 1,4 ГГц у «3В+» и «3А+» с оперативной памятью 1 ГБ у «3В» и «3В+» и 512 МБ у «3А+». На фоне предыдущих версии модель «4В» стала более продвинутой, получив новую СнК BCM2711 с 4-ядерным процессором ARM Cortex-A72 на частоте 1,5 ГГц. Raspberry Pi «4В» стала доступной в 4 вариантах с 1, 2, 4 или 8 ГБ ОЗУ на выбор пользователя (см. рис. 1.3).

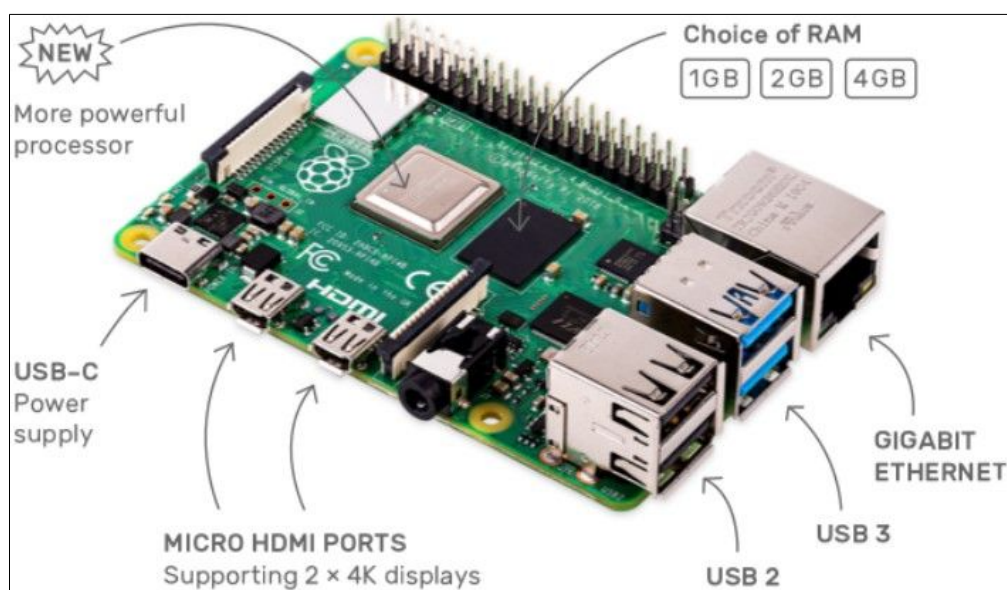




Рисунок 1.3 — Внешний вид Raspberry Pi 4B


1.2.1 Операционные системы для Raspberry Pi

Плата Raspberry Pi, как и обычный ПК, работает под управлением одной из специализированных операционных систем. Версия операционной системы может быть выбрана в зависимости от области применения, решаемой задачи или личных предпочтений.


Любая из рассматриваемых ниже версий ОС может быть установлена на одноплатный компьютер:

- *Raspberry Pi OS* (ранее *Raspbian*) была представлена в 2015 году как основная для Raspberry Pi. Она оптимизирована для процессоров с АРМ-архитектурой и достаточно активно продолжает развиваться. Основана на ОС Debian GNU/Linux. Включает графическую среду рабочего стола LXDE и менеджер окон Openbox (бесплатный менеджер для X Window System). В состав дистрибутива входят: программа компьютерной алгебры Mathematica; модифицированная версия Minecraft PI; урезанная версия Chrome. Способы установки в зависимости от основной системы и образы дистрибутива доступны по адресу  <https://www.raspberrypi.com/software/>.

- *Ubuntu Core* — версия Ubuntu для встраиваемых платформ ( <https://ubuntu.com/core>).

- *Fedora on Raspberry Pi* — версия ОС Fedora для встраиваемых систем. Подробнее об установке в документации  «Fedora on Raspberry Pi».


- Другие версии ОС Linux.

При отсутствии одноплатного компьютера и необходимости моделировать аппаратную платформу рекомендуется использовать средства виртуализации, например Virtualbox, QEMU, VMware Workstation. Варианты установки операционной системы Raspberry Pi рассматриваются в статье  3 Ways to Run Raspberry Pi Desktop on a Virtual Machine.

1.2.2 Пакет симуляции платы Sense HAT

Sense HAT — это дополнительная плата для Raspberry Pi, созданная специально для соревнований Astro Pi. Плата позволяет производить измерения температуры, влажности, давления и ориентации, а также выводить информацию с помощью встроенной светодиодной матрицы (см. рис. 1.4).

Реализован пакет эмуляции платы Sense HAT, который может быть использован для моделирования показаний датчиков. Может быть установлен как пакет **npm** или как пакет **deb**.

Детали установки пакета **node-red-node-pi-sense-hat-simulator** с помощью менеджера пакетов **npm** представлены в статье  A Node-RED node to simulate a Raspberry Pi Sense HAT.

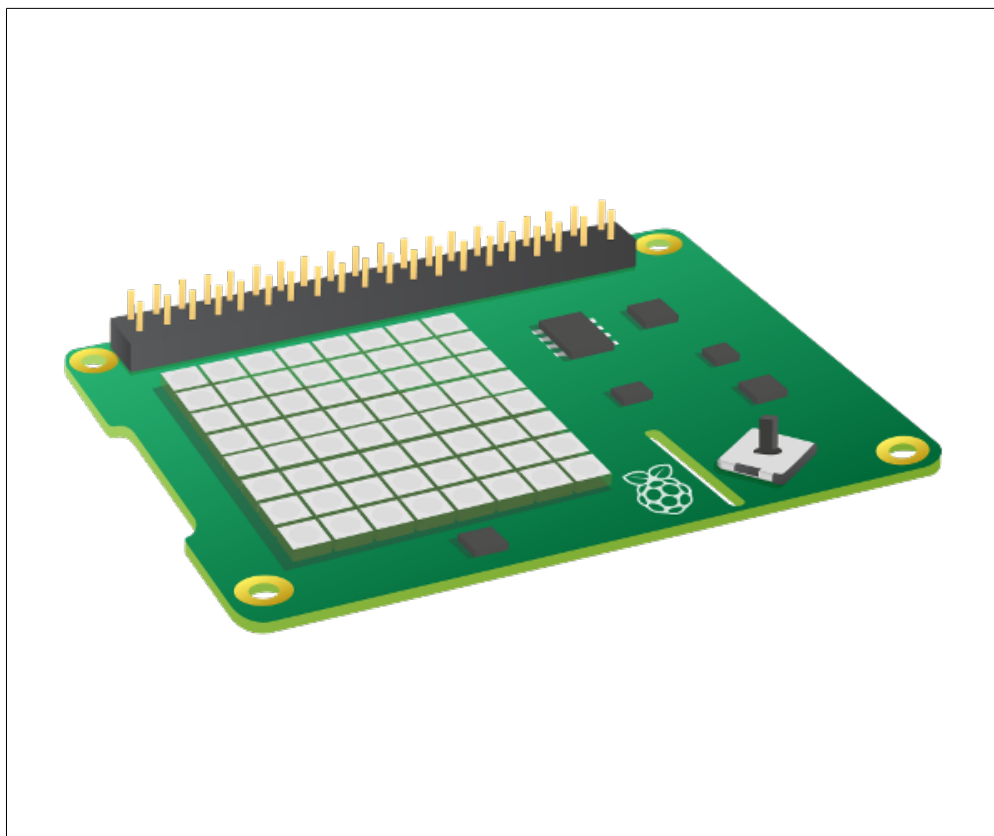



Рисунок 1.4 — Внешний вид платы Sense HAT

Проект  Sense HAT Emulator включает API для взаимодействия с пакетом симуляции, интерактивное GTK-приложение, позволяющее манипулировать эмулируемыми датчиками, утилиты командной строки для записи и воспроизведения показаний датчиков из реальной платы HAT. Документация включает классы и примеры реализации датчиков на языке Python.

1.3 Node-RED

Node-RED — это инструмент потокового программирования с открытыми исходными кодами, который упрощает программирование при создании проектов в области интернета вещей. Разработан командой IBM Emerging Technology Services и поддерживается OpenJS Foundation. Он использует технологию визуального программирования и предназначен для моделирования потока событий. Чтобы настроить проект на выполнение какой-либо задачи, пользователь подключает друг к другу блоки кода, которые называются *узлами* (nodes). Система из подключенных друг к другу узлов называется *поток* (см. рис.).

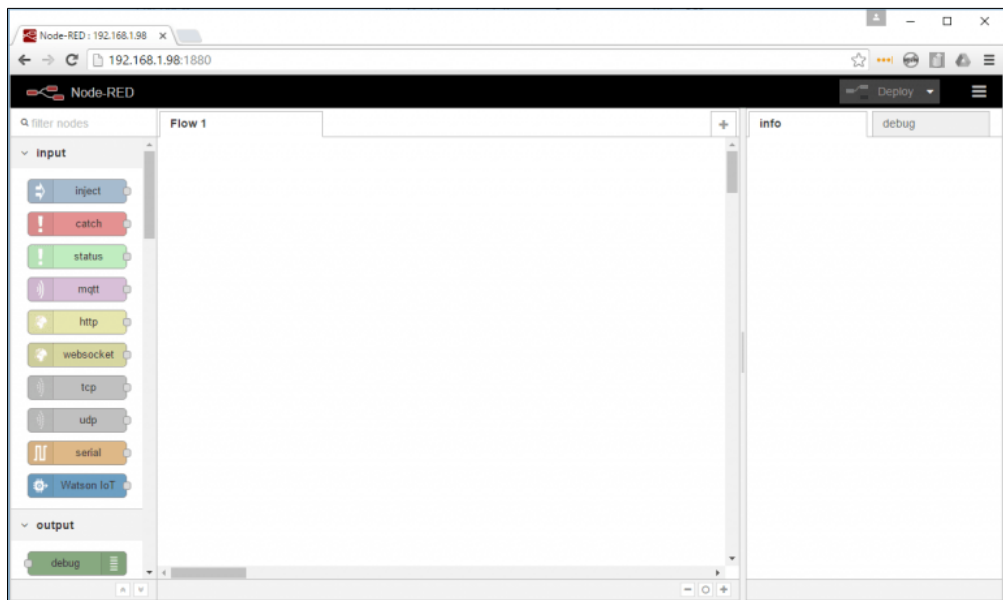





Рисунок 1.5

Программирование на основе потоков, изобретенное Дж. Полом Моррисоном в 1970-х годах, представляет собой способ описания поведения приложения как сети черных ящиков или «узлов» (nodes). Каждый узел имеет четко определенную цель; ему предоставляются некоторые данные, он выполняет операции над данными, а затем передает эти данные дальше. Сеть отвечает за поток данных между узлами.

Node-RED состоит из среды выполнения на основе Node.js, адрес которой указывается в веб-браузере для доступа к редактору потоков. Приложение создается в браузере за счет перетаскивания узлов из библиотеки (палитры) в рабочую область и их соединения. Запуск на выполнение обеспечивает развертывание приложения в среде выполнения.

Библиотека узлов расширяется за счет установки новых узлов, создания собственных узлов, представленных в виде файлов JSON.

Рекомендации по установке пакета *Node-RED* представлены в статьях:

-  Installing and Upgrading Node-RED.
-  Getting Started with Node-RED on Raspberry Pi.
-  Введение в использование Node-RED вместе с Raspberry Pi.

1.4 Дополнительные средства для проектирования встраиваемых систем


Онлайн-сервис *Pusher.com* предоставляет ресурсы для размещения двунаправленных гибких, масштабируемых и простых в использовании API. Данная инфраструктура обеспечивает обмен сообщениями и позволяет создавать функции реального времени.

InfluxDB — система управления базами данных с открытым исходным кодом для хранения временных рядов; изначально написана на языке Go, но позже переписана на Rust (язык программирования) и не требует внешних зависимостей. Основной фокус — хранение больших объёмов данных с метками времени (таких, как данные мониторинга, метрики приложений и показания датчиков), и их обработка в условиях высокой нагрузки на запись.

Grafana — это платформа с открытым исходным кодом для визуализации, мониторинга и анализа данных.

2 Методические указания и задания

2.1 Методические указания

Общие рекомендации по выполнению заданий лабораторной работы — изучить примеры из глав 1-5  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p. и реализовать систему моделирования показаний датчиков, их сбора, мониторинга и визуализации.

2.1.1 Критерии оценивания

Оценка 4-5

Выполнено задание 1. Представлен отчет с ответами на контрольные вопросы, исходный код проекта в `git`-репозитории.

Оценка 6-7

Выполнены задания 1-2. Представлен отчет с иллюстрацией хода выполнения примеров проектов и ответами на контрольные вопросы, исходный код проектов в `git`-репозитории.

Оценка 8-9


Выполнены задания 1-3 на отличном уровне. Представлен отчет с иллюстрацией хода выполнения примеров проектов и ответами на контрольные вопросы, исходный код проектов в `git`-репозитории.

2.1.2 Отчет по лабораторной работе

Отчет по лабораторной работе должен быть опубликован в репозитории и отвечать требованиям:

1. Отчет по лабораторной работе состоит из письменного отчета, кода приложений, опубликованных в репозиторий
2. Письменный отчет содержит цель работы.
3. Письменный отчет включает вариант задания.
4. Письменный отчет добавить описание ключевых моментов реализации и тестов.

5. Исходный код программ для каждого задания опубликовать в подкаталоге `/src` соответствующего каталога проекта (задания) и в соответствующей ветке репозитория.

Ссылка на репозиторий для лабораторной работы 10 доступна в курсе  «Программирование мобильных и встраиваемых систем».

В файле `README` в корневом каталоге проекта на *github* должна быть ссылка на отчёт и краткие сведения о выполненных заданиях (проектах).

Отчет опубликовать во внешнем хранилище или в репозитории в каталоге `/docs`. **! Отчёт должен сведения о ходе разработки и ответы на контрольные вопросы.**

2.1.2.1 Требования к репозиторию

Корневой каталог репозитория должен включать:

1. файл `README`, содержащий ссылку на отчет;
2. при публикации отчета в репозитории, разместить его в папке `/docs`;
3. каждое задание находится в каталоге проекта, структура которого соответствует требованиям задания;
4. в корневом каталоге репозитория и папках проектов должен быть добавлен файл `.gitignore`.

Пример оформления файла `README` может быть таким:

```
1      # Overview
2
3      Отчет по лабораторной работе.
4
5      # Usage
6
7      // Заменить <<link>> и <<folder>> на соответствующие ссылки и названия
8
9      To check, please, preview report by <<link>> and script files in
      <<folder>>.
10
11     # Author
12
13     Your name and group number.
14
15     # Additional Notes
16
17     // СКОПИРОВАТЬ И ВСТАВИТЬ ССЫЛКУ
18     // НА СВОЙ РЕПОЗИТОРИЙ, НАПРИМЕР
19
```

2.1.2.2 Защита отчета по лабораторной работе

Каждая лабораторная работа содержит тексты задач и контрольные вопросы, ответы на которые проверяются преподавателем при приёме работы у студента.

Выполнение студентом лабораторной работы и сдача её результатов преподавателю происходит следующим образом:

1. Студент выполняет разработку программ.

В ходе разработки студент обязан следовать указаниям к данной задаче (в случае их наличия). Исходные тексты программ следует разрабатывать в соответствии с требованиями к оформлению для программ на языке Dart.

2. Студент выполняет самостоятельную проверку исходного текста каждой разработанной программы и правильности её работы, а также свои знания по теме лабораторной работы.

Исходные тексты программ должны соответствовать требованиям к оформлению, приведённым в приложении. Недопустимо отсутствие в тексте программы следующих важных элементов оформления: спецификации программного файла и подпрограмм, а также отступов, показывающих структурную вложенность языковых конструкций.

Для проверки правильности работы программы студенту необходимо разработать набор тестов и на их основе провести тестирование программы. Тестовый набор должен включать примеры входных и выходных данных, приведённые в тексте задачи, а также тесты, разработанные студентом самостоятельно.

Самостоятельная проверка знаний по теме лабораторной работы выполняется с помощью контрольных вопросов и заданий, приведённых в конце текста лабораторной работы.



3. Студент защищает разработанные программы. Защита заключается в том, что студент должен ответить на вопросы преподавателя, касающиеся разработанной программы, и контрольные вопросы.

К защите необходимо представить исходные тексты программ, оформленных в соответствии с требованиями, исходные коды модульных тестов (unit-тестов), при наличии.

2.2 Задания



2.2.1 Задание 1

Цель — установить ОС Raspberry Pi в виртуальном окружении, пакет Node-RED и настроить тестовый проект.

1. Изучить рекомендации по установке Raspberry Pi и Node-RED из методических рекомендаций лабораторной работы.
2. Изучить материалы глав 1-2 из  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p.
3. Изучить пример приложения, представленный в главе 2  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p.


2.2.2 Задание 2

Цель — установить ОС Raspberry Pi в виртуальном окружении, пакет Node-RED и настроить тестовый проект с моделированием показаний датчиков Sense HAT:

1. Изучить рекомендации по установке Raspberry Pi и Node-RED из методических рекомендаций лабораторной работы.
2. Изучить материалы глав 1-4 из  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p.
3. Изучить примеры приложений, представленные в главах 2-4  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p. и реализовать их.

2.2.3 Задание 3

Цель — установить ОС Raspberry Pi в виртуальном окружении, пакет Node-RED, настроить тестовый проект с моделированием показаний датчиков Sense HAT, сбором данных и их визуализацией в среде Grafana:

1. Изучить рекомендации по установке Raspberry Pi и Node-RED из методических рекомендаций лабораторной работы.
2. Изучить материалы глав 1-5 из  Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p.

3. Изучить примеры приложений, представленные в главах 2-5
🔗 Bernardo Ronquillo Japón. Learn IoT Programming Using Node-RED. — BPB Publication, 2022. — 251 p. и реализовать их.

2.2.4 Контрольные вопросы

1. Что такое IoT? Какой термин используется для IoT на русском языке?
2. Что такое встраиваемая платформа? Что такое встраиваемая система?
3. Перечислите возможности одноплатного компьютера Raspberry Pi.
4. Охарактеризуйте плату Arduino. В чем отличие от Raspberry Pi?
5. Приведите примеры одноплатных компьютеров, отличные от линейки Raspberry Pi.
6. Опишите возможности операционную систему Raspberry Pi.
7. Для чего предназначен сервис Node-RED?
8. Какие возможности предоставляет пакет Sense HAT?
9. Охарактеризуйте InfluxDB и приведите примеры других баз данных, которые используются в проектах для журналирования данных.
10. Особенности и возможности средства визуализации Grafana.