

Міністерство освіти і науки України
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторних робіт
з дисципліни **“Проектування та моделювання
програмного забезпечення вбудованих
систем”**
для студентів магістратури
121 «Інженерія програмного забезпечення»
освітньої програми «Програмне забезпечення систем»
всіх форм навчання

2017

Методичні вказівки до лабораторних робіт з дисципліни «Проектування та моделювання програмного забезпечення вбудованих систем» для студентів магістратури спеціальності 121 «Інженерія програмного забезпечення» освітньої програми «Програмне забезпечення систем» всіх форм навчання / Г. В. Табунщик, Н. О. Миронова. – Запоріжжя: ЗНТУ, 2017. – 21 с.

Автори: Г. В. Табунщик, к.т.н., професор
Н. О. Миронова, к.т.н., доцент

Рецензент: А.В. Притула, к.т.н., професор

Відповідальний
за випуск: Г.В. Табунщик, к.т.н., професор

Затверджено
на засіданні кафедри
програмних засобів

Протокол №11
від 06 червня 2017

ЗМІСТ

Вступ.....	4
Лабораторна робота №1 Інсталяція Raspbian GNU/Linux Та настройка SSH-підключення на Raspberry Pi.....	5
Лабораторна робота №2 Інсталяція та налаштування веб-серверу на Raspberry Pi	9
Лабораторна робота №3 Робота з платою розширення та датчиками для Raspberry Pi.....	12
Лабораторна робота №4 Робота з платою розширення та датчиками для Raspberry Pi за допомогою Node-RED.....	18

ВСТУП

Метою даного курсу є навчити студентів використовувати навички об'єктно-орієнтованого аналізу та проектування для проектування інформаційних систем з використанням сучасних CASE-засобів.

На основі вивчення дисципліни «Проектування та моделювання програмного забезпечення вбудованих систем» студент повинен знати етапи життєвого циклу програмного забезпечення, елементи об'єктної моделі, процедури об'єктно-орієнтованого аналізу, процедури об'єктно-орієнтованого проектування, основи уніфікованої мови моделювання UML, основні концепції уніфікованого процесу розробки програмного забезпечення UP, архітектуру програмного забезпечення, CASE-засоби формування вимог до програмного забезпечення та проектування програмного забезпечення.

Для виконання лабораторних робіт рекомендується використовувати середовища розробки Qt Creator, PyCharm та мови програмування C++, Python, для побудови діаграм UML рекомендовано використовувати вільне програмне забезпечення.

ЛАБОРАТОРНА РОБОТА №1

ІНСТАЛЯЦІЯ RASPBIAN GNU/LINUX ТА

НАСТРОЙКА SSH-ПІДКЛЮЧЕННЯ НА

RASPBERRY PI

Мета роботи

Навчитись встановлювати операційну систему Raspbian GNU/Linux та виконувати настройку SSH-підключення на Raspberry Pi.

Теоретичні відомості

Технічні характеристики. Raspberry Pi – одноплатний комп'ютер розміром з банківську карту, спочатку розроблений як бюджетна система для навчання інформатики, в подальшому отримав набагато ширше застосування та популярність. Перші версії були створені в 2011 році. Raspberry Pi побудований на основі процесору архітектури ARM.

Схема пристрою наведена на рис. 1.1.

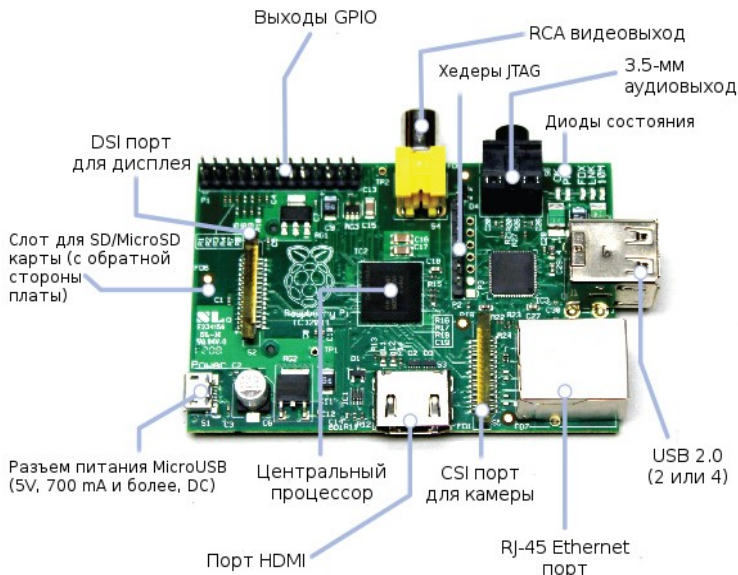


Рисунок 1.1 – Пристрій Raspberry Pi

Операційні системи, що підтримуються. Raspberry Pi працює в основному на операційних системах (ОС), заснованих на Linux ядрі, таких як:

- Raspbian (модифікація Debian);
- Pidora (модифікація Fedora);
- Arch Linux ARM;
- Kali Linux.

Так само можливо використання FreeBSD та Windows 10 IoT (Інтернет речей) Core.

Будемо використовувати операційну систему Raspbian. Це модифікація найпоширенішої операційної системи Debian GNU / Linux. У Raspbian є можливість використовувати легкий графічний інтерфейс (LXDE), а також стандартний менеджер пакетів для встановлення програм через Інтернет.

В якості ПЗУ Raspberry Pi використовує, залежно від моделі, SD або Micro SD карти пам'яті обсягом більше 8 Гб.

Raspberry Pi може працювати зі звичайними периферійними пристроями, такими як USB-клавіатура, миша, до неї можна підключити дисплей за допомогою HDMI.

Для включення комп'ютера достатньо подати живлення за допомогою microUSB роз'єму.

Встановлення Raspbian GNU/Linux. Для встановлення та налаштування Raspbian ми скористаємося так званим «headless» підходом – налагодження операційної системи, без підключення периферії до Raspberry Pi.

Для цього спочатку потрібно завантажити образ операційної системи Raspbian з офіційного сайту (<https://www.raspberrypi.org/downloads/raspbian/>), а потім записати його на microSD-карту (від 4 Гб).

Для запису образу на Windows можна скористатися утилітою Win32DiskImager з графічним інтерфейсом.

На Linux можна скористатися командою dd:

```
dd if=<file> of=<device>
```

де file – шлях до файлу образу системи;

device – шлях до карти пам'яті(наприклад, /dev/sdb1).

Карта пам'яті повинна попередньо бути відформатована як fat32.

Налаштування мережевих підключень. Далі нам необхідно налаштувати з'єднання з мережею. У разі, якщо в мережі немає DHCP сервера з автоматичною роздачею адрес, потрібно налаштувати мережу вручну. Для цього відкриваємо розділ з системою на карті пам'яті (другий, не boot) та редагуємо файл `etc / network / interfaces`. Приклад конфігурації:

```
auto eth0
allow-hotplug eth0
iface eth0 inet static
address 10.0.84.52
netmask 255.255.255.0
network 10.0.84.0
broadcast 10.0.84.255
gateway 10.0.84.200
```

Після цього ми можемо вставити карту пам'яті в слот Raspberry Pi, включити його в локальну мережу та підключити живлення. Дочекавшись, коли він завантажиться, можна спробувати підключитися до нього по ssh з комп'ютера, що знаходиться в тій же мережі.

На Windows для цього можна використовувати утиліту putty, на Linux – команду ssh:

```
ssh pi@<ip>
```

де ip – адрес, вписаний раніше в конфігурацію Raspbian.

За замовчуванням, пароль користувача – raspberry. Далі запускаємо команду:

```
sudo raspi-config
```

щоб закінчити налаштування системи. У цьому меню можна поміняти пароль користувача, перерозбити розділи карти пам'яті та інше.

Виявлення периферійних пристроїв. Для виявлення периферійних пристроїв в Linux можна використовувати наступні команди:

- `lspci` – для відображення всіх пристроїв, підключених на шину PCI;
- `lsusb` – для відображення всіх пристроїв, підключених до USB роз'ємів (якщо роз'єм не зайнятий – інформація про нього все одно виводиться).

Завдання на лабораторну роботу

1.1 Провести встановлення та базове налаштування Raspbian GNU / Linux на Raspberry Pi в режимі «headless» (без використання периферійних пристроїв).

1.2 Налаштувати підключення до мережі Raspberry для можливості доступу за допомогою SSH з локальної мережі.

1.3 Написати програму для визначення периферійних пристроїв для Raspberry Pi. Скласти діаграму компонентів.

1.4 Скласти звіт по проведеній роботі.

1.5 Відповісти на контрольні запитання.

Контрольні запитання

1. Що таке Raspberry Pi та чому її варто використовувати?
2. Які можливості надає Raspberry Pi?
3. Наведіть технічні характеристики Raspberry Pi.
4. Наведіть схему пристрою Raspberry Pi.
5. Перелічіть ОС, що підтримуються Raspberry Pi.
6. Які існують способи встановлення ОС на Raspberry Pi?
7. Як виконується встановлення Raspbian GNU/Linux без підключення периферійних пристроїв?
8. Що потрібно для роботи з Raspberry Pi?
9. Яким чином виконується налагодження мережі у Raspberry Pi?

ЛАБОРАТОРНА РОБОТА №2 ІНСТАЛЯЦІЯ ТА НАЛАШТУВАННЯ ВЕБ- СЕРВЕРУ НА RASPBERRY PI

Мета роботи

Навчитись виконувати установку та налаштування веб-серверу на Raspberry Pi.

Теоретичні відомості

Застосування LAMP на Raspberry Pi. Так як Raspberry Pi представляє з себе повноцінний комп'ютер з підтримкою роботи в мережі, ми можемо розгорнути на ньому класичний веб-сервер. В даному випадку ми скористаємося зв'язкою, яку називають LAMP (Linux, Apache, MySQL, PHP). Raspberry Pi можна використовувати як тестовий сервер для розробника, або як невеликий веб-сервер для власних потреб.

Також, тому як Raspberry Pi часто використовують для збору даних з різних датчиків, за допомогою веб-сервера можна організувати зручний їх перегляд.

Ми спробуємо розгорнути звичайний веб-сервер та віддати по мережі прості html-сторінки.

Встановлення веб-серверу на ОС Raspbian GNU / Linux.

Для того, щоб почати установку необхідного програмного забезпечення, нам знадобиться підключення до мережі. Для установки використовується пакетний менеджер apt-get. Перед установкою також рекомендується оновити репозиторій системи командою apt-get update.

Приступимо до встановлення програмного забезпечення:

Для встановлення сервера apache використовуємо команду:

```
apt-get install apache2
```

Для встановлення сервера баз даних MySQL:

```
sudo apt-get install mysql-server
```

Для встановлення пакету PHP та зв'язки його з MySQL:

```
sudo apt-get install php5
sudo apt-get install php5-mysql
```

Після цих операцій веб-сервер буде встановлений та запущений.

Для того, щоб перевірити його працездатність, ми можемо відкрити браузер та ввести в адресний рядок: localhost або 127.0.0.1 (адреса локального сервера за замовчуванням).

За замовчуванням, файли, які віддає Apache, розташовуються в каталозі / var / www

Створення необхідних сторінок. Для того, щоб відредагувати головну сторінку нашого маленького сайту, потрібно запустити текстовий редактор, наприклад (ми запускаємо редактор від імені суперкористувача, так як за замовчуванням власник каталогу / var / www – користувач root):

```
sudo mousepad /var/www/index.html
```

У цей файл нам потрібно вписати базову HTML-розмітку:

```
<html>
<head>
    <title>Raspberry Pi web server</title>
</head>
<body>
    <table>
        <tr>
            <td>Температура:</td>
            <td></td> <!-- тут буде значення
температури →
        </tr>
        <tr>
            <td>Освітленість:</td>
            <td></td> <!-- тут буде значення
освітленості →
        </tr>
    </table>
```

```
</body>  
</html>
```

Тепер, якщо ми оновимо сторінку localhost в браузері, ми побачимо нашу сторінку.

Завдання на лабораторну роботу

2.1 Виконати встановлення та базове налаштування пакетів Apache, PHP5, MySQL в середовищі Raspbian GNU / Linux.

2.2 Створити HTML-документ, який буде відображати створений веб-сервер та переконатися в його доступності по мережі.

2.3 Побудувати діаграму розміщення компонентів (діаграму розгортання).

2.4 Оформити звіт по виконаній роботі.

2.5 Відповісти на контрольні запитання.

Контрольні запитання

1. Поняття LAMP.

2. Яким чином виконується встановлення веб-серверу на ОС Raspbian GNU / Linux?

ЛАБОРАТОРНА РОБОТА №3 РОБОТА З ПЛАТОЮ РОЗШИРЕННЯ ТА ДАТЧИКАМИ ДЛЯ RASPBERRY PI

Мета роботи

Навчитися працювати з платою розширення та датчиками для Raspberry Pi.

Теоретичні відомості

Часто Raspberry Pi використовують для роботи з низькорівневим обладнанням, зокрема, для отримання і аналізування даних, що надходять з різних датчиків. Для цього використовується шина загального введення-виведення GPIO (General Purpose Input Output). Для роботи з шиною розроблено безліч різних бібліотек.

Надалі, ці дані можна отримувати, підключаючись до Raspberry віддалено, або використовувати в інших програмах, що працюють на пристрої.

У даній лабораторній роботі ми скористаємося спеціально спроектованою навчальною платою розширення, яка оснащена, зокрема, датчиками освітленості і температури. Також на ній присутні 8 світлодіодів і CAN-шина для підключення різних пристроїв. Схема плати розширення представлена на рис. 3.1.

Для роботи із платою за допомогою Python потрібно встановити бібліотеку **GPIO**, залежно від версії Python:

```
sudo apt-get update  
sudo apt-get install python-rpi.gpio  
sudo apt-get install python3-rpi.gpio
```

Основні команди модуля **RPi.GPIO**.

Імпортування модуля **RPi.GPIO**:

```
import RPi.GPIO as GPIO
```

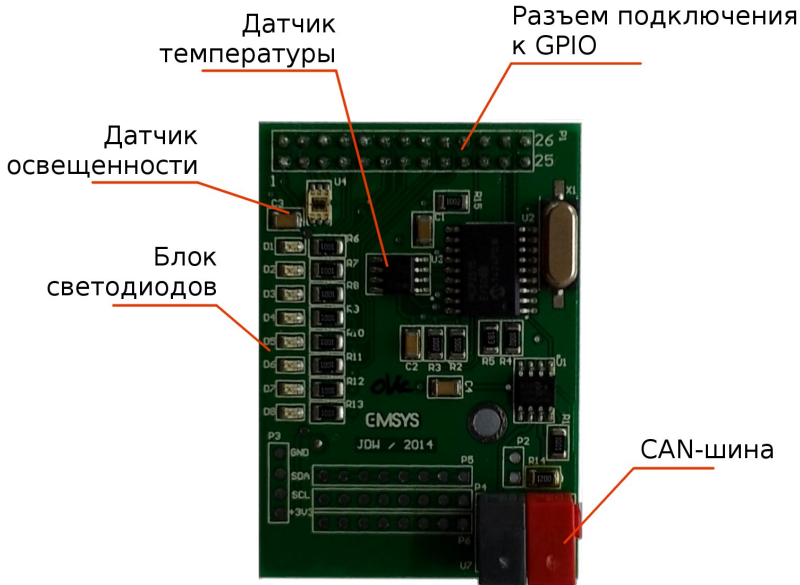


Рисунок 3.1 – Схема платы расширения для Raspberry Pi

Нумерація контактів:

```
GPIO.setmode(GPIO.BOARD)
# or
GPIO.setmode(GPIO.BCM)
```

Щоб визначити, яка система нумерації контактів була встановлена (наприклад, іншим модулем Python):

```
mode = GPIO.getmode()
```

Необхідно налаштувати кожен канал, який використовується, як вхідний або вихідний. Щоб налаштувати канал як вхідний:

```
GPIO.setup(channel, GPIO.IN)
```

Щоб налаштувати канал як вихідний:

```
GPIO.setup(channel, GPIO.OUT)
```

Для зчитування та встановлення стану контакту GPIO відповідно (state може бути 0 / GPIO.LOW / False або 1 / GPIO.HIGH / True):

```
GPIO.input(channel)
GPIO.output(channel, state)
```

Для очищення наприкінці скрипту, існує наступна команда:

```
GPIO.cleanup()
```

Нижче наведений приклад дозволяє виконувати маніпулювання світлодіодами:

```
import RPi.GPIO as GPIO
import time

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
arr = [7,8,18,16,15,13,12,11]

while True:
    for i in arr:
        GPIO.setup(i, GPIO.OUT)
        GPIO.output(i, True)
        time.sleep(0.1)
    for i in arr:
        GPIO.setup(i, GPIO.OUT)
        GPIO.output(i, False)
        time.sleep(0.1)
```

Для маніпулювання світловими датчиками використаємо **SMBus** (System Management Bus) – протокол обміну даними, заснований на інтерфейсі I2C.

Встановимо **smbus-cffi** – Python модуль, що надає доступ до SMBus через інтерфейс I2C / dev на Linux:

```
sudo apt-get install python-pip
sudo apt-get install build-essential libi2c-dev
i2c-tools python-dev libffi-dev
```

```
pip install smbus-cffi
```

Основні команди модуля.

- `long write_quick(int addr)` – надіслати адресний байт;
- `long read_byte(int addr)` – прочитати один байт з пристрою, не вказуючи регістр пристрою;
- `long write_byte(int addr, char val)` – надіслати пристрою один байт;
- `long read_byte_data(int addr, char cmd)` – Read Byte Data транзакція читання байта;
- `long write_byte_data(int addr, char cmd, char val)` – Write Byte Data транзакція запису байта;
- `long read_word_data(int addr, char cmd)` – Read Word Data транзакція читання слова;
- `long write_word_data(int addr, char cmd, int val)` – Write Word Data транзакція запису слова;
- `long process_call(int addr, char cmd, int val)` – Process Call транзакція визову процесу;
- `long[] read_block_data(int addr, char cmd)` – Read Block Data транзакція читання блоку даних;
- `write_block_data(int addr, char cmd, long vals[])` – запис до 32 байтів; ця функція додає початковий байт, який вказує довжину масиву `vals` перед самим масивом – можна натомість використовувати `write_i2c_block_data`;
- `long[] block_process_call(int addr, char cmd, long vals[])` – Block Process Call транзакція, що поєднує у собі запис блоку з наступним читанням блоку за тією ж адресою пристрою.

Для перегляду приєднаних пристроїв можна виконати наступну команду:

```
sudo ls /dev/i2c*
```

Результатом буде список, наприклад, `/dev/i2c-0 /dev/i2c-2 /dev/i2c-4 /dev/i2c-6`.

Для того, щоб отримати доступ до пристрою у програмі, треба використати команду `smbus.SMBus()` та передати у якості аргумента

останню цифру, що позначає девайс у вище наведеному списку, наприклад:

```
bus = smbus.SMBus(6)
```

Приклад програмної реалізації:

```
import smbus
import time
```

```
bus = smbus.SMBus(1)
address = 0x29
```

```
bus.write_byte(address, 0xa0)
bus.write_byte(address, 0x03)
time.sleep(0.5)
```

```
while True:
    bus.write_byte(address, 0xac)
    a = bus.read_byte(address)
    bus.write_byte(address, 0xad)
    b = bus.read_byte(address)
    print a + b*256
    time.sleep(0.1)
```

Порядок виконання лабораторної роботи

3.1 Застосувавши конспект лекцій та додаткову літературу, вивчити принципи програмування Raspberry Pi.

3.2 Використовуючи код прикладів, реалізувати програму, яка буде через встановлені проміжки часу отримувати значення з датчика освітленості, і показувати рівень освітленості за допомогою світлодіодів.

3.3 Використовуючи код прикладів, реалізувати програму, яка буде при запуску отримувати значення з датчиків освітленості і температури і поміщати отримані числа в заданий аргументом текстовий файл.

3.4 Оформити звіт. Звіт з лабораторної роботи повинен містити: мету роботи; завдання на виконання роботи; тексти програм; результати роботи; висновки.

3.5 Відповісти на контрольні запитання.

Контрольні запитання

1. Що таке Raspberry Pi та чому її варто використовувати?
2. Які можливості надає Raspberry Pi?
3. Наведіть технічні характеристики Raspberry Pi.
4. Перелічіть ОС, що підтримуються Raspberry Pi.
5. Що потрібно для роботи з Raspberry Pi?
6. Що таке GPIO та яким чином можна нею керувати?
7. Дайте визначення та характеристику модулю SMBus.

ЛАБОРАТОРНА РОБОТА №4 РОБОТА З ПЛАТОЮ РОЗШИРЕННЯ ТА ДАТЧИКАМИ ДЛЯ RASPBERRY PI ЗА ДОПОМОГОЮ NODE-RED

Мета роботи

Навчитися працювати з платою розширення та датчиками для Raspberry Pi, використовуючи Node-RED.

Теоретичні відомості

Node-RED – це потужний open-source інструмент, який спрощує програмування при створенні проектів в області інтернету речей. Він використовує технологію візуального програмування: щоб налаштувати проект на виконання будь-якої задачі, користувач підключає один до одного блоки коду, які називаються «вузлами». Система з підключених один до одного вузлів називається «поток». Система з підключених один до одного вузлів називається «поток».

Можливості Node-RED:

- налаштування з'єднання з іншими платами (Arduino, ESP8266 та інше);
- створення для свого проекту інтерактивного графічного інтерфейсу;
- взаємодія зі сторонніми сервісами;
- зчитування даних з мережі (прогноз погоди, курси акцій, електронні листи та інше);
- запуск подій, призначених на певний час;
- зберігання і зчитування даних з баз даних.

Node-RED надає редактор потоків на основі браузера, який спрощує об'єднання потоків, використовуючи широкий діапазон вузлів в палітрі. Потім потоки можуть бути розгорнуті під час виконання одним клацанням миші.

Функції JavaScript можуть бути створені за допомогою текстового редактора. Вбудована бібліотека дозволяє зберігати корисні функції, шаблони або потоки для повторного використання.

На рис. 1.1 зображено головне вікно редактора потоків:

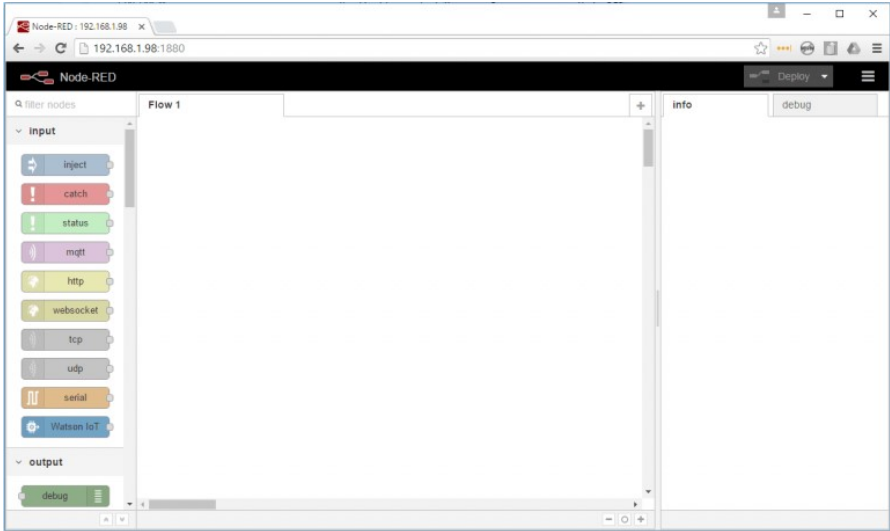


Рисунок 1.1 – Редактор потоків

У даній лабораторній роботі необхідно використати наступні типи вузлів:

Inject – вставляє в повідомлення часову мітку або налаштований користувачем текст. Може бути налаштований для введення вручну, із заданим інтервалом або в певний час.

Debug – надає простий спосіб перегляду повідомлень, що відображаються на панелі налагодження. Можна налаштувати відображення тільки повідомлення `msg.payload` або всього об'єкта `msg`.

Function – універсальний програмований функціональний вузол. Використовуючи стандартний JavaScript, вузол може бути адаптований для виконання складної обробки на своїх вхідних повідомленнях, що генерують одне або кілька вихідних повідомлень.

Delay – загальний вузол, який затримує повідомлення на певний або випадковий час. Також можна налаштувати обмеження потоку повідомлень (наприклад, 10 повідомлень за секунду).

Rpi_gpio out – вихідний вузол Raspberry Pi. Очікує `msg.payload` з 0 або 1 (або true або false). Обраний фізичний контакт буде переведено у стан HIGH або LOW в залежності від переданого значення.

Початкове значення контакту також може бути встановлено у 0 або 1. При використанні режиму PWM очікується вхідне значення числа від 0 до 100.

Sense HAT – цей вузол відправляє свідчення з різних датчиків на Sense HAT, згруповані в три набори; події руху, події середовища і події джойстика. Події руху включають в себе свідчення з акселерометра, гіроскопа і магнітометра, а також поточний курс компаса. Вони відправляються зі швидкістю приблизно 10 разів на секунду. Події середовища включають показники датчиків температури, вологості і тиску. Вони відправляються зі швидкістю приблизно 1 раз на секунду. Події джойстика відправляються при взаємодії з джойстиком Sense HAT.

Завдання на лабораторну роботу

4.1 Застосувавши конспект лекцій та додаткову літературу, вивчити принципи роботи з Node-RED.

4.2 Використовуючи приклади, реалізувати програму, яка буде послідовно вмикати та вимикати світлодіоди на платі розширення.

4.3 Використовуючи приклади, реалізувати програму, яка буде при запуску отримувати значення з датчика температури та виводити дані на екран.

4.4 Оформити звіт. Звіт з лабораторної роботи повинен містити: мету роботи; завдання на виконання роботи; тексти програм; результати роботи; висновки.

4.5 Відповісти на контрольні запитання.

Контрольні запитання

1. Які існують можливості інструментарію Node-RED?
2. Які існують типи вузлів в Node-RED?

ЛИТЕРАТУРА

1. Петин В.А. Микрокомпьютеры Raspberry Pi. Практическое руководство. – СПб.: БХВ-Петербург, 2015. – 240 с.
2. Мацяшек Л.А. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML / Л.А. Мацяшек // М.:Изд. дом «Вильямс», 2002. – 432 с.
3. Описание Rational Software Architect for WebSphere Software версии 7.5: [Электрон. ресурс]. – Режим доступа: http://www.ibm.com/developerworks/ru/library/r-0926_arnold/index.html.
4. Что нового в IBM Rational Software Architect 8.0: [Электрон. ресурс]. – Режим доступа: <http://www.ibm.com/developerworks/ru/library/r-whats-new-in-rational-software-architect-8/>.
5. Введение в IBM Rational Software Architect: [Электрон. ресурс]. – Режим доступа: <http://www.ibm.com/developerworks/ru/library/kunal/>.
6. Кватрани, Ю. Визуальное моделирование с помощью Rational Rose 2002 и UML: пер. с англ. [Текст] / Кватрани Ю. – М. : Вильямс. 2002. – 240 с.
7. Кватрани, Т. Визуальное моделирование с помощью IBM Rational Software Architect и UML: пер. с англ. [Текст] / Т. Кватрани. Дж. Палистрант. – М.: КУДИЦ-ПРЕСС. – 2007. – 192 с.
8. Ларман, К. Применение UML и шаблонов проектирования. 2-е издание [Текст] / Ларман К. – М. : Вильямс, 2002. – 624 с.
9. Соммервилл, И. Инженерия программного обеспечения: пер. с англ. [Текст] / Соммервилл И. – М.: Вильямс. 2002. – 624 с.
10. Шаблоны [Patterns] проектирования: [Электронный ресурс]. – Режим доступа: <http://oad.asf.ru/Patterns.aspx>.
11. Шаллоуей А. Шаблоны проектирования. Новый подход к объектно-ориентированному анализу и проектированию. Пер. с англ. –М.: Изд. дом «Вильямс», 2002. – 288 с.