МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

ТАТЧУК ОКСАНА МИКОЛАЇВНА

Допускається до захисту:
Завідувач кафедри
інформаційних технологій,
д.т.н., доцент, Нескородєва Т. В.
«__»_____20__р.

ІДЕНТИФІКАЦІЯ ТА КОНТРОЛЬ ДАВАЧІВ ІНТЕРНЕТУ РЕЧЕЙ ЗА ДОПОМОГОЮ СТАТИСТИЧНОГО АНАЛІЗУ ПОТОКУ ДАНИХ

Спеціальність 105 Прикладна фізика та наноматеріали
Кваліфікаційна (бакалаврська) робота

Науковий керівник:
Крижановський В.Г,
професор кафедри
інформаційних технологій,
д.т.н., професор

_____
(підпис)

Оцінка : _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____
(підпис)

Вінниця – 2022

АНОТАЦІЯ

Татчук О. М. Ідентифікація та контроль давачів Інтернету речей за допомогою статистичного аналізу потоку даних. Спеціальність 105 «Прикладна фізика та наноматеріали», спеціалізація «Технології Інтернету речей». Донецький національний університет імені Василя Стуса, Вінниця, 2022.

У кваліфікаційній (бакалаврській) роботі досліджено процес вимірювання температури за допомогою давачів Інтернету речей. Показано збір, обробку та аналіз дослідної системи. Встановлено статистичні відмінності між давачами.

Ключові слова: випадковий процес, вимірювання, дисперсія Аллана, нейромережа, статистичний аналіз.

49 с., 14 табл.,, 27 рис., 1 дод., 21 джерел.

ANNOTATION

Tatchuk O. M. Identification and control of Internet of Things sensors using statistical analysis of data flow. Specialty 105 "Applied Physics and Nanomaterials", specialization "Internet of Things Technologies". Vasyl Stus Donetsk National University, Vinnytsia, 2022.

In the qualification (bachelor's) work the process of temperature measurement with the help of Internet of Things sensors is investigated. Collection, processing and analysis of the research system are shown. Statistical differences between sensors are established.

Key words: random process, measurements, Allan variance, neural network, statistical analysis.

49 pp., 14 tables, 27 figures, 1 appendix, 21 sources.

ЗМІСТ

ВСТУП

В даний час Інтернет речей (IoT) є популярною парадигмою, яка передбачає світ, в якому всі види фізичних об’єктів пов’язані з Інтернетом, маючи можливість взаємодіяти один з одним і співпрацювати для досягнення поставлених цілей.

Розумні системи, що виникли природним шляхом або створені іншими розумними системами, успішно функціонують у реальному світі лише тоді, коли вони отримують якісні і об'єктивні дані. Для того, щоб системи IoT працювали безперервно та надійно, сигнали, що надходять від давачів мусять нести достовірну інформацію, що служить основою для розумної поведінки, для формування уявлень про модель цієї дійсності.[14]

Інтернет речей набуває все більших обсягів, тому й кількість сенсорів та приладів збільшуються, при цьому є великі шанси отримати недостовірні дані, за рахунок нестравності або виходу із ладу систем.

Системи IoT потребують універсальних методів, здатних швидко реагувати на відхилення від звичного режиму роботи, забезпечити мінімальне енергоспоживання та мати можливість обробляти великі потоки даних.

Метою цієї роботи є оцінка виявлення порушення звичайного режиму роботи сенсорної мережі Інтернету речей внаслідок зовнішнього втручання або аварії за допомогою аналізу зміни статистичних параметрів даних від сенсорів.

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ АНАЛІЗУ ВИМІРЮВАЛЬНОЇ СИСТЕМИ IoT

Аналізувати систему, процес або сигнал потрібно опираючись на так звані детерміністичний та статистичний підходи. Основну частину останнього складають методи математичної статистики, теорії ймовірностей та випадкових процесів.[2] Статистичний підхід до розв'язування задачі передавання інформації в системах IoT є об'єктивною реальністю і дає змогу не тільки краще зрозуміти проблему, а і використовувати отримані результати для розробки вдосконаленої версії систем.

1.1 Теорія вимірювання у застосуванні до Інтернету речей

Постійне дослідження системи та її вивчення можливе на основі емпіричного та теоретичного підходів[5] а також їх поєднання. Особливості даних підходів зазначені у таблиці 1.1

Таблиця 1.1 – Особливості способів дослідження системи

Підхід	Особливості
Емпіричний	Зібрання відомостей про систему; первинне узагальнення даних; опис даних; класифікація; систематизація.
Теоретичний	Обробка фізичних даних; формулювання проблем, гіпотез та теорій

Слід зазначити, що обидва підходи є рівноцінними і мають можливість доповнювати один одного. Одночасно, виходячи з ряду суттєвих чинників, доцільно застосовувати теоретичний підхід, основний принцип якого базується на математичному моделюванні[5] й передбачає вивчення об'єкта на основі його математичної моделі. Головними чинниками є: відображення тільки тих властивостей системи, які є найбільш важливими в конкретному випадку, ігноруючи велику кількість другорядних; неможливість проведення

експерименту з певних причин (економічних, технічних тощо); передбачення чи прогнозування результатів у нестабільних умовах вимірювання. [2]

При описі вимірювання користуються термінами такими, як наслідок і результат[1]. Наслідок – це те, що отримується в процесі проведення комплексу дій за певних умов, при чому присутність експериментатора не є обов'язковою. Результат вимірювання є сукупністю наслідків, що з'явилися в ході експерименту

У теорії вимірювання оперують таким поняттям, як «Помилки»[16]. Вони є неминучою похибкою, що залежить від ходу експерименту.

Результат вимірювань залежать від правильної постановки задачі[5] та об'єкту досліджень. Одержаний результат несе за собою характеристики точності, що є основної задачею вимірювання.

Для кожного вимірювання важливим є визначення похибки, адже без даної інформації не можна визначити достовірність й точність вимірювальних величин [4].

Якщо розглянути застосування теорії вимірювань на прикладі давачів IoT, потрібно враховувати характеристики кожного сенсора. Крім того, враховуються особливості системи в цілому та зовнішні фактори. Це допоможе визначити достовірність власних вимірювань.

1.2 Вимірювання, як випадковий процес

Передавання інформації на відстань, як сукупності знаків чи символів, що утворюють деяку форму, їхнє збереження можливе тільки за допомогою деякого матеріального носія чи фізичного процесу [2]

Розглядаючи вимірювання температури, як випадковий процес $X(t)$ зазначимо, що він являє собою функцію, яка відрізняється тим, що значення в довільні моменти часу по координаті t є випадковими[7]. З теоретичної позиції, випадковий процес $X(t)$ слід розглядати як сукупність тимчасових функцій $x\,k(t)$, які містять загальну статистичну закономірність[4]. При реєстрації

випадкового процесу на певному часовому інтервалі здійснюється фіксування одиничної реалізації $x_k(t)$ з незліченної кількості можливих реалізацій процесу $X\ (t)$. Ця одинична реалізація називається вибірковою функцією[2] випадкового процесу $X\ (t)$. Можливість виконати оцінки статистичних характеристик має окрема вибіркова функція, при цьому не характеризуючи процес в цілому.

З практичної точки зору вибіркова функція є результатом окремого вимірювання[6], після якого її реалізацію $x_k(t)$ можна вважати детермінованою функцією. Статистичною характеристикою процесу є N-мірна щільність ймовірностей $p(x_n t_n)$.[6] Зазвичай в ході експерименту обмежуються одно-та двовимірною щільністю ймовірностей процесів.

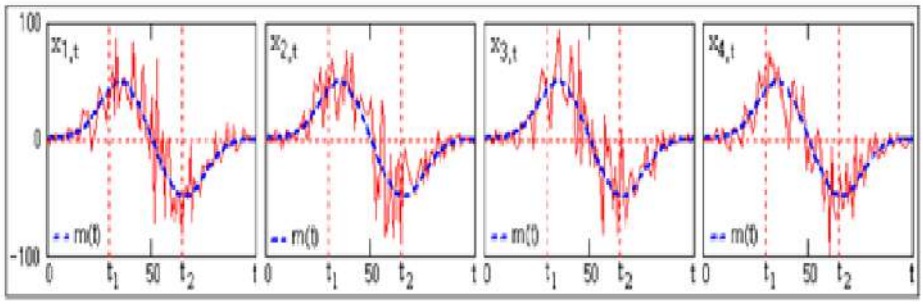

Рисунок 1.1 – Вибіркові функції випадкового процесу

Прикладом випадкового процесу можна розглянути графіки зі довільними значеннями[7], а саме для випадкового процесу $X\ (t)$ задано ансамблі реалізацій $\{x_1(t), x_2(t), \dots x_k(t), \dots\}$. У довільний момент часу t_1 зафіксовано значення всіх реалізацій $\{x_1(t_1), x_2(t_1), \dots x_k(t_1), \dots\}$. Сукупність цих значень являє собою випадкову величину $X\ (t_1)$ і є одновимірним перетином випадкового процесу $X\ (t)$. Приклади перетинів випадкового процесу $X\ (t)$ по 100 вибірках $x_k(t)$ в точках t_1 і t_2 наведено на рис.1.1 [7].

Щоб вирішити задачі різного роду, маємо справу із аналізом випадкових величин, при якому немає потреби в знаходженні повній ймовірнісній характеристиці випадкових величин.[1] Завдання одномірної щільності ймовірності $p(x_y, t_x)$ дозволяє зробити статистичне усереднення[6] і самої

величини x_y будь-якої функції f(x). Статистичне усереднення (ensemble averaging) у даному випадку є усередненням по ансамблю реалізацій в будь який визначений часовий момент.

Середнє значення функції:

$$\overline{\varphi(x)} = \int_{-\infty}^{\infty} \varphi(x)p(x,t)dx \quad (1.1)$$

де $\varphi(x)$ – середнє значення випадкового випробування;

$p(x,t)$ – статистичне усереднення[7]

Формула (1.1) показує усереднення по результатам випадкових випробувань.

Шляхом усереднення даних процесів по ансамблю реалізацій, моменти розподілу можуть оцінити випадкові процеси.

Момент 2-го порядку $X(t)$ – середнє значення 2-го ступеня змінної:

$$m_n(t) = \overline{x^n(t)} = \int_{-\infty}^{\infty} x^n(x)p(x,t)dt \quad (1.2)$$

де $x^n(t)$ – випадкова величина n-го порядку.

$p(x,t)$ – статистичне усереднення[7]

Найбільш використовувана характеристика закону розподілу випадкового процесу – математичне очікування.[2]

Співвідношення основних числових характеристик випадкових процесів зображені на формулі 1.3[7].

$$m\{f(x)\} = \int_{\infty}^{\infty} f(x)p_x(x,t)dt \quad (1.3)$$

Оскільки прийнято $f(x) =$ Ху , то математичне очікування є середнім значенням випадкового процесу моменту часу t.

Математичне сподівання – це оцінка середнього значення.[2]

$$m_x(t) = \int_{-\infty}^{\infty} xp(x,t)dt \quad (1.4)$$

Другий момент випадкового процесу:

$$\overline{x^2(t)} = \int_{-\infty}^{\infty} x^2p(x,t)dt \quad (1.5)$$

Дисперсія обраховується математичним очікуванням квадрата відхилення випадкового процесу [7]:

$$D_x(t) = \sigma_x^2(t) = \int_{-\infty}^{\infty}(x(t) - m_x(t))^2\, p(x,t)dx \quad (1.6)$$

При нульовому математичному сподіванні дисперсія характеризує середню потужність випадкового відхилення процесу[6]. Дисперсія випадкової величини є додатною. Якщо випадкова величина – константа, то $Dx(t) = 0$. Дисперсія $Dx(t)$ випадкової величини позначається через квадрат середнього квадратичного відхилення $a_2 = Dx$. Відхилення показує розкид значень вимірюваної величини.

Серед двовимірних функцій розподілу особливе місце займає функція кореляції[5], яка показує статистичний зв'язок випадкового процесу у різні часові проміжки. Вона дозволяє користуватися перетворення Фур'є і об’єднуватись з спектральної щільністю потужності:

$$R_x(t_1, t_2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\big(x(t_1) - m_x(t_1)\big)\big(x(t_2) - m_x(t_2)\big)p(x_1, x_2; t_1, t_2)dx_1dx_2 \quad (1.7)$$

Коли $t_1 = t_2$ при поєднанні перетинів випадкового процесу, функція кореляції дорівнює дисперсії, як показано у формулі (1.8).

$$R_x(t,t) = R_x(0) = \sigma_x^2(0) \quad (1.8)$$

Чим швидше функція $R(t)$, тим менший статистичний зв'язок між миттєвими значеннями випадкового сигналу у різні часові проміжки[2].

Виберемо закон розподілу Пуассона для сигналів, що надходять від давачі. Такий вибір обумовлений часовим та діапазонним обмеженням.[7], якщо вона використовує множині значення (m=0,1,2,...) з імовірностями:

$$P(X = m) = \frac{a^m}{m!}e^{-a}, (a > 0) \quad (1.9)$$

де m – кількість значень;

a – математичне сподівання [7].

Кількість подій, що настають в один і той же час, незалежно один від одного та зі сталою інтенсивністю, описуються даним розподілом.[16]. Математичне сподівання і дисперсія однакові і дорівнюють a. Також для даного розподілу є таблиці різних значень a (0,1-20)[6]. У них для певних значень a наведено $P(x \geq m), P(X = m)$ ймовірності .

Закон розподілу Пуассона показують ламаною лінією, що з'єднує точки з координатами *(n, P_n)*. На рис. 1.2 наведений даний закон для випадкової величини *X*, з параметром *X* = 0,5; 1, 2; 3,5; 5 [6].

Для подальшого аналізу інформації, що надходить з давачів обираємо тип «випадковий телеграфний сигнал».

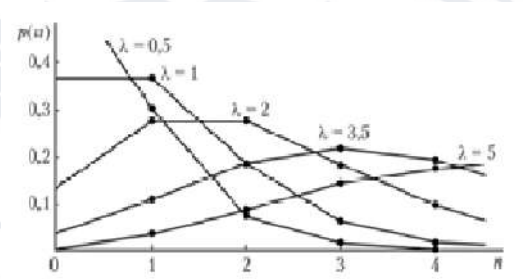

Рисунок 1.2 – Графічне зображення розподілу Пуассона

Для детального розгляду процесу проведення вимірювань[4], позначимо величину, в нашому випадку температуру, як *x*. [16] За найвигіднішу оцінку вважаємо середнє значення.

$$x_{\text{найкр}} = \bar{x} = \frac{x_1 + x_2 + \cdots + x_N}{N} = \frac{\sum x_i}{N} \tag{1.12}$$

де $x_1 \ldots x_N$ – вимірювальні значення;

N – кількість вимірювальних значень[16]

Стандартне відхилення[5] результатів вимірювання $x_1 \ldots x_N$ – це оцінка середньої похибки даних.

Якщо прийняти, що середнє $\bar{x}$ – це найкраща оцінка величини x, то ми розглядаємо вираз $x_i - \bar{x} = d_i$. Ця різниця має назву «залишок» x_i від $\bar{x}$ і показує різницю результату виміру від середнього значення.

Розрахунок достовірності проводиться таким чином, що потрібно піднести до квадрату всі відхилення, які утворять набір позитивних чисел, а після цього усереднити їх.

$$\sigma_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(d_i)^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{1.13}$$

Середнє стандартне відхилення[16] можна описати, як середньоквадратичне відхилення результатів вимірювання $x_1 \dots x_{N^2}$.

Щоб порахувати σ_x потрібно визначити d_i, піднести значення в квадрат.

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum(x_i - \bar{x})^2} \quad (1.14)$$

Отримуємо два вирази для прорахунку σ_x. Вираз (1.14) є більш точним та покращеним і при ньому маємо більше значення на відміну від (1.13). [16]

Продуктивність обладнання сильно залежить від якості сенсора[4] Давачі, мають властивість захоплювати шуми і не можуть забезпечити чистий результат.

1.3 Застосування дисперсії Аллана

Відхилення Аллана, $\sigma_\gamma(\tau)$ або ADEV, введене в 1966 році є найчастіше використовуваною мірою стабільності частоти у часовій області[19]. Якщо коротко розглянути, то дане відхилення було розроблене як альтернатива стандартному «σ», оскільки останнє не часто відповідає постійному значенню для мерехтіння і навіть більш розбіжних процесів FM-шуму[3] пов'язаних з джерелами частоти. ADEV[21] використовує першу різницю частоти замість середньої. Індекс *«y»* в ADEV позначає безрозмірну дробову частоту:

$$y(t) = (f(t)/f_0 \quad (1.15)$$

а її залежність τ вказує на те, що вона є функцією часу усереднення. Випадкове відхилення фази, в одиницях секунд, позначаються $x(t)$. Частота є швидкістю зміни фази:

$$y(t) = dt(t)/dt \quad (1.16)$$

Маючи на увазі різницю між статистикою та її оцінкою[4], можна обчислити відхилення Аллана за допомогою декількох методів: його оригінального виду, який повністю перекривається ADEV формулою; або більш детально за допомогою інших, дещо упереджених оцінок.

Блок-схема, що показує процес вимірювання ADEV[19], показана на рисунку 1.3**,** де коливання частоти джерела $y(t)$ входять зліва, а результати

ADEV отримуються з правої сторони. Блок, що містить h_τ і h_Δ представляє функцію вибірки ADEV, яка вводиться як фільтрація низьких, так високих частот, а також має певну пропускну здатність. Останній блок — це оператор статистичного очікування.

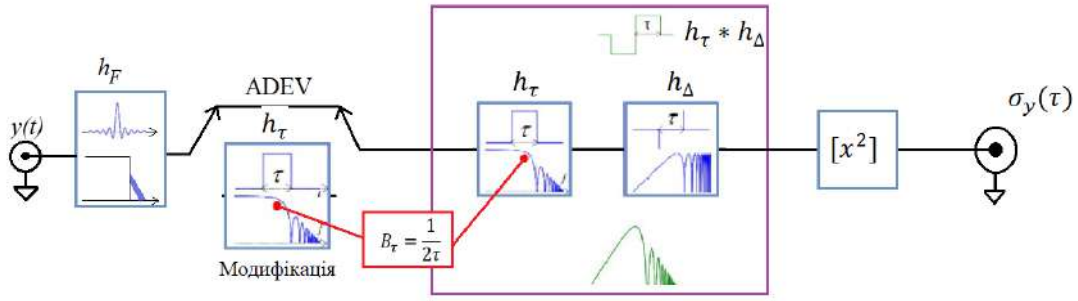

Рисунок 1.3 – Блок-схема вимірювання ADEV

Розгляд пропускної здатності для аналізу ADEV можна розділити на три категорії: вихідного апаратного забезпечення; процесу вибірки вимірювань та програмного забезпечення для аналізу. Таким чином, процес вибірки вимірювання зазвичай підлягає накладенню і впливає на спектр властивості даних, але це не впливає на загальну потужність шуму чи дисперсію, а пропускна спроможність програмного забезпечення для аналізу є ще меншою, наполовину від зворотної величини аналізу для ADEV і також звужується коефіцієнтом усереднення за допомогою фазового усереднення MDEV.[19]

Дисперсія Аллана[3] має безліч формулювань. Початковою є дисперсія Аллана, що перекривається. Класична формула відхилення Аллана включає лише дві послідовні точки обчислення. Тому вона називається Дисперсія двох вибірок:

$$\sigma^2_{ADEV}(\tau) = \frac{1}{2}\langle \bar{y}_n + 1 - \bar{y}_n)^2\rangle \qquad (1.17)$$

де $\bar{y}_n$ – вибрана точка даних[21]

Часто також використовується метод перекриття:

$$\sigma^2_{O\ ADEV}(m\tau_0) = \frac{1}{2((m\tau_0)^2(N-2m)}\sum_{n=1}^{N-2m}(x_n + 2m - 2x_{n+1m} + x_n)^2 \qquad (1.18)$$

Даний метод рахує точки даних і середні значення σ , що є початковим.[21].

На рисунку 1.4 зображений довільний зразок кривої відхилення Аллана.

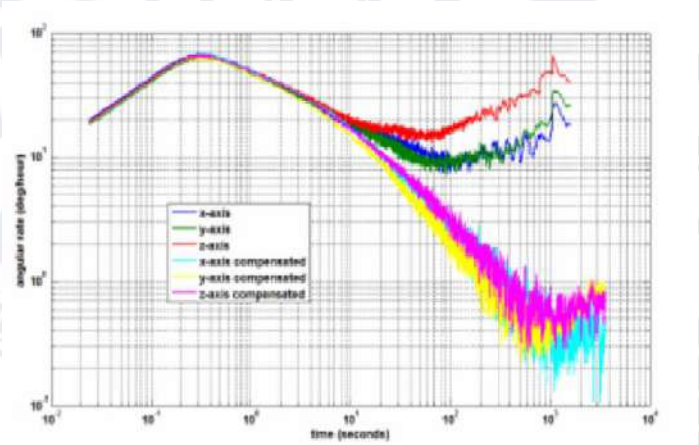

Рисунок 1.4 – Крива відхилення Аллана

Дане відхилення збільшується і до певного моменту часу, потім починає падати. Вісь x являє собою час, протягом якого дані згруповані. Від усереднення даних залежить зниження відхилення[16].

Початкові часові кластери є представниками високочастотних шумів. Дане відхилення швидко зменшується при нахилі однієї із половин і описує можливий білий шум. Точка кривої, що розташована внизу, визначається стійкістю зміщення, це є критичним параметром шуму.[3].

За межами верхньої та нижньої точок відхилення збільшується, це означає, що низькочастотний шум домінує.. Біла шумова зона має назву – Angle Random Walk (ARW) або iVelocity Random Walk (VRW) Коефіцієнт випадкового блукання знаходиться за допомогою нахилу лінії в певний кластерний час [21].

1.4 Нейронна мережа IoT

Мережі IoT відрізняються від традиційних тим, що мають плоску сітчасту структуру, а ось більшість інших систем працюють за простою централізованою побудовою. Розглянувши систему Інтернету речей, що складається із давачів, які мають спільні риси (одного і того ж власника, або місцезнаходження в одному

середовищі тощо), можна побачити, що їхнє функціонування залежить від підключення до відповідного центру збору даних.[17]

У середовищі IoT можна спробувати прискорити оцінку роботи нейронної мережі, розподіливши її частини між усіма доступними пристроями. Проте комунікаційні витрати з розповсюдження нейронних мереж швидко стають обмежуючим фактором. Підхід в основному використовується для прискорення етапу навчання на кластері вузлів[12], що підключені через високошвидкісну мережу.

Для об'єднання різних окремих центрів, часто використовується ієрархічна модель рівнів збору даних[13](від давачів до хмарного простору).Базова архітектура цього методу зображена на рисунку 1.5 .

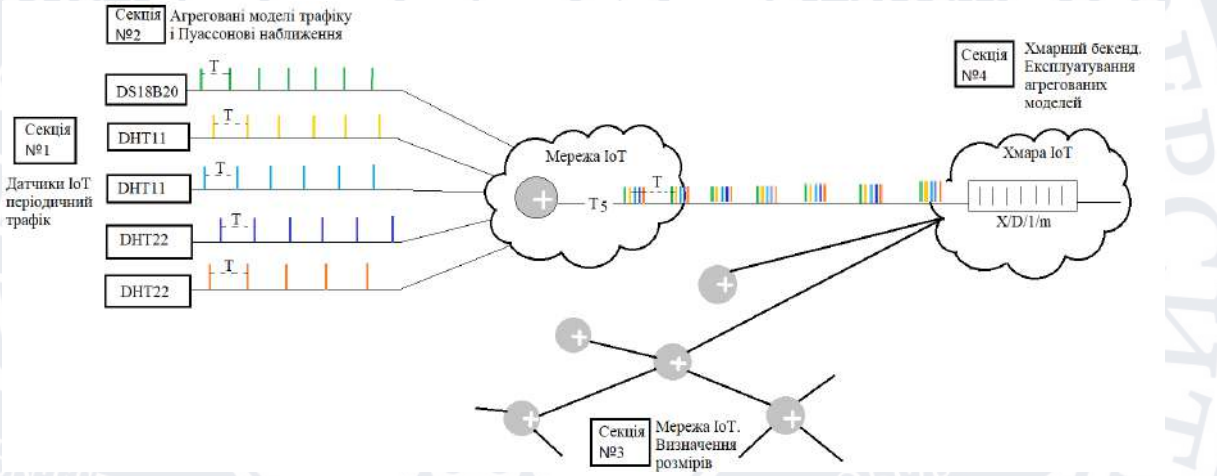

Рисунок 1.5 – Базова архітектура ієрархічної моделі нейронної мережі

Розташовані у секції №2 APP і Пуассонові наближення обробляють збір трафіку, який в свою чергу показує приклад використання хмарних обчислень Інтернету речей і демонструє точність і доречність використання процесу Пуассона. Отримані результати збору застосовуються до хмарного автоматичного масштабування разом із експериментальною оцінкою даних в секції №3, крайній етап завершує цю роботу.[12]

Крім базової архітектури, до пристроїв Інтернету речей можна застосувати нейронну мережу типу Big-Little[17]. Вона класифікується лише підмножиною вихідних класів і може виконуватися локально з обмеженою потужністю. Коли

маленька нейронна мережа не може класифікувати вхідну вибірку, робиться запит до великої мережі, що працює в хмарі. Архітектура нейронної мережі зображена на рисунку 1.6.

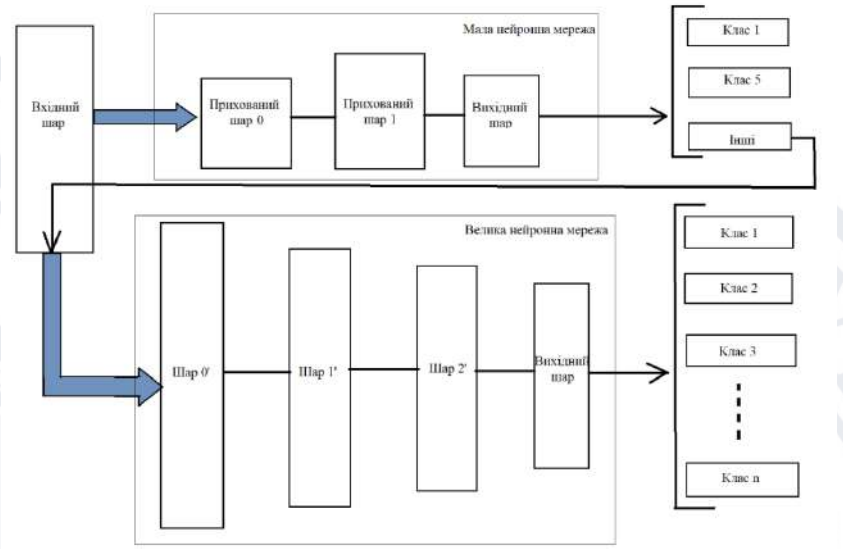

Рисунок 1.6 –Нейронна мережа типу Big-Little

Впровадження нейронної мережі несе за собою ряд проблем, одною із яких є технології радіо доступу для IoT. Тому в багатьох налаштуваннях, таких як домашня автоматизація бажано мати зв'язок з персональною мережею (PAN). (наприклад, за допомогою інтерфейсу з сімейства IEEE 802.15.4), інші сценарії вимагають радіочастотних інтерфейсів із більшою зоною покриття щоб досягти вузла агрегації. Це може бути забезпечено новими форми радіозв’язку ультра вузького діапазону (UNB) та відкриває абсолютно нову категорію способів спілкування, що має назву так, як малопотужна глобальна мережа (LPWAN)[20]. Це загальний термін включає широкий спектр різних протоколів, включаючи LoRa на основі розширеного спектру та його стандартизований стек канального рівня LoRaWAN.

На діючих стандартах LPWAN можуть виникнути певні перевантаження, які мало чим відрізняються від Wi-Fi 2,4 ГГц, де заняття смуги і пов'язані з цим зіткнення та перешкоди стали тривожно високими, особливо в сучасному світі. Через набагато більший діапазон LoRa[20] функціонує більш менш стабільно,

але перенавантаження може стати реальністю рано чи пізно. Тому дотримання суворої комунікаційної дисципліни та мінімізація кількість переданих даних і ефірний час трансивера може полегшити ситуацію.

Інші варіанти LPWAN включають нові дружні до Інтернету речей варіанти стільникових мереж 3GPP, зокрема вузько смугового Інтернету речей (NB-IoT), які зараз починають застосовувати оператори мережі. У той час як LoRa вказує лише дуже дрібний стек протоколів і залишає всі інші деталі, NB-IoT несе звичайний глибокий стек 3GPP. Завдяки широкому вибору стеків протоколів і радіочастотних інтерфейсів в IoT з'являється кілька різних моделей зв'язку. Використовуючи підхід RESTful або опублікування/підписки, є можливість постійно надсилати великі обсяги даних або бути обмеженим мінімальними даними, але при цьому отримати великі періоди й це безпосередньо буде впливати на характеристики руху.[20]

1.5. Висновки по розділу 1

У даному розділі розглянуто інформацію про вимірювання, як випадкових процесів. Визначено тип розподілу Пуассона, основною причиною є саме визначена кількість вимірів, адже вимірювання температури обмежене діапазоном та часом. Сигнали, що приходять від сенсорів мають тип «випадкових телеграфних сигналів».

Встановлені критерії точності та правила отримання достовірного результату. Проаналізовано особливості застосування дисперсії Аллана, необхідність впровадження нейронної мережі та проблеми які виникають в ході розробки.

З огляду на те, що подібні системи переважно мають більшу кількість давачів, прийнято рішення розширити експериментальний макет, але в такому разі виникає потреба у виконанні основної технічної задачі, що полягає у підвищенні надійності за рахунок створення автономної системи, яка здатна самостійно накопичувати вимірювальні дані. Крім того складові цього розділу допоможуть при аналізі отриманих результатів. та при вирішенні основної мети, яка полягає у виявленні порушень звичайного режиму роботи системи.

РОЗДІЛ 2. ВИМІРЮВАЛЬНА СИСТЕМА ІНТЕРНЕТУ РЕЧЕЙ

В наш час IoT відноситься до багатьох фізичних пристроїв по всьому світу, які підключені до Інтернету і оброблюють й аналізують велику кількість даних. Прогнозується, що в майбутньому інтернет-речі будуть активними учасниками бізнесу, інформаційних та соціальних проектів, при цьому взаємодія відбуватиметься без потреби втручання людини.

Розглядаючи систему вимірювання температури, що включала в себе п’ять давачів температури та плату Arduino Uno, було прийняте рішення розширити кількість сенсорів у два рази та зробити дану систему автономною за допомогою годинника реального часу, а також модуля SD-карти. Маючи нову інтерпретацію, можна задуматись над впровадженням нейронної мережі, яка в свою чергу самостійно зможе відрізняти давачі один від одного. Це дозволить системі бути більш надійною та у разі аварійної ситуації реагувати швидше.

2.1. Програмування для використання в IoT

Експеримент відбувається за рахунок зчитування інформації від давачів через ПК, за допомогою коду розробленого набором інструментів PlatrofmIO та IDE VSCode(рис.2.1).

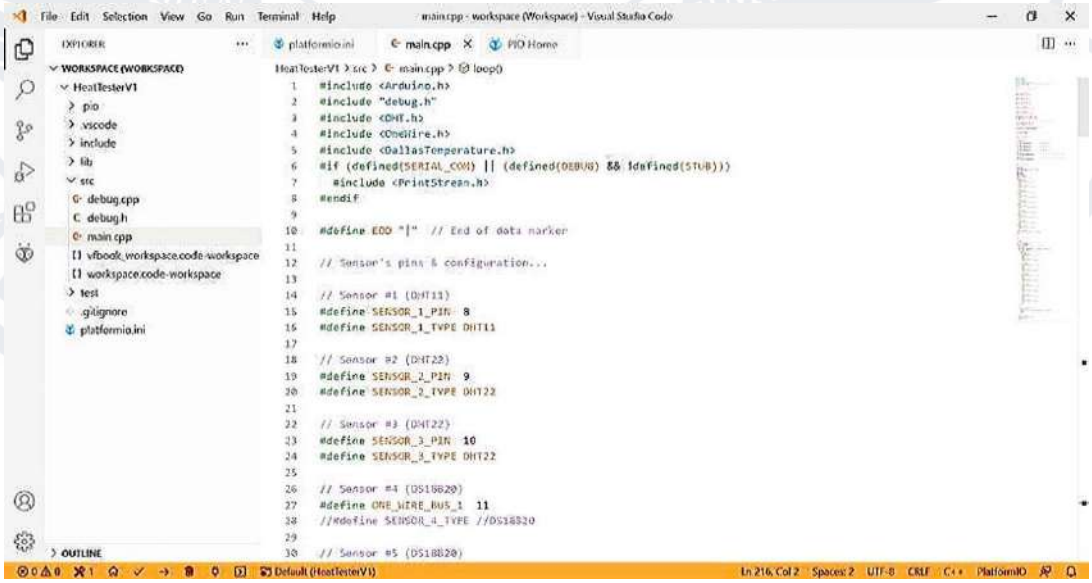

Рисунок 2.1 – Програма контролера у середовищі Visual Code Studio

Кожен давач має вхід на платі Arduino UNO. В оновленій версії велика кількість давачів й кожен із них відповідає порту підключення відповідно: типу DHT22 (4 екземпляри) підключені до портів 6-9, DHT11 підключається до 2-5, а DS18B20 – до 14 та 15 порту відповідно, але з урахуванням двох режимів 12 біт та 9 біт.

Збільшуючи швидкість циклу передачі даних, це стає помітним джерелом тепла, що негативно впливає на надійність та безперервність роботи сенсорів. Програмним кодом також передбачений контроль підключення окремих датчиків до системи.

Для ініціалізації та перевірки у програмі застосовується конструкція «if/else», яка перевіряє істинність певної умови, а також в залежності від результатів перевірки, виконує певний код. Умова ставиться після ключового слова. Вона повинна представляти значення «bool». Це може бути саме значення, або результат умовного виразу, який повертає значення «bool». І якщо ця умова вірна, то спрацьовує код, який поміщається далі після стану всередині фігурних дужок. Ініціалізація сенсорів відображена у Додатку, а також є ідентичною до попередньої версії написання коду, за винятком останнього виразу, що відповідає годиннику реального часу DS3231.

За процедуру опитування відповідає команда «poll_sensors()». Також програма містить варіанти режимів у яких їй доведеться працювати, вони виглядають таким чином:

#define TARGET 0 // 0 - autonomous (full), 1 - autonomous (logged), 2 - semi-autonomous (logged), 3 - interactive (serial communication)

В даному рядку:

0 – режим роботи повністю автономної системи із наявністю флешки;

1, 2 – відповідають за опитування сенсорів, другий відрізняється від першого, переводом значень у цілі числа, задля економії пам’яті Arduino;

3 – для роботи у стандартному режимі з ПК.

Налаштування роздільної здатності DS18B20 виконується за допомогою команди «sensors0.setResolution(DS18_0_RES)» для обох сенсорів цього типу, вона встановлює дозвіл на 12 біт та 9 біт.

У спеціально розробленій IDE Lazarus програмою Ar5sen, виконується накопичення даних для подальшого статистичного аналізу. Крім того визначаються значення у реальному часі.

У автономній системі присутня карта пам’яті, на яку виконується запис даних від давачів. В результаті отримуємо такий самий текстовий файл із температурними показниками, як і у випадку збору даних за допомогою ПК, але різниця в тому, що кожен сенсор має окремий файл показників.

Насамперед визначається команда, де контролер зчитує завдання. Якщо файл існує і в ньому є рядки із завданням, функція надає їх кількість у файлі. Після цього відбувається основний цикл «void loop()», в якому виконується підготовка файлів для запису даних нового завдання. Безпосередньо вимірювання та запис виконується за допомогою команди «poll_ticker.update()», також перевіряється виконання, зчитування нового завдання й зупинка програми за потреби.

Для зчитування даних присутній таймер «poll_ticker.update()», він відпрацьовує процедуру «do_measure()» з інтервалом завдання, що вказується у змінній tau. Таким чином після перевірки, налаштовується таймер на нове завдання. Назви для файлів програма вигадує, дивлячись на годинник у початковий момент вимірювання температури, а далі розподіляє готові файли по папкам із назвами давачів. Всі функції програми, зазначених вище показані у Додатку.

Експеримент складається зі збору даних , що накопичуються протягом певного часу від давачів в оновленій версії на SD-карту.

Передбачається статистичний аналіз зібраних від різних давачів даних у програмі AlaVar 5.2.

2.2. Опис експериментального макету

Попередня версія системи сенсорного вузла IoT складалася з плати Arduino Uno, давачів температури типів DS18B20, DHT11,DHT22, підключення відбувалося через USB з’єднання до ПК під керуванням ОС Windows 10. Разом одночасно використовувалися п’ять давачів температури.

В оновленій версії даної системи використовуються такі сенсори температури як, DS18B20-12bit (1 екземпляр), DS18B20-9bit (1 екземпляр), DS3231 (1 екземпляр), а також давачі температури та вологості двох типів DHT11 (4 екземпляри) та DHT22 (4 екземпляри). Підключення проводиться ідентично до попередньої версії.

Arduino Uno – це прилад в основі якого розташований мікроконтролер ATmega328 (рисунок 2.2). [18] Він має 14 цифрових входів/виходів, роз'єм USB, живлення, роз'єм для програмування (ICSP), кнопка скидання. Характеристики Arduino Uno наведено в таблиці 2.1.

Рисунок 2.4 – Плата Arduino Uno

Таблиця 2.1 – Характеристики Arduino Uno

Мікроконтролер	ATmega328
Робоча напруга	5 В
Напруга живлення (рекомендована)	7-12 В
Аналогові входи	6
Максимальний струм одного виводу	40 мА
Максимальний вихідний струм виводу	3.3V 50 мА

DHT11 (рис.2.5) містить кілька вимірювальних приладів – ємнісний датчик(вимірює температуру) температури і гігрометр(вологість). В ролі АЦП працює, чіп, що розміщений всередині. Він випромінює цифровий сигнал, що зчитується мікроконтролером.[9]

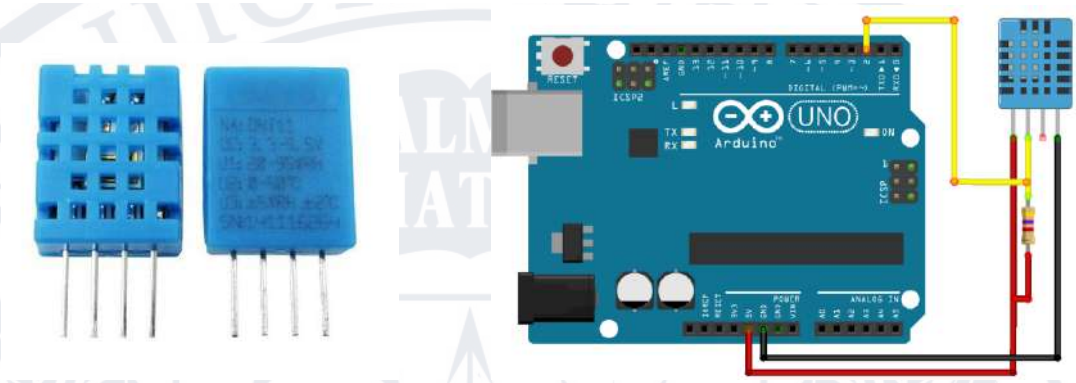

Рисунок 2.5 – Давач DHT11 та схема його підключення DHT11

DHT22 – давач для визначення вологості і температури, що складаються з ємнісного датчика вологості і термістора. Подібно DHT11 давач має АЦП. Протокол даних давачів використовує одну шину з відкритим колектором.

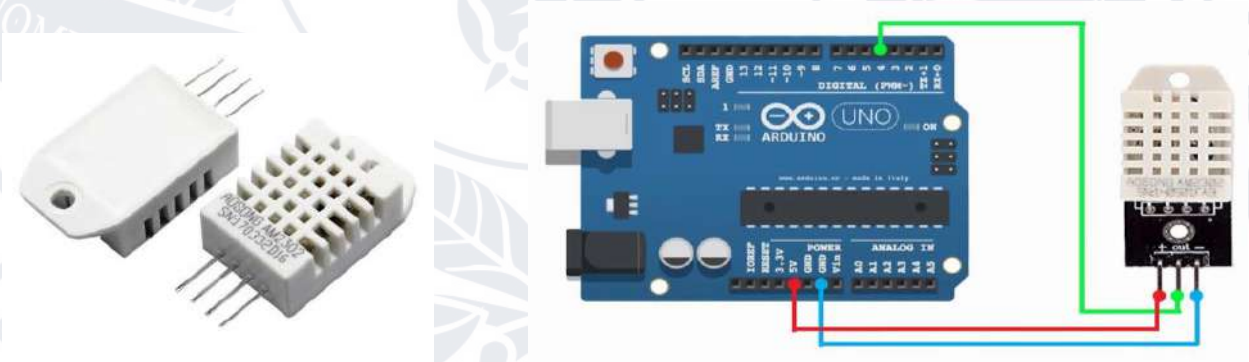

Рисунок 2.6 – Давач DHT22, схема його підключення

Основна відмінність між DHT11 і DHT22 полягає в тому, що DHT22 має більшу роздільну здатність і є більш точним. Підключення давача зображено на рисунку 2.3, а характеристики DHT22 показані на табл. 2.1 [10].

DS18B20 – давач цифрового типу вимірювання температури, що має роздільну здатність 9-12 цифр. У пам’яті давача зберігаються встановлені користувачем параметри.

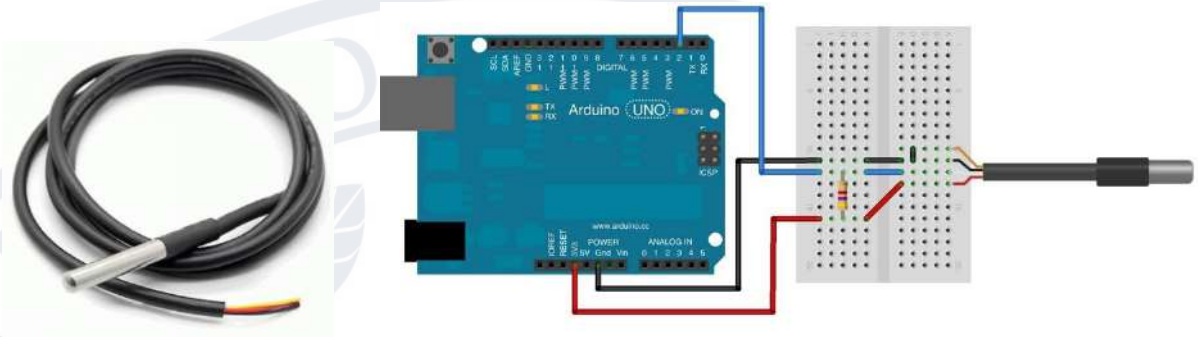

Рисунок 2.7 – Вимірювач температури DS18B20 та схема його підключення

Кожен чіп DS18B20 має унікальний 64-бітний код. Він дозволяє кільком давачам мати підключення до загальної шини[11].

. Оперативна пам'ять містить 2-байтовий регістр температури, який зберігає її значення по закінченню процесу вимірювання. Також у пам’яті EEPROM є два однобайтових регістра контролю температури (T_h і T_l) і регістр конфігурації. Регістр конфігурації дозволяє користувачеві у цифровому перетворювачі температури встановлювати розрядність (9, 10, 11, 12 біт), що впливає на час конвертації температури. T_h і T_l регістри і регістр конфігурації є комірками енергонезалежної пам'яті даних (EEPROM). Таким чином, вони зберігають дані, коли прилад вимкнено.

Дані сенсори мають конфігураційні регістри. У них задається температурна розрядність датчиків DS18B20. Початкове значення регістра залежить від вмісту EEPROM-пам'яті. Формат конфігураційного регістра представлено на таблиці 2.2, де R1, R0 – біти, що задають температурну розрядність термометра, від якої залежить максимальний час перетворення температури (таблиця 2.3).

Таблиця 2.2 – Формат конфігураційного регістра

Біт 7	Біт 6	Біт 5	Біт 4	Біт 3	Біт 2	Біт 1	Біт 0
0	R1	R0	1	1	1	1	1

Таблиця 2.3 – Залежність розрядності та максимального часу перетворення від значень бітів

R1	R0	Розрядність	Час перетворення(max.)
0	0	9 біт	93,75 мс
1	1	12 біт	750 мс

В таблиці 2.4 наведені метрологічні характеристики досліджуваних давачів.

Таблиця 2.4 – Особливості давачів температури

Тип	DHT11	DHT22 (AM2302)	DS18B20
Діапазон температур	від 0 до 50ºC	від –40 до 80ºC	від –55 до 125ºC
Похибка (по документації)	+/– 2ºC (від 0 до 50ºC)	+/– 0.5ºC (від –40 до 80ºC)	+/– 0.5ºC (від –10 до 85ºC)
Роздільна здатність (фактична)	0.1ºC	0.1ºC	1)0.06ºC, 2)0.5ºC

DS3231 – це модуль, що містить в собі точний годинник реального часу (RTC) та температурний давач.

Модуль включає в себе літієву батарею, яка забезпечує безперебійну роботу джерела живлення(рисунок 2.8). Вбудований генератор забезпечує точність пристрою і зменшення кількості компонентів.

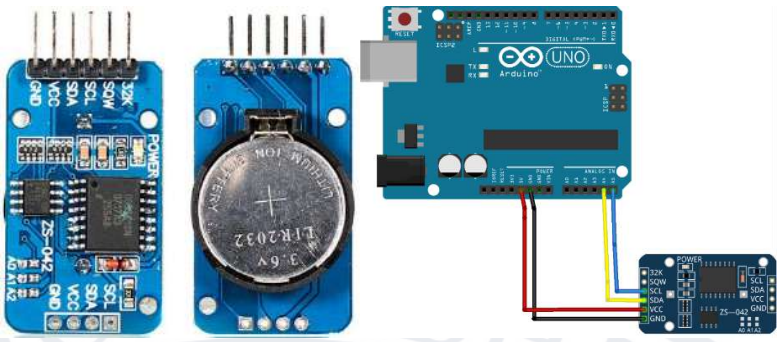

Рисунок 2.8 Плата модуля DS3231 та схема його підключення

Таблиця 2.4 – Технічні параметри модуля DS3231

Напруга живлення	5 В
Точність	±0,432 сек в день
Кварцова частота	32.768 кГц
Підтримуваний протокол	I2C
Мікросхема пам’яті	AT24C32 (32КБ)

В модулі DS3231 встановлені кварцовий осцилятор і датчик температури, який компенсує перепади температур, щоб час залишався точним. Також чіп DS3231 підтримує роботу годинника в двох форматах 24 і 12 годин.

DS3231 має в собі вбудований температурний давач. Діапазон температури становить від -40°C до +85°C. Частотний вихід TCXO доступний на контакт 32 кГц. програмований прямокутний вихід. INT/SQW надає або сигнал переривання через стан тривоги або прямокутну хвилю на вихід. Схема опорної напруги та компаратора з компенсацією температури контролює рівень VCC, щоб виявляти збоїв живлення та автоматично переключатися на резервне джерело живлення, коли необхідно. Контакт RST забезпечує зовнішню кнопку функцією індикатора події збою живлення.

Модуль підключення Micro-SD карт призначений для їх використання в якості файлового зберігання інформації.(рисунок 2.9).

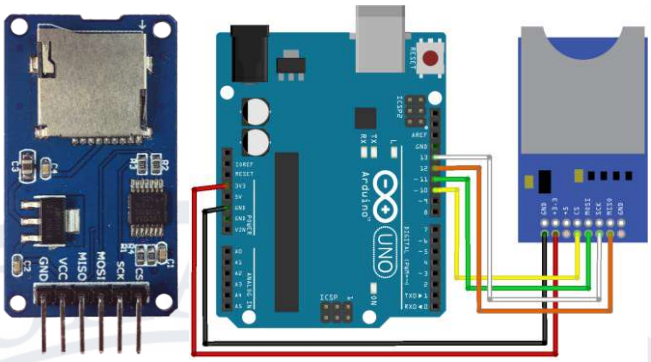

Рисунок 2.9 – Модуль підключення Micro-SD та схема підключення до Arduino UNO

Таблиця 2.5 – Характеристики модуля SD-карти

Напруга живлення	від 4.5 В до 5.5 В
Вбудований стабілізатор	AMS1117-3.3
Перетворювач рівнів	74LVC125
Логічні рівні	3.3В або 5В
Струм	0.2 мА до 200 мА

2.3. Розробка нейронної мережі

Для того щоб застосувати нейронну мережу до вибраної системи, потрібно зрозуміти за якою схемою вона працює. Найперше, що потрібно зазначити про математику нейронної мережі, це те, що вона дуже проста. Однак, у більшості випадків трапляється сотня тисяч нейронів, тому такі розрахунки можуть зайняти багато часу, а також їх основна частина включає в себе матриці(рисунок 2.10).

У нейронних мережах є таке поняття, як вага. Вона представляє собою зв'язок між нейронами, який несе значення. Чим вище значення, тим більша вага, і тим більше значення надається нейрону на вхідній стороні.

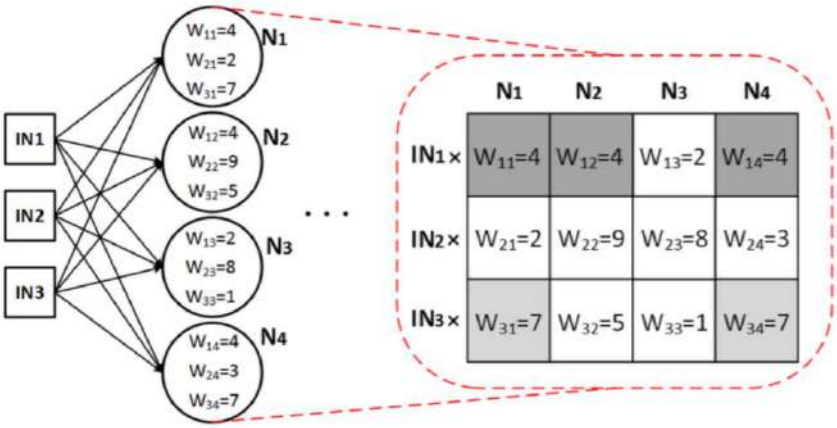

Рисунок 2.10 – Матриця нейронної мережі

Вхідний шар має 3 нейрони, а наступний шар (прихований) має 4, як зображено на рисунку 2.10. Таким чином створюється матриця з 3 рядків і 4 стовпців. Ця матриця буде мати назву W1. У випадку, присутньо більше шарів, тоді буде більше матриць W2, W3 і т.д.

Загалом, якщо шар L має N нейронів, а наступний шар L+1 має M нейрони, матриця ваги - це матриця N-на-M (N рядів і M стовпців).

Потрібно підкреслити, що найбільшим числом в матриці є W22, яке несе значення 9. Дане число з'єднує IN2 на вхідному шарі з N2 на прихованому шарі. Це означає, що в даний час N2 вважає, що вхід IN2 є найважливішим з усіх 3 входів, які він отримав при прийнятті власного крихітного рішення.

У нейронній мережі існують непередбачені або неспостережувані фактори. Вони називаються упередженнями. Кожен нейрон, якого немає на вхідному шарі, отримує зміщення, прикріплене до нього, і зміщення, як і вага, несе в собі значення.

Важливим кроком у роботі нейромережі є активація, вона означає, що нейрон повинен прийняти своє рішення на виході й повернути ще один вихід. Якщо позначити активацію, як $f(y)$, то y – агрегація всього входу. Існує дві категорії активації, лінійні і нелінійні. Якщо $f(y) = y$, то активація вважається лінійною(тобто нічого не відбувається), інші варіанти є нелінійними.[20]

На рисунку 2.11 показано нейрон, вхідні дані якого $(x_1 - x)$, відповідні ваги - $(w_1 - w)$, зміщення b і функції активації f застосовані до зваженої суми вхідних даних.

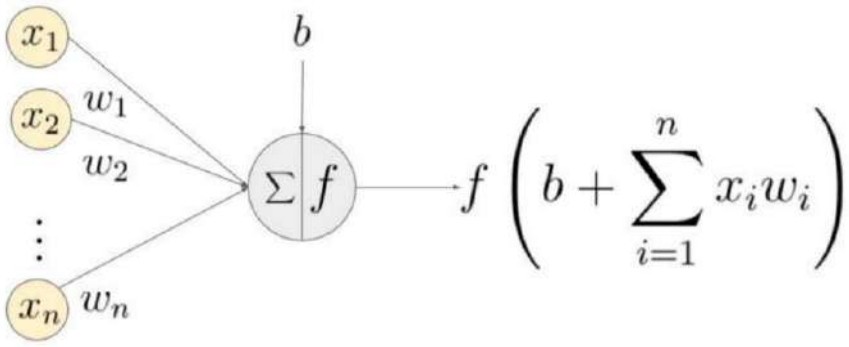

Рисунок – 2.11 Нейрон, що показує прийняття рішення

Для створення нейронної мережі у системі вимірювання температури потрібно заповнювати таблиці 2.6, 2.7, що містять у собі основні характеристики зібраних вимірювальних даних. Маючи такі дані нейромережу можна навчити відрізняти сенсори різних типів за їхніми особливостями.

Таблиця 2.6 – Вхідні дані нейромережі системи вимірювання температури

TAU	n	MIN	MEAN	MAX	0	1	2	3	4	5	6
[sec]	2^n	[C]	[C]	[C]	2^0=1	2^1=2	2^2=4	2^3=8	2^4=16	2^5=32	2^6=64
2,5	10		26.04		0.076	0.104	0.136	0.163	0.181	0.214	0.290

Таблиця 2.7 – Передбачення для нейронної мережі

DHT11	DHT22	DS18B20	DS18B20	DS18B20	DS18B20	DS3231
		9 bit	10 bit	11 bit	12 bit	
1	0	0	0	0	0	0

2.4 Висновки по розділу 2

В даному розділі виконана основна технічна задача кваліфікаційної роботи, а саме оновлення системи для вимірювання температури за допомогою допоміжних модулів. Нова інтерпретація дозволяє надійно проводити збір даних від давачів Інтернету речей, також проведено експеримент вимірювання температури на оновленій системі.

РОЗДІЛ 3. АНАЛІЗ ДОСЛІДНОЇ СИСТЕМИ

Дисперсія Аллана застосовується для часових послідовностей температури та вологості, але давачі типу DS18B20 вимірюють тільки температуру, тому в роботі розглянуті залежності вимірюваних даних температури.

3.1. Збір даних від давачів

Спостерігаючи за реакціями давачів на зміну температури, можна зазначити, що вони мають різну динаміку, відштовхуючись від типів та характеристик. Якщо порівнювати типи давачів, видно, що DS18B20 на відмінну від DHT22, DHT11 довше зберігає найвищу точку виміру температури, затримуючись там (похибка 0,5) , в той час як інші давачі реагують на зменшення температури й це видно на графіках. Крім того помітна різниця в динаміці вимірювання між давачами одного типу DHT11, вона залежить від положення сенсорів.

На рисунках 3.1 – 3.4 представлені часові та частотні характеристики процесу вимірювання для одинадцяти дослідних екземплярів(включаючи DS3231), які демонструють відмінності, що можуть бути присутні у потоці даних, зумовлені метрологічними характеристиками та конструкцією сенсорів.

На графіках можна бачити також відмінності у дискретизації значення вимірюваної величини між різними типами давачів.

Ще однією динамічною характеристикою давачів температури є крутизна фронтів зростання та зменшення значення величини при швидкісній зміні температури, що обумовлена розмірами та масивністю окремого типу давача та їх конструкцією.

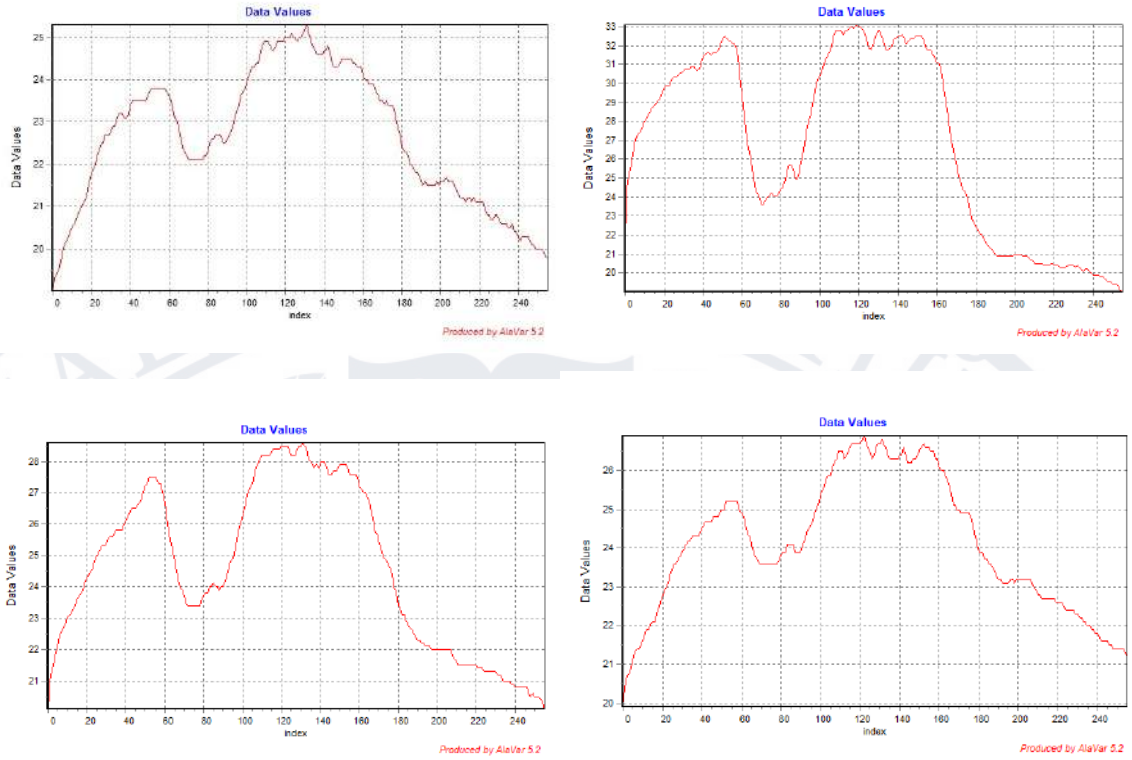

Рисунок 3.1 – Вхідний потік даних давачів типу DHT11(1-4 екземпляри)

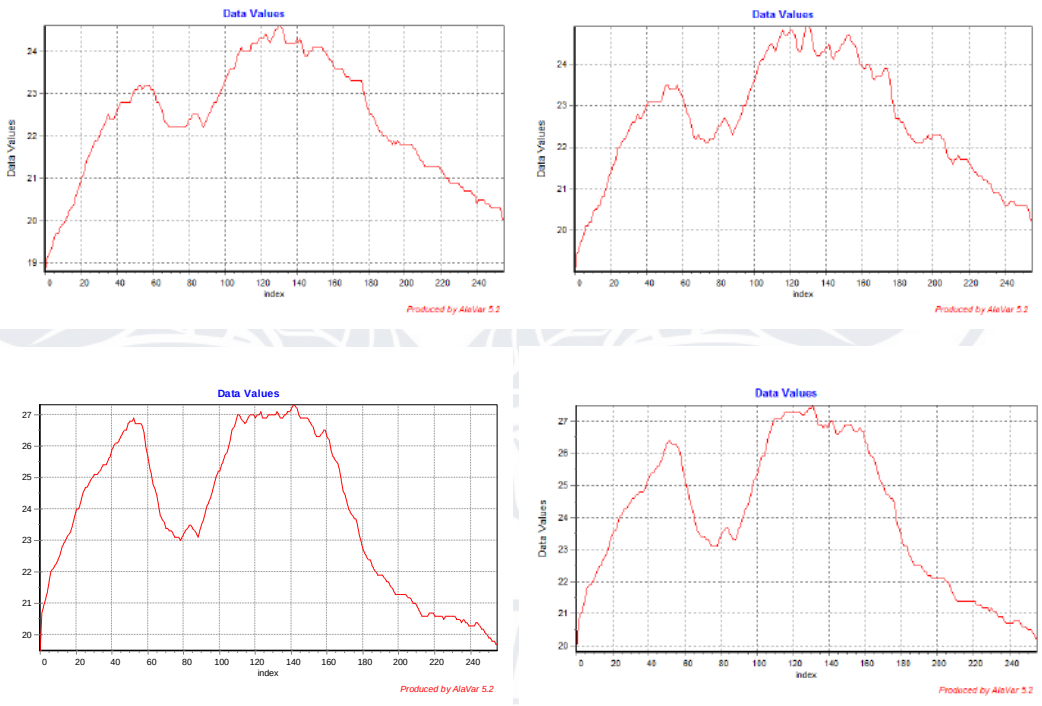

Рисунок 3.2 – Вхідний потік для давачів типу DHT22(1- 4 екземпляри)

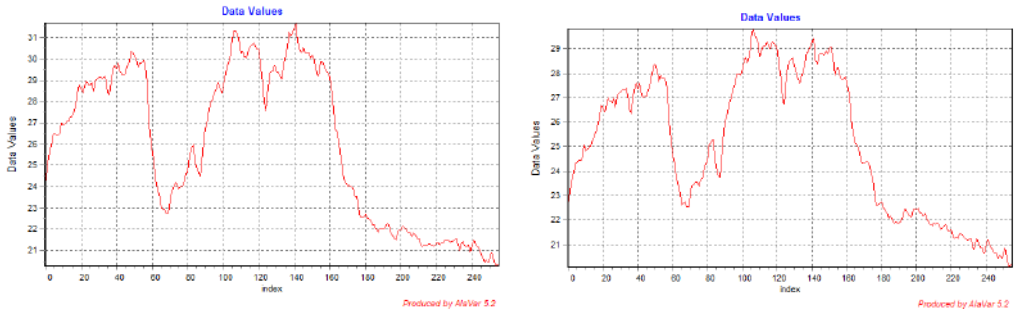

Рисунок 3.3 – Вхідний потік для давачів типу DS18B20(1-2 екземпляри)

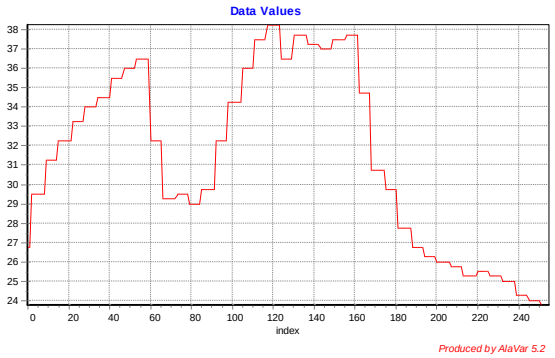

Рисунок 3.4 – Вхідний потік для давача температури модуля типу DS3231

3.2. Первинна обробка даних

Причина обробки сигналів полягає в необхідності отримання достовірних даних, що вимірюються сенсорами. Ці дані наявні в амплітуді сигналу, в частоті й в спектральному складі, у фазі й у часових залежностях сигналів. Графіки спектральної потужності для видів давачів зображені на рисунках 3.5 – 3.8.

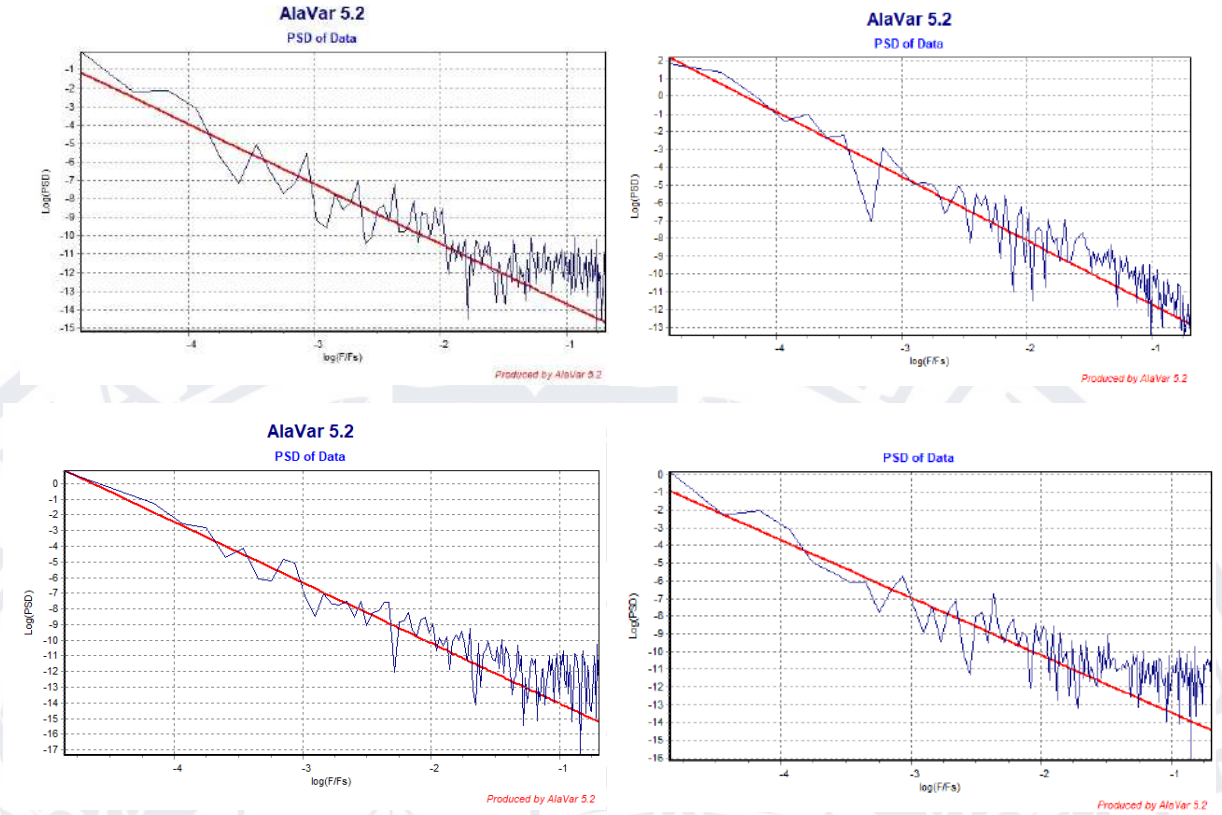

Рисунок 3.5 – Спектральна густина потужності даних давача типу DHT11

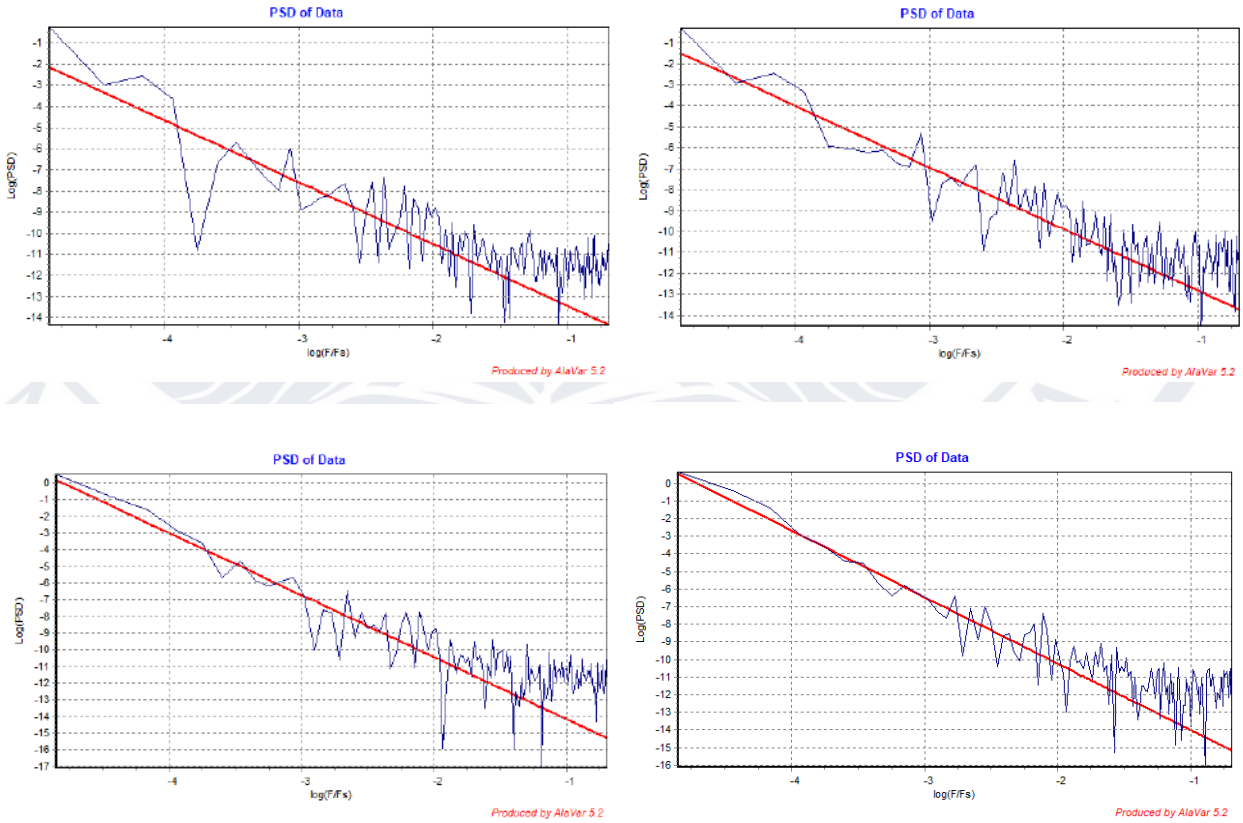

Рисунок 3.6 – Спектральна густина потужності даних давача типу DHT22

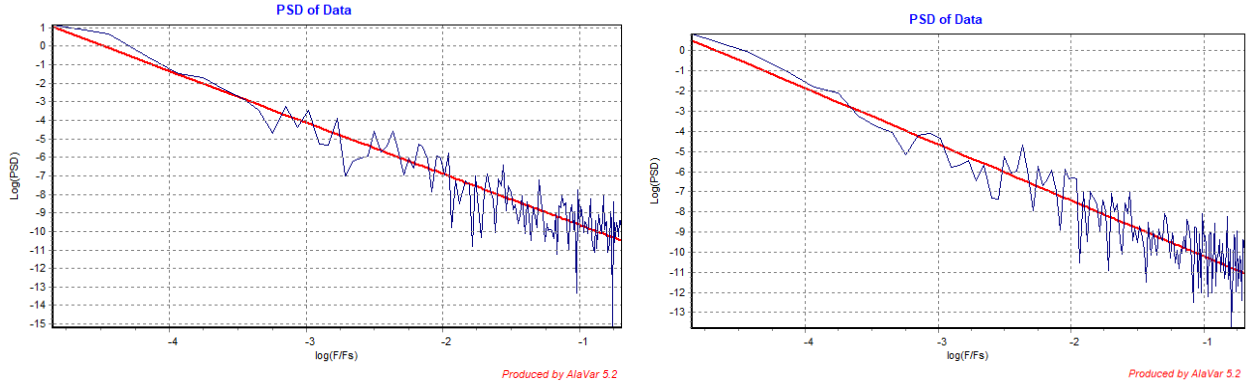

Рисунок 3.7 – Спектральна густина потужності даних давача типу DS18B20

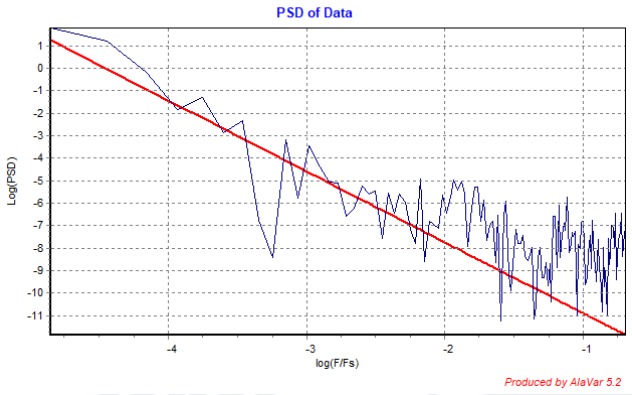

Рисунок 3.8 - Спектральна густина потужності даних давача модуля DS3231

Лінії на графіках, що показані вище, відповідають функціям спектральної густини, що наведені в таблиці 3.1.

Таблиця 3.1 – Характеристика давачів температури

Тип давача	DHT11	DHT22	DS18B20	DS3231
Спектральна густина потужності S(f)	1).$1/f^{-3.3}$ 2).$1/f^{-3.6}$ 3).$1/f^{-3.9}$ 4).$1/f^{-3.2}$	1).$1/f^{-2.9}$ 2).$1/f^{-2.9}$ 3).$1/f^{-3.8}$ 4).$1/f^{-3.7}$	1).$1/f^{-2.8}$ 2).$1/f^{-2.8}$	1).$1/f^{-3.2}$

3.3. Одночасне вимірювання великої кількості давачів

Завдання експерименту полягає в розрахунку дисперсії. За допомогою програми AlaVar 5.3 було розраховано показники відхилення Аллана (ADEV) [21], що представлені на табл. 3.2 та інші параметри процесу. На рисунках 3.9–3.12 показані графіки відхилення дисперсії Аллана (ADEV та MDEV) і Адамара (HDEV), TDEV[21].

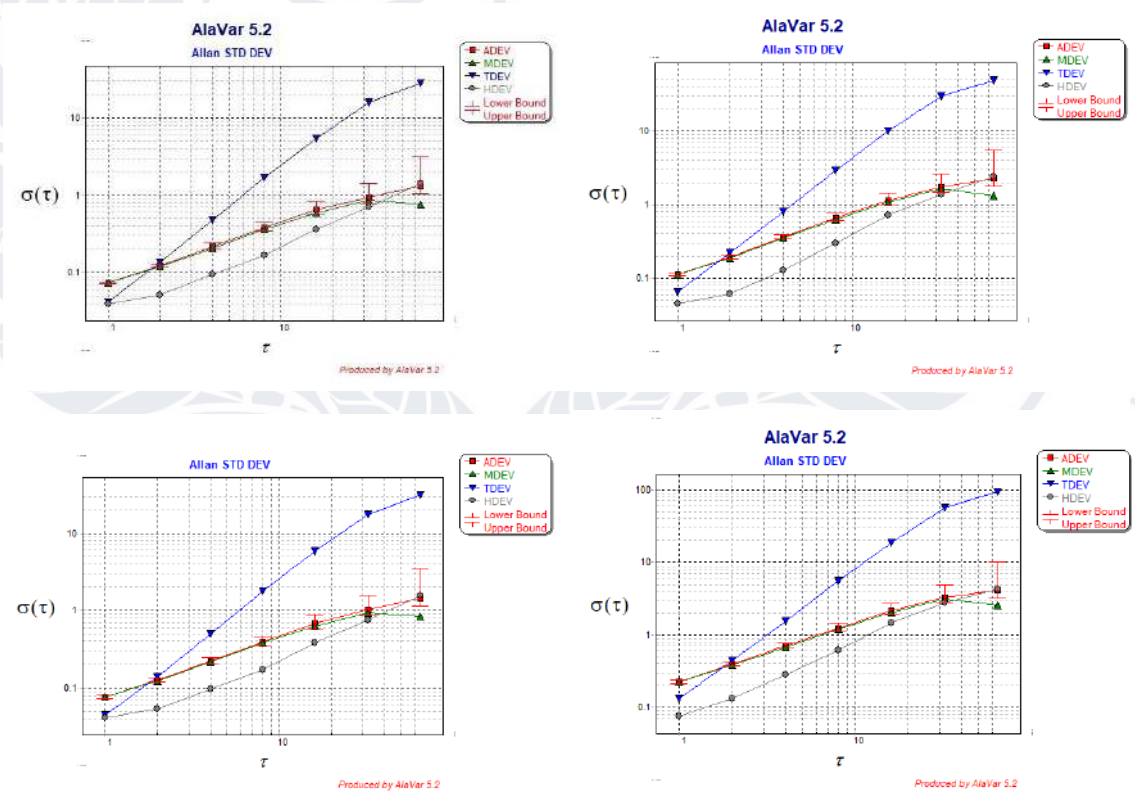

Рисунок 3.9 – Модифікації дисперсії Аллана давачів типу DHT11

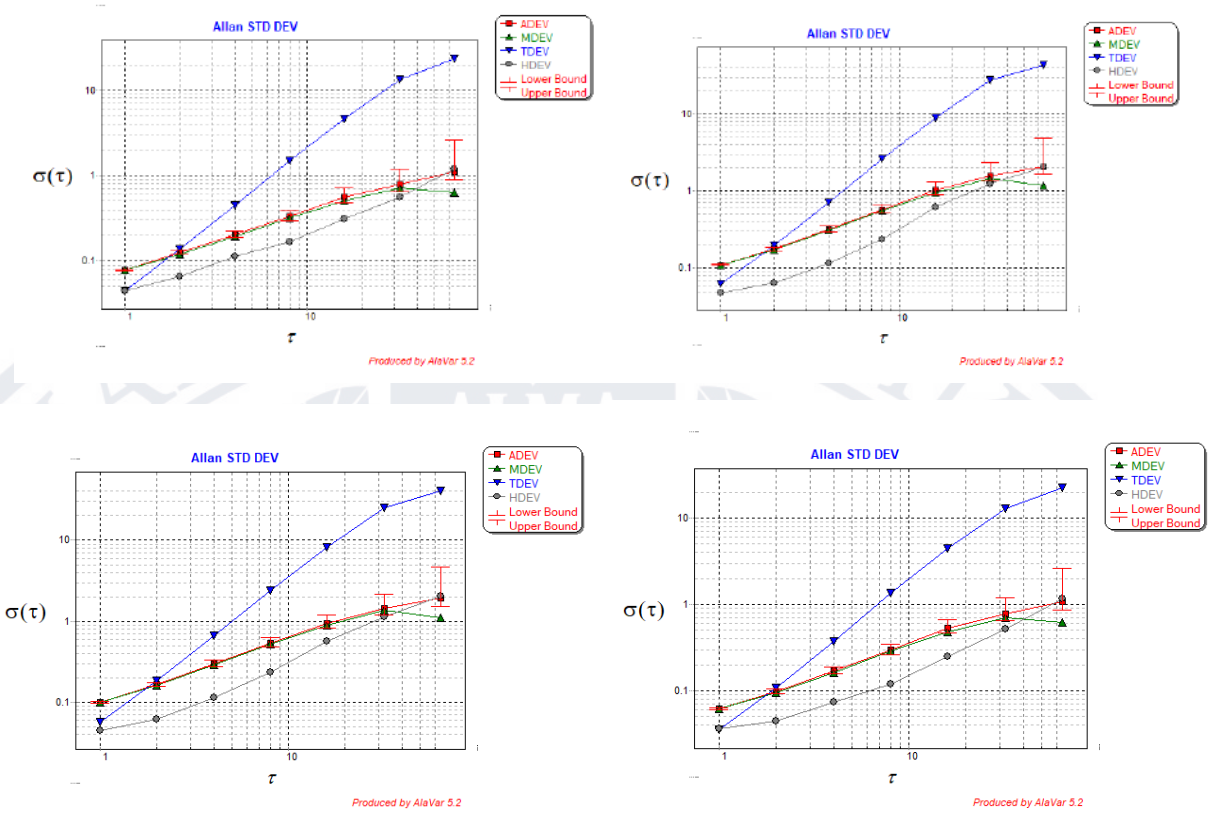

Рисунок 3.10 – Модифікації дисперсії Аллана давачів типу DHT22

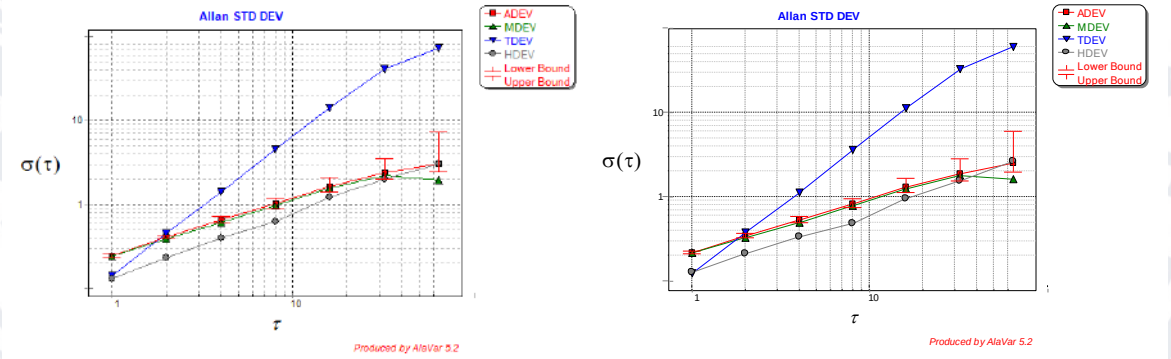

Рисунок 3.11 – Модифікації дисперсії Аллана давачів типу DS18B20(9, 12 bit)

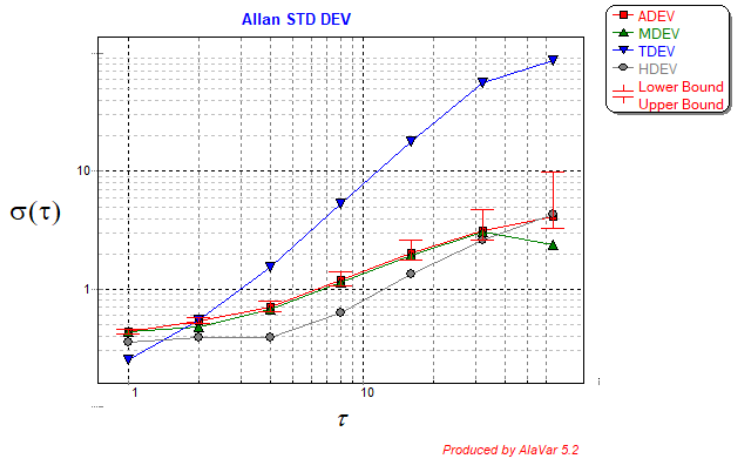

Рисунок 3.12 – Модифікації дисперсії Аллана давача модуля DS3231

В залежності від роздільної здатності досліджуваних датчиків, можна побачити на середній та крайній ділянці наведених графіків відмінності Вони зазначаються розбіжностями високочастотних коливань. У екземпляра DS18B20 криві вищі для відповідних значень τ. Також на графіках добре помітна різниця в нахилі ділянки з τ ~ 10 для різних типів датчиків.

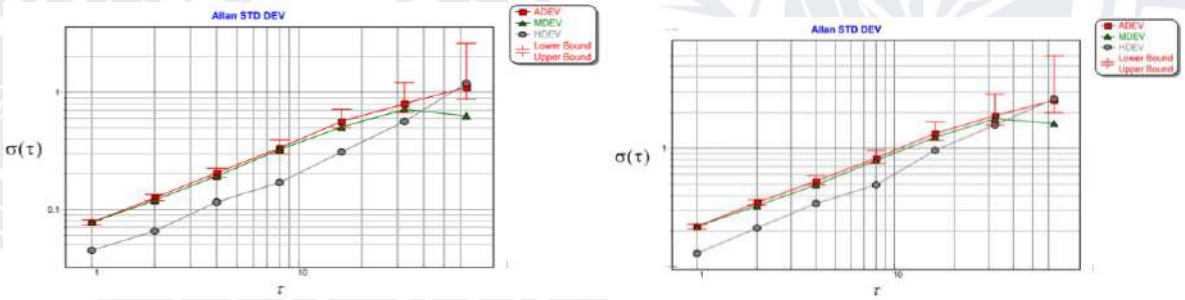

Рисунок 3.13 – Відхилення Аллана (ADEV та MDEV)
і відхилення Адамара (HDEV)

З наведених графіків, зазначимо, що кожний давач будь якого типу має певну особистий список метрологічних характеристик, яку використовують для ідентифікації типу або виявлення нетипової інформації, яка надходить.

Таблиця 3.2 –Дисперсія Аллана для давачів типу DHT11

TAU	DHT11(1)	DHT11(2)	DHT11(3)	DHT11(4)
1	0.073229427	0.22780711	0.11183146	0.078340634
2	0.12259162	0.3883559	0.19439554	0.12836383
4	0.21769134	0.70088195	0.35836536	0.22577121
8	0.38284505	1.2545927	0.65465904	0.40017339
16	0.65323435	2.1565039	1.1398868	0.69543308
32	0.94767607	3.2667096	1.7191224	1.0298786
64	1.3232136	4.1896139	2.2863227	1.4539833

Таблиця 3.3 – Співвідношення дисперсії Аллана для давачів типу DHT22

TAU	DHT22(1)	DHT22(2)	DHT22(3)	DHT22(4)
1	0.061357199	0.077712391	0.10846523	0.098816526
2	0.097473012	0.12471311	0.17604852	0.16453011
4	0.16926192	0.20485755	0.31686213	0.29672112
8	0.30146223	0.33727613	0.57585348	0.53771738
16	0.53023964	0.5632364	1.0202358	0.94399036
32	0.78526245	0.79877192	1.5565127	1.4364961
64	1.0924481	1.1063047	2.0377489	1.9285518

Таблиця 3.4 – Співвідношення дисперсії Аллана для давачів типу DS18B20, модуля DS3231

TAU	DS18B20(1)	DS18B20(1)	DS3231
1	0.2475753	0.21610183	0.43555239
2	0.41029724	0.34304798	0.53204501
4	0.65830832	0.52731011	0.71162932
8	1.0459726	0.82321474	1.2029031
16	1.6671118	1.3206908	2.0416992

32	2.3926261	1.8839477	3.1620474
64	3.0764357	2.5300138	4.1104968

У експериментальному макеті міститься велика кількість давачів різних типів. У процесі проведення експерименту зустрілися із низкою особливостей, що відповідають саме взаємодії сенсорів та їх вимірюваній діяльності в одному й тому ж самому середовищі. Для того, щоб отримати максимально наближені значення й зрозуміти який давач теоретично найбільш вірно показує температуру, застосовується класична теорія вимірювання.

У ході експериментального дослідження одним із основних завдань є встановлення істинного значення вимірювальної величини. Дане завдання вважається за окремим випадком статистичної задачі. Кінцева оцінка, обрахована за даними, виражається окремим числом та є функцією і відповідно випадковою величиною. За оцінку величини приймаємо середнє значення температури кожного давача.

Із таблиці 3.5 визначимо діапазон вимірювальних величин.. Для того щоб дізнатися наближене значення до дійсного й визначити давач, який є теоретично найбільш точним, потрібно результат будь-якого виміру порівняти із середнім значенням даного діапазону.

Таблиця 3.5 – Середні значення вимірювань

Тип давача	Значення температури [ºC]
DHT11(1)	22.624258
DHT11(2)	26.759766
DHT11(3)	24.606211
DHT11(4)	24.103125
DHT22(1)	22.353906
DHT22(2)	22.639844
DHT22(3)	23.873047
DHT22(4)	24.017578

DS18B20(1)	25.940898
DS18B20(2)	25.053555
DS3231	31.388672

Середнє значення температури дорівнює 24.8509873ºC(дані від всіх давачів). Значення є в межах похибок для всіх давачів окрім одного екземпляру DS3231, причиною є те, що даний давач – частина модуля, який виступає в ролі годинника.

Із розрахунків, які проведені, визначено, що значення давача DS18B20(12 bit), приближене до нуля, а середнє температурне, до середнього значення сукупних даних усіх давачів. Це означає, що теоретично даний давач є найбільш точним серед усіх запропонованих.

3.4 Висновки по розділу 3

За допомогою теорії вимірювання визначено дані, що наближені до середнього значення сукупності всіх давачів. Продемонстровано спектри густини потужності та наявність шуму і його тип для даних визначених давачів.

Проаналізувавши значення від сенсорів, встановлено, що за допомогою дисперсії Аллана можна побачити дані про загальні характеристики часових залежностей, на основі показників, що визначають стан системи.

Виконанні у роботі спостереження підтверджують, що сенсори мають відмінності у потоці даних, дискретизації значення вимірюваної величини та у властивостях особистої сигнатури метрологічних характеристик, дану інформацію застосовують для подальшого аналізу, що може знадобитися при виявлення моменту аварійної ситуації або зовнішнього втручання у систему.

ВИСНОВКИ

Створено експериментальну установку для вимірювання температури, що є автономною, це забезпечує надійність збору даних.

Проведено експеримент, у якому виміряно температурні значення від десяти екземплярів давачів у на скільки це можливо ідентичних умовах для подальшого аналізу.

За результатами обробки встановлено, що давачі як різних, так і однакових типів, мають статистичні відмінності, які використовуються для подальшого аналізу, ідентифікації та виявлення відхилень від штатного режиму роботи.

Список використаних джерел

1. Бахрушин В.Є. Методи аналізу даних / Запоріжжя 2011. – 268 с.
2. Гумен, М. Б. Аналіз випадкових процесів / М. Б. Гумен, В. М. Співак. С. К. Мещанінов, Г. Г. Власюк / / Основи теорії процесів в інформаційних системах. /: Київ. «Кафедра», 2017 – 328 с.
3. Крижановський В. Г., Комаров В. Ф., Сергієнко С. П., Загоруйко Л. В. Використання дисперсії Аллана для ідентифікації нормальної роботи сенсорних вузлів // Вісник ВПІ. Вип. 3. С. 78–83.
4. Крижановський В. Г., Комаров В. Ф., Сергієнко С. П., Крижановський В. В. Безпека сенсорів в кіберфізичних системах // Вимірювальна та обчислювальна техніка в технологічних процесах: матеріали двадцять першої міжнародної науково-технічної конференції (3-7 червня 2021 р., м. Одеса), Державний університет інтелектуальних технологій і зв'язку. Одеса-Тернопіль: Видавництво "Крок". 2021. С 30–32.
5. Марченко Б.Г., Щербак Л.М. Основи теорії вимірювань//Праці Інституту електродинаміки Національної академії наук України.. Київ, 1999. С. 221-230.
6. Філіпський, Ю. К. Випадкові процеси у радіотехнічних колах. / Одесса: Наука і техніка, 2012. – 175 с.
7. Швець В. А. Випадкові процеси з заданими характеристиками / В. А. Швець, Ю. В. Тимченко, В. Є. Осіпова // III міжнародна наукова-практична конференція "Комплексне забезпечення якості технологічних процесів та систем". Чернігів. 2013, С. 165 – 169.
8. Agilent Open Agilent Spectrum Analysis Basic /, Agilent Technologies. USA. 2006.119 p.
9. D-Robotics DHT11 Humidity & Temperature Sensor / DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. 2010.9 p.
10. Datasheet DHT22 / Maxim Integrated Products / 2019.15 p.
11. Datasheet DS18B20 / Maxim Integrated Products / 2019.20 p.
12. Elias De Coninck, Tim Verbelen, Bert Vankeirsbilck, Steven Bohez, Pieter Simoens, Piet Demeester, and Bart Dhoedt. Distributed neural networks for Internet of Things: the Big-Little approach / Ghent University – iMinds, Department of Information Technology Gaston Crommenlaan. Gent, Belgium. 2020. 5 p.
13. Florian Metzger, Tobias Hossfeld, André Bauer, Samuel Kounev. Modeling of Aggregated IoT Traffic and Its Application to an IoT Cloud / Norwegian University of Science and Technology, Trondheim. Norway. March 2019.15 p.

14. Greengard S. The Internet of Things / The Mit Press Essential Knowledge series.2018.176 p.
15. Jing Zhao1 Fan Zhang1 Chao Zhao1 Gang Wu1 Haitao Wang and Xinyu Cao1. The Properties and Application of Poisson Distributio / China National Institute of Standardization, Beijing, China, IWAACE 2020.5 p.
16. John R. Taylor An introduction to error analysis / University Science Books Sausalito, California, 1985. 323 p.
17. Ke Zhang, Hanbo Ying, Hong-Ning Dai, Lin Li, Yuangyuang Peng, Keyi Guo, Hongfang Yu. Compacting Deep Neural Networks for Internet of Things: Methods and Applications / IEEE Internet of Things Journal. 20 Mar. 2021.25 p.
18. McRoberts M.R Arduino Starter Kit Manual / Earthshine Electronics, 2009. 101 p.
19. Riley W.J. MDEV versus ADEV /.Hamilton Technical Services Beaufort. USA. 2020.20 p.
20. Rajalakshmi Krishnamurthi, Adarsh Kumar, Dhanalekshmi Gopinathan, Anand Nayyar, Basit Qureshi. An Overview of IoT Sensor Data Processing, Fusion, and Analysis Techniques / Department of Computer Science and Engineering: Jaypee Institute of Information Technology. Noida. India. 26 October 2020.23 p.
21. V. Krizhanovski, V. Komarov, S. Serhiienko and V. Kryzhanovskyi, "Application of Allan variance for data control in sensor networks," 2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT), 2021, pp. 88-91.

Додаток А

Ініціалізація.

```
#if TARGET > 0
    STREAM << F("Started ") << date << ENDL;
    // Sensors initialization
    STREAM << F("Sensors: ");
  #endif
 #endif
 //
 dht0.begin();
 //
 sensors0.begin();
 uint8_t temp = sensors0.getResolution();
 if (temp != DS18_0_RES) {
  sensors0.setResolution(DS18_0_RES); // set global sensors0 resolution
  //STREAM << F("DS18(0) resolution updated: ") << temp << ENDL; // get global resolution
 } else {
  //STREAM << F("DS18(0) resolution: ") << temp << ENDL; // get global sensors1 resolution
 }
 //
 sensors1.begin();
 temp = sensors1.getResolution();
 if (temp != DS18_1_RES) {
  sensors1.setResolution(DS18_0_RES); // set global sensors1 resolution
  //STREAM << F("DS18(1) resolution updated: ") << temp << ENDL; // get global resolution
 } else {
  //STREAM << F("DS18(1) resolution: ") << temp << ENDL; // get global resolution
 }
 // sensor chek-up & report
 poll_sensors();
 //
 check_up(0); coma0(); check_up(1); coma0(); check_up(2); coma0(); check_up(3); coma0();
 check_up(4); coma0(); check_up(5); coma0(); check_up(6); coma0(); check_up(7); coma0();
 check_up(8); coma0(); check_up(9); coma0();
 check_up(10);
```

```
//
#if TARGET > 0
  #if TARGET == 3
    STREAM << F("<READY") << EOD << endl;
  #else
    #ifdef CHATTER_LESS
      STREAM << sensors_active << '/' << SENSORS << F(" active") << ENDL;
    #else
      STREAM << F(" (active ") << sensors_active << '/' << SENSORS << ')' << ENDL;
    #endif
  #endif
#endif
```

Підключення карти пам'яті:

```
  // Searching for original task on SD
    tasks = scan_task();
    if (tasks > 0) task = scan_task(task);
    //
    if (task == 255) {
      #if TARGET == 0
        digitalWrite(LED_RED_PIN, HIGH);
      #else
        msg_smthwrong();
      #endif
      return;
    }
uint8_t scan_task(uint8_t for_what = 255) {
    uint8_t res = 0; boolean found = false;
    #if TARGET > 0
      if (for_what == 255) STREAM << F("Search for new task in \"" TASKFILE "\"... ");
    #endif
    #ifdef LOCAL_FILE
      File
    #endif
    xfile = SD.open(TASKFILE); // open the file for reading
    if (xfile) {
```

```
  do {
    boolean r = false;
    boolean n = false;
    String line = "";
    while (xfile.available()) {
      char c = (char)xfile.read();
      switch (c) {
        case '\r': { r = true; break; }
        case '\n': { n = true; break; }
        default  : { line += c; break; }
      }
      if (r && n) break;
    }
    switch (for_what) {
      case 255: {
        if (parse_line(line) == 4) { found = true; res++; }
        break;
      }
      default: {
        if (parse_line(line, res == for_what) == 4) {
          if (res == for_what) found = true;
          else res++;
        }
        break;
      }
    }
  } while (!(found && (res == for_what)) && xfile.available());
  xfile.close(); // close the file
}
#if TARGET == 0
  if (!found && (for_what != 255)) res = 255;
#else
  if (found) {
    if (for_what == 255) { STREAM << res; msg_found(); }
  } else {
```

```
      if (for_what != 255) res = 255;
      else msg_notfound();
     }
   #endif
   return res;
 }
```

Основний цикл:

```
void loop() {
```

підготовка файлів для запису даних нового завдання:

```
   if (new_file) {
    if (new_take) {
     #if TARGET == 0
      digitalWrite(LED_GREEN_PIN, HIGH);
     #else
      digitalWrite(LED_BLUE_PIN, HIGH);
     #endif
     anon();
     if (new_task) {
      #if TARGET == 0
       digitalWrite(LED_RED_PIN, HIGH);
      #else
       uint32_t time = ((uint32_t)tau * probes / 1000) * takes;
       STREAM << ENDL << F("Task\t ") << dec4_form(task+1) << '/' << dec4_form(tasks) <<
F(": ") << task_name << F(", ") << takes << F(" x ") << probes << F(" x ") << tau << F(" [ms] (") <<
time << F(" seconds)") << ENDL;
      #endif
     }
     #if TARGET > 0
      STREAM << ENDL << F("Take\t ") << dec4_form(take+1) << '/' << dec4_form(takes) <<
F(": ") << date << ENDL;
     #endif
    }
    report();
    file_operation(FOP_PREPARE);
    poll_ticker.start(); //poll_ticker.pause();
```

```
  }
```

Перевірка виконання завдання і зчитування нового, якщо є, або зупинка програми:

```
  if (poll_done) {
   if (new_file) poll_ticker.stop();
   if (new_task) {
    if (task == tasks) {
     #if TARGET == 0
      digitalWrite(LED_GREEN_PIN, HIGH);
     #else
      STREAM << F("All tasks completed!") << endl;
     #endif
     sleep();
     return;
    } else {
     task = scan_task(task);
     if (task < tasks) {
      poll_ticker.interval(tau);
     } else {
      #if TARGET == 0
       digitalWrite(LED_RED_PIN, HIGH);
      #else
       msg_smthwrong();
      #endif
      sleep();
      return;
     }
    }
   }
   poll_done = false;
  }
```

Дивлячись на годинник у момент початку вимірювання, програма вигадує назви файлів:

```
 uint8_t file_operation(uint8_t for_what = 255) {
  uint8_t error = 255;
  #if TARGET == 0
```

```
  digitalWrite(LED_GREEN_PIN, HIGH);
 #else
  digitalWrite(LED_BLUE_PIN, HIGH);
 #endif
 if (new_file) {
  #if TARGET == 0
   digitalWrite(LED_BLUE_PIN, HIGH);
  #endif
  strcpy(file_name, Time_form(now, '_').c_str()); strcat(file_name, ".txt");
  strcpy(probe_name, Date_form(now, '\0').c_str());
 }
```

Розподіл файлів по папкам з назвами датчиків:

```
 for (uint8_t sid = 0; sid < SENSORS; sid++) {
  uint8_t id = (uint8_t)pgm_read_byte(&sensors[sid]);
  char kind[10]; strcpy(kind, sensor_kind(id).c_str());
  id = (sid < 10) ? (sid & 0b11) : 0;
  char dir[43];
  strcpy(dir, task_name);          strcat(dir, "/");
  strcat(dir, kind);               strcat(dir, "/");
  strcat(dir, dec2_form(id).c_str()); strcat(dir, "/");
  strcat(dir, probe_name);
  // try open the file (only one file can be open at a time,
  // so you have to close this one before opening another)
  if (new_file) { if (!SD.exists(&dir[0])) SD.mkdir(&dir[0]); }
```