**МIНIСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Західноукраїнський національний університет**

**Факультет комп’ютерних інформаційних технологій**

Кафедра комп’ютерної інженерії

ПОРОХНЯК Давид Романович

**Алгоритми маршрутизації в безпровідних**

**сенсорних мережах систем побутової**

**автоматизації / Routing algorithms in wireless**

**sensor networks of household automation systems**

спеціальність: 123 - Комп’ютерна інженерія

освітньо-професійна програма - Комп’ютерна інженерія

Кваліфікаційна робота

Виконав студент групи КІм- 21

ПОРОХНЯК Давид Романович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Науковий керівник

к.т.н., доцент Мельник Г.М.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кваліфікаційну роботу допущено

до захисту:

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ р.

Завідувач кафедри

**Тернопіль – 2022**

ПЕРЕЛІК СКОРОЧЕНЬ І СПЕЦІАЛЬНИХ ТЕРМІНІВ

БСМ – Бездротова сенсорна мережа;

КМ – Комп’ютерна мережа;

ПЗ – програмне забезпечення;

АМ – алгоритм маршрутизації ;

MTU – максимальна одиниця передачі;

IP – інтернет протокол.

ВСТУП

Актуальність. У наш час бездротові сенсорні мережі є важливою частиною сучасних інформаційних технологій. Вони використовуються в багатьох сучасних системах, в тому числі і безпекових, системах побутової автоматизації і не тільки. Алгоритми маршрутизації дозволяють пришвидшити роботу цих систем та мінімізувати втрати від виходу з ладу певних частин системи, або інших проблем.

Бездротові сенсорні мережі (БСМ) складаються з великої кількості невеликих сенсорних вузлів, які здатні вимірювати певні змінні (наприклад, температуру, тиск повітря,) і надсилають їх своїм сусідам у вигляді цифрового радіосигналу. Мета такої мережі полягає в тому, щоб розмістити численні датчики на великій географічній території та мати можливість зручно зчитувати всі їхні показання з одного місця. Крім того, окремі елементи повинні бути дешевими у виробництві та функціонувати протягом дуже тривалого періоду часу. Це передбачає, що вузли можуть працювати надзвичайно енергоефективно.

Найпростіший спосіб передачі даних до центру керування — коли кожен датчик надсилає записані ним змінні безпосередньо на базову станцію. Однак для цього передача має відбуватися на велику відстань, що, у свою чергу, призводить до дуже високого енергоспоживання і, отже, до короткого терміну служби вузла. Тому мета полягає в тому, щоб радіолінії були якомога коротшими. З цією метою пакети даних у DSN передаються від одного вузла до сусіднього вузла, поки вони нарешті не прибудуть до приймача даних. Це означає, що показання можуть проходити через велику кількість елементів, перш ніж прибути до місця призначення. Оскільки кожен вузол може мати кілька сусідів, необхідно, щоб найкращий вибрано відповідний. Тому окремі вузли повинні мати можливість приймати розумні рішення, щоб знайти вигідний шлях через мережу (маршрутизація).

Прикладом використання сенсорних мереж може бути боротьба з лісовими пожежами. Для цього кілька сенсорних вузлів розподіляються в зоні небезпеки. Кожен модуль вимірює температуру через певні проміжки часу. Коли нагрівається, сенсорний вузол надсилає запит своїм сусідам. Якщо тут також вимірюється підвищена температура, то на базову станцію (пожежну команду) передається попереджувальний сигнал, що містить інформацію про місце пожежі. Запит до сусідів важливий, тому що один датчик може відволіктися через дефект або інші впливи (наприклад, багаття). Він також може надати інформацію про масштаби пожежі.

Метою роботи є побудова бездротової сенсорної мережі та удосконалення алгоритму маршрутизації .

Для досягнення мети роботи слід виконати такі задачі:

* проаналізувати види сенсорних мереж;
* описати апаратну частину мереж;
* проаналізувати види алгоритмів маршрутизації;
* провести моделювання бездротової сенсорної мережі;
* написати програмний код для реалізації алгоритму маршрутизації.

Об’єкт дослідження. Бездротова сенсорна мережа.

Предмет дослідження. Алгоритми маршрутизації в бездротових сенсорних мережах.

Наукова новизна одержаних результатів:

1. Проведено імітаційне моделювання БСМ.
2. Вдосконалено роботу мурашиного алгоритму.

Практична цінність роботи: Розробено алгоритм маршрутизації зі швидкою роботою, який забезпечує високу автономність БСМ

Апробація роботи. Отримані результати опубліковані в межах VI Науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі», опубліковано тези доповіді.

У першому розділі проаналізовані основні поняття на рахунок бездротових сенсорних мереж, розібрана їх апаратна частина, датчики, можливості підключення. Розібрані основні поняття на рахунок алгоритмів маршрутизації, проаналізовані загальні проблеми впровадження бездротових сенсорних мереж у системах автоматизації.

У другому розділі розібрані загальні вимоги до алгоритмів маршрутизації для того, аби можна було точніше обрати підходящий алгоритм для наших задач. Було проведено порівняння їх між собою. Розібрано методи імітаційного моделювання бездротових сенсорних мереж.

У третьому розділі була розглянута загальна архітектура системи. Була проведена реалізація бездротової сенсорної мережі та написаний програмний код для алгоритму маршрутизації, проведено експерементальне дослідження часу автономної роботи бездротової сенсорної мережі.

1 АНАЛІЗ МЕРЕЖЕВИХ ТЕХНОЛОГІЙ

1.1 Бездротові сенсорні мережі

Сьогодні бездротові мережі є важливою частиною телекомунікацій нашого часу. У бездротового зв'язку є ціла низка плюсів: він мобільний, його легко створювати і реструктурувати, він має високу швидкість доступу до інтернету, і його можна підключати до мереж іншого типу – усі ці переваги спонукають все більше людей ставати користувачами бездротових технологій. Цей ринок має дуже багатий асортимент подібних технологій , десятки, а то і сотні фірм пропонують власні рішення

|  |  |  |  |
| --- | --- | --- | --- |
| Бездротова технологія (стандарт) | ZigBee  (IEEE 802.15.4) | Wi-Fi  (IEEE 802.11b) | Bluetooth  (IEEE 802.15.1) |
| Частотний діапазон | 2,4-2,483 ГГц | 2,4-2,483 ГГц | 2,4-2,483 ГГц |
| Пропускна здатність, кбіт/с | 250 | 11000 | 723,1 |
| Розмір стека протоколу, кбайт | 32-64 | Більше 1000 | Більше 250 |
| Час безперервної автономної роботи від батареї, дні | 100-1000 | 0,5-5 | 1-10 |
| Максимальна кількість вузлів в мережі | 65536 | 10 | 7 |
| Діапазон дії, м | 10-100 | 20-300 | 10-100 |
| Галузі застосування | Віддалений моніторинг і управління | Передача мультимедійної інформації | Заміщення дротового з’єднання |

Таблиця 1.1 – Порівняння характеристик технологій: BlueTooth, Wi-Fi і ZigBee

Основними прикладами бездротової передачі даних є наступні технології: Wi-Fi, WiMAX, Bluetooth, EDGE, ZigBee. Яку технологію передачі обрати? В першу чергу це залежить від вимог, які ставлять до конкретних мереж. Часто критеріями виступають такі параметри як пропускна здатність, діапазон частот, можлива кількість вузлів у цій мережі, споживання енергії, тощо. Характеристики різних технологій для подальшого порівняння можна побачити у таблиці1.1.У цій таблиці (1.1) ми можемо побачити, що найбільша кількість вузлів, а ще найбільший час роботи від батарейки є саме у мережі з технологією ZigBee, яка працює по стандарту IEEE 802.15.4. Ця технологія відома як «Бездротові сенсорні мережі (скорочено - БСМ)» (англ. WSN – Wireless Sensor Network).

Сам термін «сенсорна мережа» означає розподілену, а також стійку до зупинки функціонування окремих частин мережу, яка самоорганізовується і складається з багатьох невеликих і недорогих пристроїв із напівпровідників, які обмінюються даними за допомогою бездротового зв'язку, вони можуть довго не обслуговуватись і не вимагають спеціальну установку. В числі перших прототипів сенсорних мереж можна назвати систему SOSUS (Sound Surveillance System, Система спостереження звуку), Яку створювали для знаходження і ідентифікації підводних лодок. Вже потім дану систему перекваліфікували для цивільного використання і вона досі використовується океанографічними і атмосферними службами для моніторгу, різних природніх явищ таких як, скажемо, сейсмічна активність. В часи холодної війни оборонний сектор розробив комплекс протиповітряного захисту континентальної частини США і Канади. Певна особливість цього комплексу як сенсорної мережі стало використання аеростатів як певних точок збору і контролю інформації. А потім цю систему доповнили літаки повітряного попередження і керування AWACS (Airborne Warning and Control System). Дві ці військові системи є прикладом будування сенсорних мереж на їх початковій стадії розвитку, коли ще не йшла мова про збільшення ефективності, їх оптимізацію удешевлення і автономності , в той час людина брала безпосередню участь у їх функціонуванні – саме людина була відповідальна за надійність роботи і подальшого функіонування, аналіз інформації тощо. Активний розвиток технологій бездротових сенсорних мереж почався у 1990-х роках, а на початку 2000-х мікроелектроніка розвинулась ще сильніше, дозволивши виготовляти для цих приладів досить недорогу елементну базу. Технологія бездротових сенсорних мереж сьогодні є єдиною технологією, яка може вирішити задачу контролю і моніторингу, де є залежність від часу роботи датчиків. З'єднані у бездротову сенсорну мережу сенсори (датчики) створюють розподілену територіально систему збирання, обробки і пересилання інформації, яка може самоорганізовуватися. Така мережа має змогу ретранслювати повідомленя по ланцюгу від одного до іншого, а це дозволяє нам організувати передачу інформації у випадку поломки одного з вузлів, використовуючи сусідні вузли, при цьому не втрачати якість. Мережа може визначати вигідний маршрут потоків інформації.

Основне, що відрізняє бездротову сенсорну мережу від звичних телефонних, чи комп’ютерних мереж це відсутність регулярної інфраструктури, яка належить провайдеру, чи оператору. Усі термінали у такій мережі мають змогу працювати не лише як кінцевий пристрій, а й як певний транзитний вузол (рис. 1.1)

В безпровідних сенсорних мережах дані передаються способом перенаправлення інформації від близького вузла, навідміну від традиційних мереж, в яких є фіксована інфраструктура, де дані передаються на базову станцію, чи точку доступу. Усі ці мережі носять назву багатокрокові (multi-hop).



Рисунок 1.1 – Приклад з'єднання сенсорів мережі

Будувати ефективні і надійні сенсорні мережі це не зовсім просте завдання і воно має враховувати різні аспекти.

По-перше, надійність і стійкість при різних несправностях. Ця мережа має не виходити з ладу і повноцінно працювати, надавати потрібні дані навіть тоді, коли один сенсор чи ціла група сенсорів, які є у цій мережі, перестануть працювати. Алгоритми маршрутизації та протоколи керування мережею повинні підтримувати швидку адаптацію до нової фізичної топології.

Друге це масштабованість. Загалом число вузлів в певній мережі може мінятися в залежно від її розвитку. Сьогодні розміри сенсорних мереж можуть бути різними і обмежуватись навіть не сотнями, а тисячами вузлів. Проте занадто сильне збільшення числа сенсорів може призвести до перезавантаження таблиць маршрутів, блокування сенсорів, а це може призвести до виходу із ладу усієї мережі.

Третє це топологія. Її зміна у бездротовій сенсорній мережі може бути викликана підклченням все нових і нових сенсорів в працюючу мережу, поломками сенсорів, які належать мережі, або, якщо сенсори є мобільними.

Не можна забувати і про обмеження бездротового інтерфейсу і апаратного забезпечення. Важлива вимога для широкого і комерційного застосування є невеликий розмір. У разі зменшення габаритів і ваги апаратної частини сенсора можна підвищити кількість можливих додатків до сенсорних мереж. Загалом, сенсори які передають інформацію у мережі можуть користуватися різними бездротовими технологіями з певними характеристиками.

Ну і споживання енергії. У зв’язку із своїми невеликими і компактними габаритами сенсор обладнується невеликим і обмеженим джерелом живлення, тому зменшення кількості енергії, яку сенсор споживає і збільшення ємності джерела живлення, що у свою чергу сильно підвищує автономність сенсору є одним із приоритетних і важливих завдань, яке виконується не лише збільшенням ємності джерела живлення, але й більш ефективним його використанням.

1.2 Апаратне забезпечення мереж

Бездротові сенсорні мережі в даний час є однією з основних технологій, які мають потенціал трансформації майбутнього людської цивілізації. Бездротові сенсорні мережі використовують у різних сферах, таких як промислова, медична, військова, допомога при стихійних лихах тощо. У них є багато додатків, і з кожною програмою з'являються нові протоколи. Отже, огляд існуючих апаратних платформ і протоколів зв'язку необхідний для порівняльного огляду як перший крок у розгортанні бездротових сенсорних мереж для використання в будь-якій області. Ми обговорюємо апаратні платформи, такі як Shimmer, TelosB, Ember та два набори для розробки Texas Instruments. Також широко використовуються протоколи бездротового зв'язку, в яких використовуються механізми перемикання каналів, для поліпшення надійного зв'язку.

Останні досягнення у вбудованих обчислювальних системах призвели до появи бездротових сенсорних мереж, що складаються з невеликих «мотів» на батарейках з обмеженими обчислювальними та радіозв'язковими можливостями. Сенсорні мережі дозволяють глибоко вбудовувати збір даних та обчислення у фізичне середовище. Ця технологія має потенціал вплинути на доставку даних (будь-якого формату) до приймача, який автоматично їх збирає, це повністю інтегрований пристрій, і це працює в режимі реального часу, кореляції з різними обчисленнями та довгостроковими спостереженнями.

Велика кількість вузлів, обмежені енергетичні ресурси, мережа, орієнтована на дані , внутрішньо-мережева обробка – на сьогодні є звичним явищем у літературі. Менше розглядається привласнене поняття надійності:

Датчики дешеві, і їх показання, таким чином, можуть бути гучними, що породжує проблему точності інформації. Також вони не мають надійних протоколів, закладених в них, через що надійність стає критичною в WSN.

Апаратних платформ БСМ є доволі багато, ось наступні з них:

SHIMMER повністю інтегрований з високорівневим прикладним середовищем BioMOBIUS, що дозволяє швидко створювати прототипи. Це бездротова сенсорна платформа для неінвазивних біомедичних досліджень. Це дозволяє швидко протоуватипінг додатків для біомедичних досліджень. Платформа SHIMMER DESIGN складається з плінтуса, який надає такі можливості, як, наприклад, обчислення датчиків, як пасивний датчик вібрації нахилу, датчик PIR використовується як тригер пробудження енергозбереження при наближенні користувача. Зберігання даних, які полегшують запис даних на карту microSD. Зв'язок включає радіо CC2420, Bluetooth і IEEE

802.15.4. І він також має включене з'єднання Daughterboard . SHIMMER також підтримує TinyOS-2.x. SHIMMER - надзвичайно гнучка сенсорна платформа. Він має здатність безперешкодно розширюватися для задоволення різних біомедичних досліджень.

EM250 включає 128 kB вбудованої пам'яті Flash Read Only (ROM). Він також допускає три різні режими роботи. Активна операція дозволить виконати програмний код, як правило, використовуючи 8,5 мА струму. Робота на холостому ходу дозволяє MCU вимикатися до тих пір, поки не відбудеться переривання, в той час як периферійні пристрої та приймач працюватимуть нормально. EM250 також дозволяє виконувати операцію глибокого сну, яка живить MCU та Transceiver, поки зовнішній переривання або таймер не розбудять пристрій. В операції глибокого сну EM250 зазвичай використовує 1,5 мА струму. EM250 має чотири АЦП, з яких два використовуються для використання для фіксації аналогових даних. EM250 також має можливість спілкування через послідовний периферійний інтерфейс (SPI).



Рисунок 1.2 – Ember

Ціна за одиницю TelosB висока, близько 150 доларів США за одиницю, і немає знижки на освітні цілі. Термін служби його потужності становить близько 3–6 місяців залежно від того, як часто сигнал передається назад на сервер, що дещо не вистачає для медичних застосувань. Його радіодеталі не можуть бути покращені (ми не можемо використовувати кращий радіоприймач/антену, щоб досягти більшої відстані). Це бездротовий модуль наднизької потужності , призначений для додатків сенсорних мереж. Платформа mote пропонує оперативну пам'ять на кристалі 10 kB, а також забезпечує IEEE 802.15.4 Chipcon radio with інтегровану бортову антену, що забезпечує дальність до 125 м, структуровану навколо мікроконтролера TI MSP430 . TelosB mote також згадується як Tmote Sky.



Рисунок 1.3 – TelosB

eZ430-RF2500 - це повний USB-інструмент бездротової розробки MSP430, що надає все апаратне та програмне забезпечення для оцінки мікроконтролера MSP430F2274 та бездротового приймача CC2500 2.4-ГГц. Вартість програмного забезпечення для зневаджувача та інструменту розробки становить 29 доларів, а цільової плати, тобто моту - 20 доларів.

Зневаджувач ненав'язливий, що дозволяє користувачеві запускати програму на повній швидкості як з апаратними точками зупинки, так і з одним кроком, не споживаючи при цьому зайвих ресурсів hardware.

Цільова плата eZ430-RF2500T - це готова бездротова система, яка може використовуватися з інтерфейсом налагодження USB, як автономна система з зовнішніми датчиками або без них, або може бути включена в існуючий дизайн [3]. Функції eZ430-RF2500 включають інтерфейс налагодження та програмування USB з встановленням без драйверів та зворотним каналом додатків. Він має 21 доступний штифт розробки. З ним також дається високоінтелектуальний, малопотужний MSP430 MCU з продуктивністю 16 МГц. Два цифрових контакту вводу-виводу загального призначення, підключені до зелених і червоних світлодіодів для візуального зворотного зв'язку. Присутня переривчаста кнопка зворотного зв'язку з користувачами.



Рисунок 1.4 - Цільова плата eZ430-RF2500T

MSP430F2274 має переваги 16-MIPS продуктивності, 10-бітний SAR ADC, два вбудованих операційних підсилювача, таймер Watchdog, два 16-розрядних таймера, модуль USCI, що підтримує UART / LIN, (2) SPI, I2C або IrDA, п'ять режимів малювання низької потужності всього 700 нА в режимі очікування.



Рисунок 1.5 - MSP430F2274

CC2500 має переваги 2,4-ГГц радіочастотного (RF) приймача, програмованої швидкості передачі даних до 500 кбіт / с, низького споживання струму цільова плата eZ430-RF2500 була розроблена для оптимізації під фактори. eZ430-RF2500 може бути використаний як окрема розробка тежl. Цільова плата оснащена MSP430F2274, і більшість її контактів легко доступні.



Рисунок 1.6 CC2500

Пристрої, що підтримують інтерфейс налагодження eZ430-RF USB, можуть використовуватися як стандартний інструмент емуляції спалаху через інтерфейс Spy-Bi-Wire. Інтерфейс налагодження USB eZ430-RF підтримує наступні сімейства MSP430: MSP430F20xx і MSP430F22xx [3].

CC1110 і CC2510 - це пристрої System-on-Chip (SoC) від Texas Instruments, призначені для бездротових додатків з низьким енергоспоживанням . CC1110 працює в неліцензійних діапазонах ISM під 1 ГГц, тоді як CC2510 працює в неліцензійних діапазонах ISM 2.4 ГГц. CC1110 і CC2510 поєднують в собі відмінну продуктивність найсучасніших приймачів CC1101 і CC2500 RF відповідно з галузевим стандартом 8051 MCU, до 32 kB внутрішньо-системних програмована флеш-пам'ять і до 4 кб оперативної пам'яті, і багато інших потужних можливостей. Це забезпечується декількома передовими малопотужними режимами роботи.



Рисунок 1.7 CC1110 і CC2510

1.3 Алгоритми маршрутизації (Протоколи)

Маршрутизація мережі — це процес вибору шляху в одній або кількох мережах. Принципи маршрутизації можна застосовувати до будь-якого типу мережі, від телефонних до громадського транспорту. У мережах з комутацією пакетів, таких як Інтернет, маршрутизація вибирає шляхи для пакетів [Інтернет-протоколу (IP)](https://www.cloudflare.com/learning/ddos/glossary/internet-protocol/) для переміщення від місця відправлення до місця призначення. Ці рішення щодо маршрутизації в Інтернеті приймаються спеціальними елементами мережевого обладнання, які називаються [маршрутизаторами](https://www.cloudflare.com/learning/network-layer/what-is-a-router/) .

Розгляньте зображення нижче. Щоб пакет даних дістався від комп’ютера A до комп’ютера B, чи має він проходити через мережі 1, 3 і 5 чи мережі 2 і 4? Пакет проходитиме коротшим шляхом через мережі 2 і 4, але мережі 1, 3 і 5 можуть пересилати пакети швидше, ніж мережі 2 і 4. Це види вибору, які мережеві маршрутизатори постійно роблять.

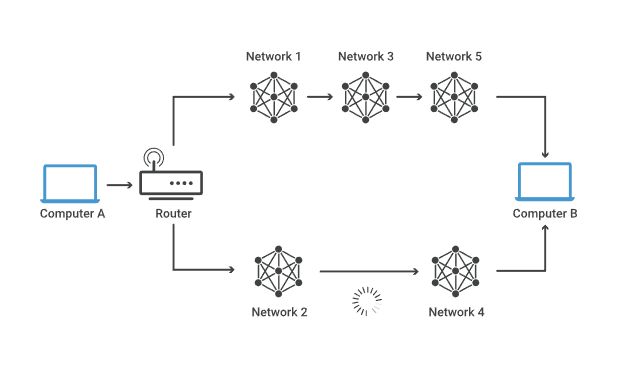


Рисунок 1.7 візуалізація маршрутизації

Маршрутизація підвищує ефективність мережевої комунікації. Збої мережного зв'язку призводять до тривалого очікування на завантаження сторінок веб-сайту для користувачів. Це також може призвести до збою серверів веб-сайтів через те, що вони не можуть обслуговувати велику кількість користувачів. Маршрутизація допомагає мінімізувати збої мережі, керуючи трафіком даних, щоб мережа могла використовувати якнайбільше своєї пропускної спроможності без перевантаження.

Маршрутизатори звертаються до внутрішніх таблиць маршрутизації, щоб приймати рішення про те, як маршрутизувати пакети по мережевих шляхах. Таблиця маршрутизації записує шляхи, якими мають пройти пакети, щоб досягти кожного пункту призначення, за який відповідає маршрутизатор. Подумайте про розклади поїздів, з якими звертаються пасажири, щоб вирішити, на який потяг сісти. Таблиці маршрутизації такі, але для мережевих шляхів, а не для поїздів.

Маршрутизатори працюють таким чином: коли маршрутизатор отримує пакет, він зчитує заголовки\* пакету, щоб побачити його призначення, подібно до того, як провідник може перевірити квитки пасажира, щоб визначити, на якому поїзді йому слід їхати. Потім він визначає, куди спрямувати пакет на основі інформації в його таблицях маршрутизації.

Маршрутизатори роблять це мільйони разів на секунду з мільйонами пакетів. Коли пакет доставляється до місця призначення, він може кілька разів маршрутизуватися різними маршрутизаторами.

Таблиці маршрутизації можуть бути статичними або динамічними. Статичні таблиці маршрутизації не змінюються. Адміністратор мережі вручну налаштовує статичні таблиці маршрутизації. Це, по суті, встановлює маршрути, якими йдуть пакети даних у мережі, якщо адміністратор вручну не оновить таблиці.

Таблиці динамічної маршрутизації оновлюються автоматично. Динамічні маршрутизатори використовують різні протоколи маршрутизації (див. нижче) для визначення найкоротших і найшвидших шляхів. Вони також роблять це визначення на основі того, скільки часу потрібно пакетам, щоб досягти місця призначення — подібно до того, як Google Maps, Waze та інші служби GPS визначають найкращі маршрути руху на основі попередніх показників водіння та поточних умов водіння.

Динамічна маршрутизація вимагає більшої обчислювальної потужності, тому невеликі мережі можуть покладатися на статичну маршрутизацію. Але для середніх і великих мереж динамічна маршрутизація набагато ефективніша.

Протокол маршрутизації — це процес вибору відповідного шляху для переміщення даних від джерела до місця призначення. Процес стикається з кількома труднощами під час вибору маршруту, який залежить від типу мережі, характеристик каналу та показників продуктивності.

Для забезпечення наскрізної доставки пакетів даних між мережами використовуються різні протоколи. Ці протоколи використовують різні типи показників для визначення найкращого шляху для доставки пакета в мережу призначення. Використання маршрутизаторів у міжмережному зв’язку вимагає забезпечення того, щоб маршрутизатори мали необхідну інформацію для визначення шляхів до мережі призначення. Деяка інформація, необхідна маршрутизатору для маршрутизації пакетів між мережами, включає адресу призначення, підключені до нього маршрутизатори, можливі маршрути до інших мереж і найкращий маршрут до мереж. Маршрутизатор визначає інформацію про інші маршрутизатори, особливо про ті, які не підключені безпосередньо до нього, одним із двох способів – статичною маршрутизацією або динамічною маршрутизацією.

При статичній маршрутизації інформацію про сусідні маршрутизатори вводить адміністратор мережі. Зміни в підключенні маршрутизаторів і карті мережевих з’єднань оновлюються адміністратором мережі, і це може бути виснажливим для великих мереж. Щоб пом’якшити цю проблему великих мереж для мережевих адміністраторів, у динамічній маршрутизації використовуються протоколи маршрутизації для визначення найкращого шляху для переміщення пакетів. Протокол маршрутизації визначає, як маршрутизатори передають інформацію про маршрут.

Переміщення або маршрутизація мережевого трафіку даних між мережами здійснюється маршрутизатором. Маршрутизатор визначає оптимальний шлях для передачі інформації між двома мережами за допомогою різних критеріїв або параметрів. Ці критерії, які також називаються протоколами для визначення найкращого шляху, поділяються на три класи: протоколи вектора відстані, протоколи стану зв’язку та гібридні протоколи (поєднання двох вищезгаданих протоколів). Протоколи стану зв’язку просто використовують стан зв’язку, щоб визначити його перевагу над іншими зв’язками. З іншого боку, протоколи векторної маршрутизації відстаней використовують відстань між мережами як показники для визначення оптимального шляху маршрутизації трафіку.

Протоколи векторної маршрутизації відстані використовують кількість переходів для оцінки відстані до мережі. Перехід відноситься до маршрутизатора, і кожен раз, коли пакет проходить через маршрутизатор, це називається стрибком.

У цьому протоколі маршрутизатори періодично інформують своїх сусідніх маршрутизаторів про зміни в топології мережі. Протоколи дистанційно-векторної маршрутизації використовують алгоритм Беллмана-Форда для прийняття рішень щодо маршрутизації. Таким чином, маршрутизатори не мають повного знання про весь шлях, що веде до мережі призначення, натомість вони використовують адресу наступного переходу та вартість досягнення певного вузла з точки зору кількості переходів, щоб визначити оптимальний шлях до мережі. Кожен маршрутизатор повідомляє обчислене значення відстані до інших маршрутизаторів і також отримує те саме. Зміни в топології мережі періодично поширюються шляхом обміну записами таблиці маршрутизації між сусідніми маршрутизаторами, які налаштовані на той самий протокол маршрутизації з вектором відстані.

Цей процес називається «маршрутизація за чутками» через те, що маршрутизатори не можуть визначити достовірність отриманої ними інформації про шлях, а можуть покладатися лише на інформацію, отриману від інших маршрутизаторів. Використовуючи ці оголошення маршрутизації, кожен маршрутизатор заповнює свою таблицю маршрутизації, і лише оновлення таблиці маршрутизації оголошуються в наступному оголошенні, доки таблиця маршрутизації кожного маршрутизатора не зійдеться, коли всі хости в мережі матимуть однакові знання про шляхи маршрутизації. Протоколи маршрутизації, які використовують вектор відстані як показники, включають протокол інформації про маршрутизацію (RIP), протокол інформації про маршрутизацію версії 2 (RIPv2) і протокол маршрутизації внутрішнього шлюзу (IGRP). кожен маршрутизатор заповнює свою таблицю маршрутизації, і лише оновлення таблиці маршрутизації оголошуються в наступному оголошенні, доки таблиця маршрутизації кожного маршрутизатора не зійдеться, коли всі хости в мережі матимуть однакові знання про шляхи маршрутизації. Протоколи маршрутизації, які використовують вектор відстані як показники, включають протокол інформації про маршрутизацію (RIP), протокол інформації про маршрутизацію версії 2 (RIPv2) і протокол маршрутизації внутрішнього шлюзу (IGRP). кожен маршрутизатор заповнює свою таблицю маршрутизації, і лише оновлення таблиці маршрутизації оголошуються в наступному оголошенні, доки таблиця маршрутизації кожного маршрутизатора не зійдеться, коли всі хости в мережі матимуть однакові знання про шляхи маршрутизації. Протоколи маршрутизації, які використовують вектор відстані як показники, включають протокол інформації про маршрутизацію (RIP), протокол інформації про маршрутизацію версії 2 (RIPv2) і протокол маршрутизації внутрішнього шлюзу (IGRP).

Перш ніж заглибитися в розуміння протоколів дистанційної маршрутизації, буде не зайвим поговорити про алгоритм Беллмана-Форда, який у маршрутизації пакетів даних між мережами.

Алгоритм Беллмана-Форда є розподіленим алгоритмом, оскільки він включає кілька вузлів маршрутизатора в автономній системі (AS). Алгоритм використовує наступні кроки для маршрутизації пакетів даних між мережами.

1. Кожен вузол маршрутизатора обчислює відстань до всіх інших вузлів відносно себе та зберігає цю інформацію в таблиці.

2. Кожен вузол маршрутизатора надсилає цю інформацію, що зберігається в таблиці (вкладці маршрутизації), усім сусіднім вузлам.

3. Коли вузол отримує таблицю від сусідніх вузлів, він обчислює найкоротший маршрут до всіх інших вузлів у мережі відносно себе та оновлює власну таблицю цією новою інформацією.

Протокол інформації про маршрутизацію (RIP) версії 1 і 2. RIP був створений і використовувався як частина набору протоколів Xerox Network System (XNS) як вимога для універсального протоколу Xerox (PUP) для можливості маршрутизації. Internet Engineering Task Force) IETF офіційно опублікувала специфікацію для RIP у документі RFC 1058 у червні 1988 року, і в цьому документі визначено те, що називається RIP версії 1 (RIPv1).

RIP як протокол дистанційної векторної маршрутизації використовує кількість переходів як метрику для маршрутизації пакетів даних від джерела до пункту призначення. Щоб запобігти петлям маршрутизації, на шляху між джерелом і кінцевим маршрутизатором дозволена максимальна кількість стрибків 15. Це обмеження накладає обмеження на розмір будь-якої мережі, що використовує RIP як протокол маршрутизації. Щоб маршрутизатори в мережі знали топологію мережі, між маршрутизаторами в мережі з підтримкою RIP періодично з інтервалом у 30 секунд обмінюються оновленнями таблиць маршрутизації.

RIP версії 2 було розроблено для усунення недоліків RIPv1. Стандартна версія була випущена в 1998 році з додаванням підтримки VLSM і аутентифікації MD5. Підтримується безкласова міждомена маршрутизація (CIDR), оскільки вона містить інформацію про підмережі в оновленнях RIP. Щоб забезпечити зворотну сумісність із першою версією, у RIPv2 збережено обмеження у 15 переходів. Зменшення навантаження на хости, які не беруть участь у маршрутизації, досягається реалізацією групової передачі на суміжні маршрутизатори замість широкомовної передачі, яка використовується в RIPv1.

IGRP є другим протоколом дистанційного вектора першого покоління, який використовує ті самі стратегії конвергенції, що й RIP. Він має велику перевагу над RIP завдяки своїй здатності розрізняти шляхи до мереж, використовуючи затримку, пропускну здатність, надійність, навантаження та максимальну одиницю передачі (MTU), які перетворюються на скалярну величину для вираження бажаності шляхів. IGRP не підтримується маршрутизаторами інших постачальників, окрім маршрутизаторів Cisco, оскільки це власний протокол Cisco.

На відміну від протоколів векторної маршрутизації на відстані, кожен маршрутизатор, який використовує протокол маршрутизації за станом зв’язку, має інформацію про повну топологію мережі. Для побудови таблиці маршрутизації маршрутизатори використовують інформацію про тип з’єднання, метрику вартості та стан з’єднання, чи працює чи не працює. Використовуючи локальну інформацію про топологію, кожен маршрутизатор самостійно обчислює для кожного можливого призначення з нього найкращий наступний маршрутизатор з використанням перелічених параметрів. Ці значення зберігаються в таблиці маршрутизації маршрутизатора та використовуються для прийняття рішень щодо маршрутизації. Потім маршрутизатор створює дерево найкоротшого шляху за допомогою алгоритму Дейкстри. Дерево — це граф вузлів і зв’язків між вузлами з певним вузлом, який називається кореневим вузлом, з якого можна дістатися до всіх інших вузлів єдиним маршрутом. Дейкстра s Алгоритм ділить вузли на два набори - попередній і постійний. Роблячи сусідів поточного вузла попередніми, алгоритм перевіряє його на відповідність критеріям наступності; якщо він відповідає критеріям, він стає постійним.

Єдина інформація, яка спільно використовується між маршрутизаторами в протоколі стану зв’язку, необхідна для побудови карти взаємозв’язку маршрутизаторів у мережі, і є контрастом до протоколів маршрутизації з вектором відстані, де кожен маршрутизатор ділиться своєю таблицею маршрутизації з сусідніми маршрутизаторами. Приклади цього протоколу включають протокол Open Shortest Path First (OSPF) і Intermediate System to Intermediate System (IS-IS).

У OSPF, коли хост отримує оновлення про зміну в мережі або таблиці маршрутизації, він передає ці зміни всім іншим хостам у мережі, щоб переконатися, що всі вони мають однакову інформацію про маршрутизацію. Цей метод надсилання лише оновлень у мережу споживає менше системних ресурсів порівняно з RIP, який надсилає всю таблицю маршрутизації кожні 30 секунд.

Оцінка шляхів у OSPF виконується не просто за допомогою підрахунку переходів, а скоріше на основі «станів посилань», які включають додаткову інформацію про мережу. Адміністратор також може призначити метрики вартості для майбутніх маршрутизаторів, щоб віддати перевагу деяким шляхам. OSPF підтримує маску підмережі змінної довжини (VLSM) для підмереж.

У мережі IS-IS є кінцеві системи, проміжні системи, області та домени. Кінцеві системи — це пристрої користувача, проміжні системи — маршрутизатори, а маршрутизатори організовані в локальні групи, які називаються «областями», а кілька областей об’єднані в «домен». IS-IS відрізняється від OSPF своєю масштабованістю для підтримки великих мереж. За тих самих ресурсів IS-IS може підтримувати більше маршрутизаторів у зоні, ніж OSPF. Хоча основні концепції обміну інформацією про топологію подібні, IS-IS також відрізняється від OSPF методами, за допомогою яких він надійно передає топологію та інформацію про зміну топології через мережу.

Протоколи гібридної маршрутизації мають спільні риси як протоколів маршрутизації з вектором відстані, так і протоколів маршрутизації за станом зв’язку. Прикладом цього є Enhanced Interior Gateway Routing Protocol (EIGRP).

EIGRP також є власним протоколом Cisco, який пропонує покращення порівняно з IGRP. Він був розроблений для використання найкращих функцій протоколів маршрутизації за вектором відстані та за станом зв’язку. EIGRP підтримує як безкласові, так і класові IP-адреси на відміну від IGRP, а також інших мережевих протоколів. Це дає змогу мережам, що використовують Classless Inter – Domain Routing, IP-адреси, сумісні з CIDR, і VLSM використовувати протокол дистанційної векторної маршрутизації Cisco.

1.4 Проблеми впровадження БСМ в системах автоматизації

Завдання проектування протоколів маршрутизації для WSN є досить складним через численні характеристики, які відрізняють їх від бездротових мереж без інфраструктури. Кілька типів проблем маршрутизації, пов’язаних із бездротовими сенсорними мережами. Нижче наведено деякі важливі проблеми:

* Для великої кількості сенсорних вузлів практично важко виділити універсальну схему ідентифікаторів. Отже, бездротові датчики не вміють використовувати класичні протоколи на основі IP.
* Створений трафік даних у більшості випадків має значну надмірність. Оскільки багато сенсорних вузлів можуть генерувати однакові дані під час зондування. Отже, дуже важливо використовувати таку надмірність за допомогою протоколів маршрутизації та максимально ефективно використовувати доступну смугу пропускання та енергію.
* Більше того, бездротові порошинки жорстко обмежені щодо енергії передачі, пропускної здатності, ємності та зберігання та бортової енергії. Через такі відмінності було спроектовано ряд нових протоколів маршрутизації, щоб впоратися з цими проблемами маршрутизації в бездротових сенсорних мережах.

1.5 Висновки до розділу.

У цьому розділі було розглянуто основні питання і означення бездротових сенсорних мереж, їх апаратне забезпечення, види датчиків, сенсорів та інших складових БСМ.

2. АЛГОРИТМИ МАРШРУТИЗАЦІЇ В БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖАХ

2.1 Вимоги до алгоритмів маршрутизації.

Для цього проекту будуть присутні певні вимоги як до апаратного, так і до програмного забезпечення. Оскільки важливу роль відіграє саме алгоритм маршрутизації, то нам потрібно деталізувати основні вимоги до нього, саме це дозволить нам у майбутньому, порівнявши показники, характеристики і можливості декількох алгоритмів маршрутизації, обрати найбільш підходящий для наших задач, тим самим підвищивши ефективність усієї системи в цілому.

Алгоритми, покладені основою формування та оновлення таблиці маршрутизації, називають алгоритмами маршрутизації. Відповідно до даних алгоритмів і визначаються найкращі маршрути до можливих пунктів призначення. Алгоритми передачі пакетів оптимальними шляхами, що вибираються з таблиці маршрутизації, називають алгоритмами комутації *.*

З наведеного вище опису стає зрозуміло, що алгоритми комутації, що задають порядок транспортування пакетів через мережу при оптимальних відомих маршрутах, є досить простими. Складними та найважливішими є алгоритми маршрутизації, які й становлять основу протоколів маршрутизації. До даних алгоритмів висувають такі функціональні вимоги:

- за оптимальністю визначених маршрутів *-* здатність визначати найкращий маршрут залежно від заданих показників та їх вагових коефіцієнтів;

- за гнучкістю *-* здатність швидко і точно адаптуватися до змін структури та умов функціонування мережі;

- за збіжністю *-* здатність досягати швидкої угоди між усіма маршрутизаторами мережі оптимальними маршрутами.

У протоколах маршрутизації показник оптимальності маршруту часто називають метрикою. Оптимальним вважається найкоротший шлях. При цьому метрика, тобто міра довжини шляху задається певною формулою, як змінні якої можуть виступати будь-які характеристики маршруту, наприклад, загальна кількість транзитних маршрутизаторів і сумарний час пересилання.

Вимоги до алгоритмів маршрутизації за гнучкістю та збіжністю взаємопов'язані один з одним. Коли в мережі відбуваються будь-які зміни, що впливають на вибір оптимальних маршрутів, наприклад, перевантаження будь-якої ділянки мережі або поява нового каналу зв'язку, маршрутизатори, які дізналися першими про ці зміни, повинні перевизначити свої оптимальні маршрути, адаптуючись до змін. Крім того, вони повинні надіслати повідомлення про зміни іншим маршрутизаторам. Ці повідомлення пронизують мережі, стимулюючи перерахунок оптимальних маршрутів. Зрештою всі маршрутизатори повинні дійти спільної угоди щодо оптимальних маршрутів.

Алгоритми маршрутизації, що не мають високої гнучкості та швидкої збіжності, призводять до утворення петель маршрутизації і навіть виходів мережі з ладу.

Алгоритм повинен бути простим і не перегруженим зайвими нюансами, які не будуть позитивно впливати на його ефективність, він повинен максимально оптимально підбирати маршрути, бути стійким до різних помилок, бути доволі гнучким і зрозумілим.

Якщо коротко, то сновними вимогами до алгоритму маршрутизації є:

* + Оптимальність вибору маршруту;
  + Простота реалізації;
  + Стійкість;
  + Швидка збіжність;
  + Гнучкість реалізації.

Оптимальність вибору маршруту є основним параметром алгоритму, що вимагає пояснень.

Алгоритми маршрутизації повинні бути прості в реалізації та використовувати якнайменше ресурсів

Алгоритми повинні бути стійкими до відмов обладнання на спочатку обраному маршруті, високих навантажень та помилок у побудові мережі.

Східність – це процес узгодження між маршрутизаторами інформації про топологію мережі. Якщо певна подія в мережі призводить до того, що деякі маршрути стають недоступними або виникають нові маршрути, маршрутизатори розсилають повідомлення про це один одному по всій мережі. Після отримання цих повідомлень маршрутизатори виробляють перепризначення оптимальних маршрутів, що у свою чергу може породити новий потік повідомлень. Цей процес повинен завершитися, причому досить швидко, інакше в мережевій топології можуть з'явитися петлі, або взагалі може перестати функціонувати мережа. Алгоритми маршрутизації повинні швидко та правильно враховувати зміни у стані мережі (наприклад, відмова вузла чи сегмента мережі).

2.2 Огляд і порівняння алгоритмів маршрутизації

Перші дослідження сенсорних мереж почалися приблизно в 1980 році. Дослідження в той час були суто військовими, але незабаром були передані поширено на цивільні території. Через постійно зростаючу щільність упаковки електронних компонентів дослідження пережили значний підйом у 1990-х роках. За останні роки вже було розроблено декілька стратегій маршрутизації для сенсорних мереж. Однак значна частина базується на алгоритмах, створених для Інтернету, і тому не відповідає особливим потребам DSN. Основною проблемою тут є високе енергоспоживання, викликане надсиланням і отриманням численних керуючих даних (без даних користувача), які використовуються для організації мережі.

Протокол маршрутизації — це процес вибору відповідного шляху для переміщення даних від джерела до місця призначення. Процес стикається з кількома труднощами під час вибору маршруту, який залежить від типу мережі, характеристик каналу та показників продуктивності.

Загалом, маршрутизацію в БСМ можна розділити на плоску маршрутизацію, маршрутизацію на основі ієрархії та маршрутизацію на основі розташування залежно від структури мережі. У плоскій маршрутизації всім вузлам зазвичай призначаються однакові ролі або функціональні можливості. Однак у ієрархічній маршрутизації вузли відіграватимуть різні ролі в мережі. Щоб керівники кластерів могли виконувати певне агрегування та скорочення даних з метою економії енергії. У маршрутизації на основі розташування позиції вузлів датчиків використовуються для маршрутизації даних у мережі. Протокол маршрутизації вважається адаптивним, якщо певні системні параметри можна контролювати для адаптації до поточних умов мережі та доступних рівні енергії.

Крім того, ці протоколи можна класифікувати на методи маршрутизації на основі багатошляхового доступу, на основі запитів, на основі узгодження, на основі QoS або когерентної маршрутизації. Протоколи багатошляхової маршрутизації: використовується більше ніж один шлях для продуктивності, особливо на рівні відмовостійкості. Маршрутизація на основі запиту: конкретний запит поширюється між вузлами, які відповідають відповідним чином. Протоколи маршрутизації на основі узгодження: використовує дескриптори даних для придушення повторюваних даних для надсилання на наступний датчик.

 Маршрутизація на основі QoS: містить певні характеристики (пропускна здатність, енергія тощо) над якістю. Узгоджена обробка маршрутизації: виконувати обробку в мережі. Протоколи маршрутизації можна класифікувати на три категорії, а саме: проактивні, реактивні, і гібридні протоколи в залежності від того, як джерело знаходить маршрут до місця призначення. Проактивні протоколи: усі маршрути обчислюються до того, як вони дійсно знадобляться, таблиця маршрутизації, що описує всі шляхи вузлів, підтримується на кожному вузлі, тому називається протоколом маршрутизації, керованим таблицею.

 Реактивні протоколи: маршрути обчислюються на вимогу, також називаються протоколом маршрутизації на вимогу. Гібридні протоколи використовують комбінацію проактивних і реактивних протоколів. Коли сенсорні вузли є статичними, краще мати керовані таблицею протоколи маршрутизації, а не використовувати реактивні протоколи. також називається протоколом маршрутизації за вимогою. Гібридні протоколи використовують комбінацію проактивних і реактивних протоколів. Коли сенсорні вузли є статичними, краще мати керовані таблицею протоколи маршрутизації, а не використовувати реактивні протоколи. також називається протоколом маршрутизації за вимогою. Гібридні протоколи використовують комбінацію проактивних і реактивних протоколів. Коли сенсорні вузли є статичними, краще мати керовані таблицею протоколи маршрутизації, а не використовувати реактивні протоколи.

У плоских мережах кожен вузол, як правило, відіграє однакову роль [2], а сенсорні вузли співпрацюють разом, щоб виконувати завдання зондування. Через велику кількість таких вузлів неможливо призначити глобальний ідентифікатор кожному вузлу, спробувати знайти якісні маршрути від вихідних вузлів до вузлів-приймачів за допомогою певної форми затоплення. Оскільки затоплення є дуже дорогою операцією [4] в мережах з браком ресурсів, розумні алгоритми маршрутизації обмежують затоплення локалізованими регіонами. Деякі алгоритми використовують імовірнісні методи, засновані на певних евристиках для встановлення шляхів маршрутизації.

Це міркування призвело до орієнтованої на дані маршрутизації, коли BS надсилає запити до певних регіонів і чекає даних від датчиків, розташованих у вибраних регіонах. Оскільки дані запитуються через запити, такі протоколи можна класифікувати як керовані запитами протоколи. Протоколи, керовані даними, припускають, що існує окрема фаза розповсюдження запиту, за допомогою якої деякі вузли датчиків розуміють, що їхні дані мають бути надіслані до приймача. Ця фаза зазвичай також відповідає за налаштування маршрутів. Протоколи підпадають під плоску маршрутизацію SPIN, спрямовану дифузію, ACQUIRE.

SPIN (Sensor Protocols for Information via Negotiation): SPIN — це протокол, який транслює всю інформацію кожному вузлу в мережі. Кожен вузол має подібні дані з сусіднім вузлом. Цей протокол поширює інформацію на всі вузли, коли користувачеві не потрібен обмін даними між вузлами. SPIN — це 3-етапний протокол. Він використовує три повідомлення, тобто ADV, REQ і DATA. ADV рекламує нові дані, REQ – це запит даних, а DATA – саме повідомлення. Коли вузол хоче поділитися даними, він транслює повідомлення ADV, що містить дані. Якщо сусідній вузол зацікавлений в отриманні даних, він надсилає повідомлення REQ назад до вузла для передачі даних, а ДАНІ надсилаються до вузла. Запущені вузли.

SPIN призначає назву високого рівня, щоб повністю описати зібрані дані (які називаються метаданими) і виконує узгодження метаданих перед передачею будь-яких даних. Це гарантує відсутність зайвих даних, які надсилаються по всій мережі. Потім сусідні вузли повторюють цей процес зі своїми сусідами, і вся мережа датчиків отримає копію даних.

Сімейство протоколів SPIN включає багато протоколів. Основні два протоколи називаються SPIN-1 і SPIN-2 [9], які включають узгодження перед передачею даних, щоб гарантувати, що буде передана лише корисна інформація. Крім того, кожен вузол має власний менеджер ресурсів, який відстежує споживання ресурсів і опитується вузлами перед передачею даних.

Сімейство протоколів SPIN розроблено на основі двох основних принципах:

* Сенсорні вузли працюють ефективніше та зберігають енергію, надсилаючи дані, які описують дані датчика, замість надсилання всіх даних; наприклад, вузли зображення та датчики повинні стежити за змінами своїх енергетичних ресурсів.
* Звичайні протоколи, як-от протоколи маршрутизації на основі затоплення або пліток, витрачають енергію та пропускну здатність під час надсилання додаткових і непотрібних копій даних датчиками, що охоплюють зони, що перекриваються.

Однією з переваг SPIN є те, що топологічні зміни локалізовані, оскільки кожен вузол повинен знати лише своїх сусідів з одним переходом. SPIN забезпечує більшу економію енергії, ніж затоплення, а узгодження метаданих майже вдвічі зменшує надлишкові дані. Однак механізм оголошення даних SPIN не може гарантувати доставку даних, тобто він має перешкоди Resource Blindness, Emplosion, Overlap.

Маршрутизація, спрямована на дифузію, орієнтована на дані. Основною функцією data-centric є об’єднання даних із різних джерел і вкорінення шляхом економії енергії, усунення надмірності, збільшення терміну служби, мінімізації кількості передач. На початку приймач вказує низьку швидкість передачі даних для вхідних подій. Після цього приймач може посилити певний датчик для надсилання подій з вищою швидкістю передачі даних. Якщо сусідній датчик отримує це повідомлення та виявляє, що зацікавлений відправник має вищу швидкість передачі даних, ніж раніше, і ця швидкість передачі даних вища за існуючий градієнт.

Робота спрямованого поширення включає: передачу інтересів, створення градієнтів і розповсюдження даних. Коли інтереси відповідають градієнтам, Шляхи потоку інформації формуються з кількох шляхів, а потім найкращі шляхи посилюються, щоб запобігти подальшому затопленню згідно з локальним правилом. Щоб зменшити витрати на зв'язок, дані агрегуються в дорозі. Мета полягає в тому, щоб знайти хороше дерево агрегації, яке отримує дані від вихідних вузлів до BS. Усі сенсорні вузли в мережі на основі спрямованої дифузії розраховані на додатки, що дозволяє дифузії досягати енергозбереження шляхом вибору емпірично хороших шляхів, а також кешування й обробки даних у мережі. Кешування може підвищити ефективність, надійність і масштабованість координації між сенсорними вузлами, що є суттю парадигми дифузії даних. Іншим застосуванням спрямованої дифузії є Мета полягає в тому, щоб знайти хороше дерево агрегації, яке отримує дані від вихідних вузлів до BS. Усі сенсорні вузли в мережі на основі спрямованої дифузії розраховані на додатки, що дозволяє дифузії досягати енергозбереження шляхом вибору емпірично хороших шляхів, а також кешування й обробки даних у мережі. Кешування може підвищити ефективність, надійність і масштабованість координації між сенсорними вузлами, що є суттю парадигми дифузії даних.Мета полягає в тому, щоб знайти хороше дерево агрегації, яке отримує дані від вихідних вузлів до BS. Усі сенсорні вузли в мережі на основі спрямованої дифузії розраховані на додатки, що дозволяє дифузії досягати енергозбереження шляхом вибору емпірично хороших шляхів, а також кешування й обробки даних у мережі. Кешування може підвищити ефективність, надійність і масштабованість координації між сенсорними вузлами, що є суттю парадигми дифузії даних. Іншим застосуванням спрямованої дифузії є надійність і масштабованість координації між сенсорними вузлами, що є суттю парадигми дифузії даних. Іншим застосуванням спрямованої дифузії є надійність і масштабованість координації між сенсорними вузлами, що є суттю парадигми дифузії даних. Іншим застосуванням спрямованої дифузії є

спонтанно поширюють важливу подію на деякі ділянки сенсорної мережі.

Спрямована дифузія відрізняється від SPIN у двох аспектах: спрямована дифузія викликає проблеми із запитами даних на вимогу, оскільки BS надсилає запити до вузлів датчиків, переповнюючи деякі завдання. Однак у SPIN датчики повідомляють про доступність даних, що дозволяє зацікавленим вузлам запитувати ці дані. І весь зв’язок у спрямованій дифузії є сусід-сусід, причому кожен вузол має можливість виконувати агрегацію даних і кешування. На відміну від SPIN, немає необхідності підтримувати глобальну мережеву топологію при спрямованій дифузії.

Однак спрямована дифузія не може бути застосована до програм (наприклад, моніторинг навколишнього середовища), які вимагають безперервної доставки даних до BS [13]. Це пояснюється тим, що модель даних на основі запиту може не допомогти в цьому відношенні. Крім того, зіставлення даних із запитами може вимагати додаткових витрат на вузлах датчиків.

Ієрархічна маршрутизація також називається кластерною маршрутизацією. Основною ідеєю розробки протоколу маршрутизації на основі кластера є зменшення мережевого трафіку до приймача. Основною метою ієрархічної маршрутизації є мінімізація енергоспоживання сенсорних вузлів. У якому вузли з більшою енергією можуть використовуватися для обробки та надсилання інформації, а вузли з низькою енергією можуть використовуватися для виконання завдань зондування. Лише низькоенергетичні вузли беруть участь у створенні мережевого шляху. Ієрархічна маршрутизація — це механізм дворівневої маршрутизації, де один рівень використовується для вибору голів кластера, а інший — для маршрутизації. Ієрархічні протоколи: LEACH, PEGASIS, TEEN і APTEEN

LEACH (ієрархія адаптивної кластеризації з низьким енергоспоживанням): Він випадковим чином вибирає кілька вузлів датчиків як головки кластера та обертається, рівномірно розподіляючи енергію між датчиками в мережі. Головний вузол кластера стискає дані, які надходять від вузлів, які належать до відповідного кластера, і надсилає агрегований пакет на базову станцію, щоб зменшити обсяг переданої інформації. LEACH використовує TDMA/CDMA MAC для зменшення внутрішньокластерних і міжкластерних колізій. Якщо є потреба в постійному моніторингу сенсорною мережею, цей протокол є найбільш доцільним.

Щоб збалансувати розсіювання енергії вузлів, головки кластерів змінюються випадковим чином з часом. Рішення приймається шляхом вибору вузлом випадкового числа від 0 до 1. Для поточного раунду вузол стає головою кластера, якщо число менше наступних порогових значень:-

Де P бажаний відсоток голів кластера, r - поточний раунд; G — це набір вузлів, які не мають голів кластера в 1/p раундах . LEACH виконує два завдання, тобто фазу налаштування та фазу стабільного стану. На етапі налаштування вибираються головки кластера. У фазі сталого стану відбувається передача даних на базову станцію. Щоб мінімізувати накладні витрати, тривалість фази сталого стану є більшою, ніж тривалість фази налаштування.

PEGASIS (енергоефективний збір у сенсорних інформаційних системах):

PEGASIS — це вдосконалення в порівнянні з протоколом LEACH, це майже оптимальний протокол на основі ланцюжка [4]. Новий раунд розпочнеться, коли завершиться раунд усіх вузлів, які спілкуються з базовою станцією. Це також включає фактор, що потужність, необхідна для передачі за раунд, зменшується.

Використання техніки спільної роботи збільшує тривалість життя кожного вузла, таким чином, тривалість життя мережі буде збільшена.

Щоб зменшити споживання пропускної здатності під час зв’язку, дозвольте лише локальну координацію між вузлами, розташованими близько один до одного, і по черзі спілкуйтеся з базовою станцією.

TEEN & APTEEN (Енергоефективна сенсорна мережа, чутлива до порогу та адаптивна періодична енергоефективна мережа датчиків, чутлива до порогу)

TEEN чуйно реагує на фізичні коливання [5], такі як температура, тиск тощо. У цьому сенсорний вузол постійно сприймає середовище, але фактична передача даних здійснюється рідше. Вузли, які знаходяться ближче один до одного, утворюють кластери, і цей процес безперервний у мережі, доки не буде досягнуто вузол-приймач. Вузли не можуть спілкуватися один з одним, якщо порогове значення не отримано та користувач не отримує жодних даних. Головні вузли кластера транслюють два пороги у своєму кластері, тобто ЖОРСТКИЙ ПОРІГ і М’ЯКИЙ ПОРІГ. Жорстке порогове значення — це абсолютне значення, за яким вузол сприймає це значення та перемикається на свій передавач, коли вузол виявляє це значення та повідомляє керівнику кластера. Цей поріг використовується для зменшення кількості передач, дозволяючи вузлам передавати лише тоді, коли

відчутий атрибут знаходиться в діапазоні інтересів. А м’який поріг — це невелика зміна значення відчутного атрибута, яка змушує вузол увімкнути передавач і передавати.

APTEEN — це вдосконалення TEEN, яке змінює порогове значення або періодичність протоколу TEEN відповідно до типу програми та умов або потреб користувачів. У протоколі APTEEN головка кластера транслює передачу додатково з пороговими значеннями, як у TEEN. Якщо вузол датчика не може надіслати дані за межі часу підрахунку, тоді використовується схема TDMA

Використовується, і кожному вузлу призначається слот передачі.

У LEACH спостерігається дуже низьке енергоспоживання завдяки розподіленій мережі, тоді як у PEGASIS потрібно більше енергії, оскільки один вузол створений як головка кластера, яка отримує дані від інших датчиків. У TEEN споживання енергії більше, ніж у PEGASIS, завдяки великій мережі, яку можна подолати в APTEEN. LEACH дає найкоротший шлях, тоді як PEGASIS вибирає маршрут жадібним методом, але TEEN і APTEEN дають найкращий вибір маршруту. У великих регіонах LEACH не застосовується до мереж через динамічну кластеризацію, але може бути застосований у випадку PEGASIS, оскільки відстань передачі є зменшений. І в TEEN, і в APTEEN є велика затримка в передачі повідомлення через мережу.

Протоколи маршрутизації на основі розташування використовують інформацію про місцезнаходження для визначення маршруту. У цьому випадку вузли оснащені GPS і розкидані в певній мережі. Положення вузлів можна визначити за допомогою GPS. На основі рівня вхідного сигналу можна оцінити відстань між сусідніми вузлами. Коли відстань між будь-якими двома вузлами в мережі визначається за допомогою потужності сигналу, ми можемо знати про координати за допомогою обміну інформацією або даними з сусідніми вузлами. Протоколи підпадають під маршрутизацію на основі розташування: GAF, GEAR, GOAFR, MECN і SMECN.

GAF (географічна адаптивна точність):

GAF — це енергоінформований алгоритм, розроблений для спеціальних мереж, який також може застосовуватися до сенсорних мереж. У цьому алгоритмі по-перше, мережева область ділиться на фіксовану кількість зон і формує віртуальну сітку. У кожній зоні вузли відіграють різні ролі, співпрацюючи один з одним. Коли сенсорний вузол переходить в сплячий режим для економії енергії, він вимикає радіо. У стані виявлення датчик обмінюється повідомленнями про відкриття з інформацією про інші датчики в сітці. В активному стані датчик безперервно надсилає свої повідомлення, щоб повідомити еквівалентні датчики про свій стан.

GEAR — це енергоефективний протокол маршрутизації, запропонований для маршрутизації запитів до цільових регіонів у полі датчика. У GEAR передбачається, що датчики мають апаратне забезпечення локалізації, оснащене пристроєм GPS, щоб знати їхнє поточне положення. GEAR використовує механізм енергоінформації, який базується на географічній інформації для вибору датчиків для маршрутизації пакету

до пункту призначення. Кожен вузол зберігає приблизну вартість і вартість навчання досягнення пункту призначення через сусідів. Розрахункова вартість – це комбінація відстані до місця призначення та залишкової енергії. Коли вузол не має ближчого сусіда до цілі, виникає діра. Якщо отвори немає, то розрахункова вартість дорівнює вивченій вартості.

GOAFR (The Greedy Other Adaptive Face Routing):

Було запропоновано геометричний спеціальний алгоритм маршрутизації, що поєднує жадібну та гранну маршрутизацію. Жадібний алгоритм GOAFR завжди вибирає сусіда, найближчого до вузла, як наступний вузол для маршрутизації. Однак його можна легко застрягти на якомусь локальному мінімумі, тобто жоден сусід не знаходиться ближче до вузла, ніж поточний вузол. Інша маршрутизація обличчям (OFR) є варіантом Face Routing (FR). Алгоритм Face Routing (FR) є першим, який гарантує успіх, якщо джерело та адресат підключені. Однак у найгіршому випадку вартість FR пропорційна розміру мережі з точки зору кількості вузлів. Першим алгоритмом, який може конкурувати з найкращим маршрутом у гіршому випадку, є алгоритм адаптивної маршрутизації обличчя (AFR).

Крім того, за допомогою аргументу нижньої межі показано, що AFR є асимптотично оптимальним у найгіршому випадку. Але AFR не є середнім ефективним.

Мета полягає в тому, щоб знайти найкращий вузол на межі, тобто найближчий вузол до місця призначення за допомогою геометричних площин. Після завершення алгоритм повертається до найкращого вузла на межі. Було показано, що алгоритм GOAFR може досягти оптимальності як у найгіршому випадку, так і в середньому випадку. Ґрунтуючись на результатах моделювання GOAFR, існує кілька способів подальшого покращення середньої продуктивності випадку.

MECN встановлює та підтримує мережу з мінімальним енергоспоживанням, використовуючи GPS з низьким енергоспоживанням для бездротової мережі. Він базується на двох фазах. По-перше, він займає положення двовимірної площини та будує огороджений (розріджений) граф, який складається з усіх огороджень у графі від кожного вузла передачі. По-друге, знаходить оптимальні зв’язки на графі корпусу. Він використовує розподілений найкоротший алгоритм Форда Беллмана з енергоспоживанням як метрикою вартості. SMECN (комунікаційна мережа малої мінімальної енергії) є розширенням MECN. У MECN неможливо, щоб кожен вузол міг передавати до кожного іншого вузла.

Послідовний вектор відстані пункту призначення (DSDV). Як випливає з назви, це вектор відстані.

Протокол маршрутизації. Основна ідея полягає в тому, що кожен вузол знає структуру всієї мережі. Щоб досягти цього, кожен учасник визначає своїх прямих сусідів і призначає їм значення (вартість) залежно від того, наскільки дешево вони можуть бути досягнуті. На наступному кроці щойно створена таблиця надсилається всім сусідам. Отримані таблиці тепер порівняємо з вашими власними. Якщо є кращі шляхи до місця призначення, власна таблиця оновлюється та надсилається знову.

Переваги:

* Маршрутизація без петель
* Після етапу організації, описаного вище, контрольні дані майже не

надсилаються

Недоліки:

На етапі організації мережа стикається з дуже великою кількістю контрольних даних. Це призводить до великого навантаження на процесор і трансивер і, таким чином, значно збільшує енергоспоживання

* Великі вимоги до пам'яті, оскільки всі маршрути повинні бути збережені в таблиці
* Етап організації потрібно повторювати кожного разу, коли сенсорний вузол виходить з ладу або додається.

На відміну від DSDV, тут немає організаційного етапу. Окремі вузли не обов'язково повинні знати всі маршрути, але вони шукаються лише за потреби (на вимогу). Коли вузол хоче надіслати дані, він спочатку дивиться у свій кеш маршрутів, щоб побачити, чи доступний шлях. Якщо жодного не знайдено, вузол починає заповнювати всю мережу запитом. Кожен учасник, який пересилає пакет, додає свою адресу до заголовка. Якщо запит нарешті досягає місця призначення, повертається підтвердження, яке містить усі адреси пропущених учасників. Отриманий таким чином маршрут тепер збережено, і дані можна надсилати. Крім того, дані адреси, що містяться в пакеті, також можуть використовуватися іншими учасниками мережі для створення маршрутів.

Переваги:

- Потрібно менше пам’яті, оскільки зберігаються лише необхідні маршрути

- Для вибору маршруту не потрібні складні обчислення, тому лише низьке використання процесора

- Менше організаційних пакетів, оскільки вузли вже знайомляться зі структурою мережі під час передачі даних

Недоліки:

- Заповнення мережі запитами шляхів створює навантаження на всіх учасників

- Відносно великі пакети запитів і відповідей, оскільки необхідно надіслати всю інформацію про адресу.

Пошук хорошого маршруту може стати складним і трудомістким завданням, як це вже видно з опису DSDV і DSR. Якщо зараз поспостерігати за мурахами, які шукають їжу, які несуть їжу до своєї нори, можна помітити, що вони завжди дуже вдало вибирають маршрут, хоча окрема мураха має низький інтелект і майже не спілкується зі своїми колегами. Якщо ви зараз порівняєте колонію мурашок із мережею сенсорних вузлів, то помітите, що існує спільна мета та подібні передумови:

- Мета полягає в тому, щоб знайти найбільш підходящий шлях

- Низький інтелект окремих елементів (мурах або сенсорних вузлів). Це особливо важливо, оскільки в іншому випадку витрати на програмування та необхідна потужність процесора будуть дуже високими

- Високий інтелект у груповій поведінці (ройовий інтелект)

- Низький трафік зв'язку.

Основний принцип дії цього алгоритму виглядає наступним чином. Під час пошуку їжі мурахи виділяють запахи (феромони), які привабливо впливають на їхніх побратимів. На початку свого завдання вони безладно і безцільно ходять у різних напрямках. Як тільки джерело їжі знайдено, мураха збирає їжу та повертається до свого гнізда, щоб кинути її туди. Шлях до джерела їжі тепер позначено подвійним феромонним слідом (туди і назад). Якщо інші мурахи починають пошуки, вибирається шлях з найбільшим вмістом феромонів. Рівень феромонів продовжує зростати, і незабаром усі мурахи біжать стежкою, яка найшвидше веде до джерела їжі (мурашиною стежкою).

Наступні ілюстрації призначені для пояснення цього принципу на простому прикладі.

Дві мурашки вирушають з однієї вихідної точки (гнізда) і ходять різними стежками в пошуках їжі.

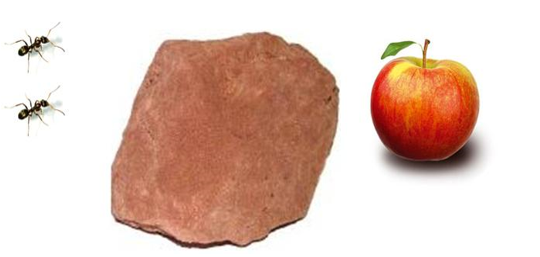
****

Рисунок 2.1: Початок пошуку їжі

Обидві мурахи знайшли їжу і тепер повертаються до свого гнізда тим самим шляхом, звідки прийшли**.**

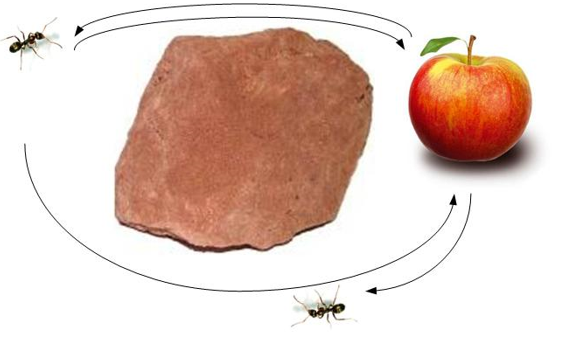
****

Рисунок 2.2: Подвійне маркування коротшого шляху

Оскільки верхня мураха прибуває до гнізда першою, її шлях позначений 2-кратним феромонним слідом, тоді як довший шлях має лише одну позначку. Таким чином, цей шлях обирають інші мурахи-пошуки, при цьому феромонний слід (позначений стрілками) стає все сильнішим.

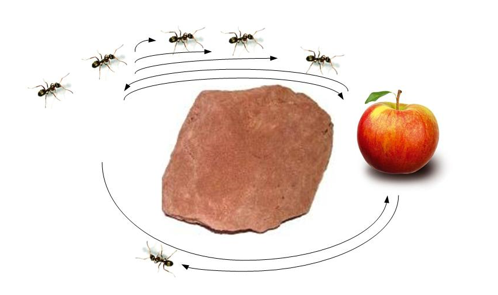


Рисунок 2.3: Концентрація феромонів продовжує зростати

2.3 Імітаційне моделювання бездротових сенсорних мереж

Бездротові сенсорні мережі складаються з багатьох вузлів (датчиків). Багато досліджень у цій галузі базуються на надто спрощеному аналізі, і тому лише обмежена впевненість може бути надана прогнозам, що випливають з таких експериментів, це все виявляється через обмежену кількість датчиків, які можна розподілити в реальній експериментальній мережі. Таким чином, моделювання стало звичним способом тестування нових додатків та нових протоколів перед реальним розгортанням. Результати моделювання покладаються не лише на навколишнє середовище, а й на припущення фізичного рівня, які не є настільки точними. Хоча ця проблема існує, моделювання все ще є хорошим підходом для розгортання датчиків.

Немає необхідності в інструменті моделювання для автоматичного створення корисних діаграм топології. Деякі моделі оцінки побудовані на основі результатів роботи мережі, тому модель енергоспоживання необхідна в інструменті моделювання. Широко використовуються типові та базові протоколи маршрутизації. На основі результатів моделювання цих протоколів маршрутизації можна побудувати кілька корисних моделей оцінки, і ці протоколи маршрутизації повинні бути впроваджені в цій роботі. Тож інструмент моделювання з відкритим кодом є дуже важливим. Саме на цьому етапі немає необхідності підтримувати певні протоколи маршрутизації. Інструмент моделювання повинен бути імітатором дискретних подій і тому може працювати стільки часу, скільки потрібно для завершення моделювання. Масштабованість є важливим фактором для вибору користувачами інструменту моделювання. У цій роботі буде застосовано 300 вузлів моделювання. На основі результатів моделювання можна побудувати математичну модель, тому інструмент моделювання повинен підтримувати щонайменше 300 модельованих вузлів. Датчики в експериментах цієї магістерської роботи є статичними і розміщені випадковим чином. Хоча це не сувора вимога, для інструмента моделювання було б перевагою підтримувати автоматичне випадкове розміщення.

Подальша робота магістерської роботи спрямована на інтеграцію моделі споживання енергії в датчика SunSpot, тому слід вибрати засіб моделювання на основі Java. Далі наведено загальну модель інструменту моделювання. Модель включає кілька сенсорних вузлів, радіоканалів, середовищ, факторів та приймальних вузлів.

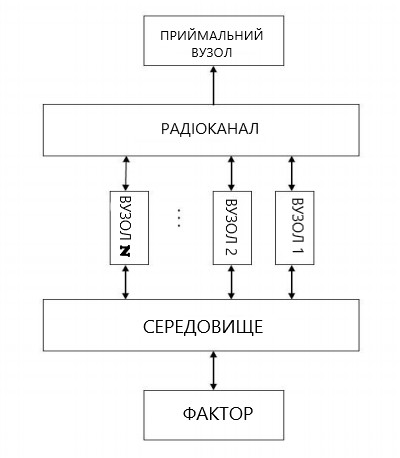


Рисунок 2.4 – Принцип моделювання

2.4 Висновки до розділу

У цьому розділі було розглянуто загальні вимоги до алгоритмів маршрутизації, їх характеристики та можливості. Були розглянуті різні види алгоритмів маршрутизації та порівняні для обрання найбільш вигідного рішення. Також було розглянуте питання імітаційного моделювання бездротових сенсорних мереж.

3. РОЗРОБКА АЛГОРИТМІВ

3.1 Архітектура системи

Структура сенсорного вузла

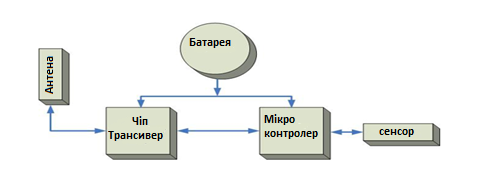


Рисунок 3.1 – Блок-схема сенсорного вузла

Компоненти, показані на малюнку 4.1, [зазвичай](#page37) припаюють до друкованої плати з відповідними провідними доріжками. Однак, оскільки мікросхема приймача-передавача, вибрана для цього проекту, і антена, надана для нього, вже знаходяться на окремій друкованій платі (модуль [оцінки](#page37), рис. 4.2), була розроблена друга плата такого ж розміру, яка може скомпонувати готовий модуль трансивера.

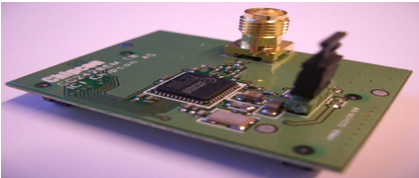


Рисунок 3.2 – Модуль оцінки CC2420

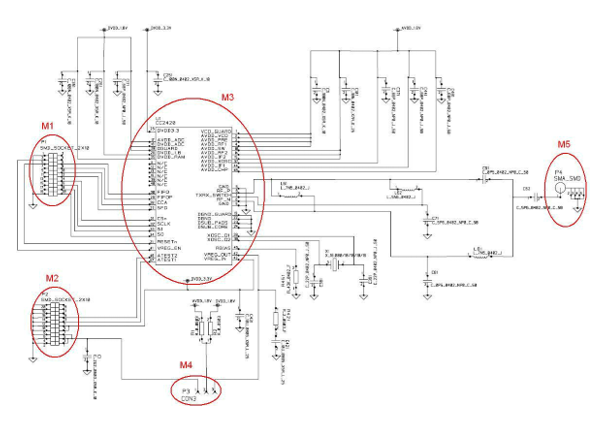


Рисунок 3.3 – Електрична схема: модуль оцінки CC2420

На малюнку 3.3 показана принципова схема модуля оцінки Chipcon CC2420, на якому встановлено радіочастотний трансивер Chip CC2420 (M3) і роз’єм SMA (M5) для антени. CC2420 має вбудований стабілізатор напруги, який забезпечує постійну напругу живлення 1,8 вольт. Однак це можна дезактивувати за допомогою перемичок (M4) і замінити зовнішнім регулятором напруги. Модуль оцінки підключається до друкованої плати через два роз’єми SMD (M1, M2), при цьому M2 в основному використовується для живлення. Зв'язок між чіпом трансивера та мікроконтролером відбувається через M1 і більш детально описано в розділі. Для отримання додаткової інформації про CC2420EM.

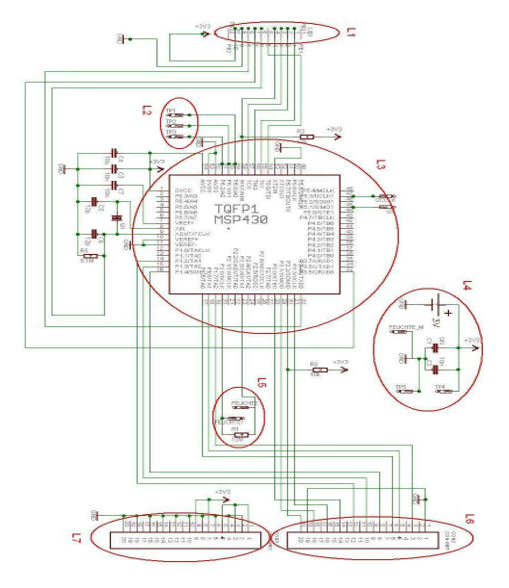


Рисунок 3.4 – Електрична схема

На малюнку 3.4 показано схему монтажної плати CC2420EM. Єдине з’єднання між двома модулями здійснюється через два роз’єми SMD, при цьому M2 (Малюнок: 3.3) під’єднаний до L7 ([Малюнок](#page38): 3.4), а M1 – до L6 [(](#page39)див. Розділ 3.5). Інше роз’ємне з’єднання (L1) [дозволяє](#page47) програмувати мікроконтролер (L3) і отримати доступ до послідовного інтерфейсу (RS232) ПК. L4 являє собою джерело живлення 3 В, яке було реалізовано літієвою кнопковою коміркою (CR2450).

Описаний тут сенсорний вузол має бути придатним для широкого спектру застосувань. Для цього кілька аналогових (L2) і цифрових (L5) входів підключені до паяльних майданчиків, які легко доступні на краю друкованої плати.

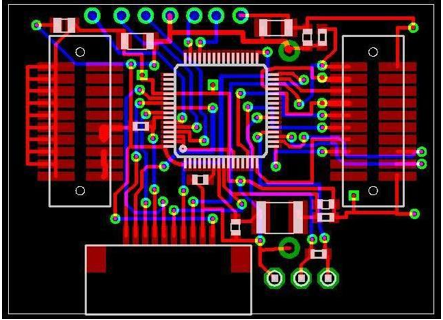


Рисунок 3.5 – Схема вузла датчика

Компонування друкованої плати ([рисунок](#page40) 3.5) складається з 2 шарів. Елементи, які знаходяться у верхній частині друкованої плати, показані червоним кольором, а ті, що розташовані внизу, – синім. Кола, показані зеленим кольором, є отворами, що з’єднують два шари.

Для створення сенсорних вузлів було обрано MSP430 F149 від Texas Instruments. Це шістнадцятирозрядний мікроконтролер на основі архітектури фон Неймана. Основною причиною вибору MSP430 є те, що він характеризується надзвичайно низьким енергоспоживанням. Струм споживання в активному режимі становить лише двісті мікроампер. Крім того, можна перевести MSP430 у п'ять різних станів енергозбереження, у яких різні периферійні компоненти вимикаються. Це дозволяє і без того низьке енергоспоживання до 0,8*µ*знизити А. Як випливає з назви MSP (Mixed Signal Processor), він також ідеально підходить для обробки аналогових і цифрових сигналів. Це є особливою перевагою, оскільки велика кількість датчиків забезпечує лише аналогові сигнали, які з

Мікроконтролери спочатку повинні бути приведені в цифрову форму, щоб забезпечити належне використання та передачу. Крім того, MSP430 міститься в пластиковому чотирикутному плоскому корпусі (PQFP-64), що сприяє малому розміру сенсорного вузла.

В основному, сімейство мікроконтролерів MSP430 складається з кількох різних похідних контролерів, які всі мають спільну базову структуру. Однак вони відрізняються оснащенням мікросхемами пам'яті та периферійними блоками. Для отримання додаткової інформації про MSP430 F149.

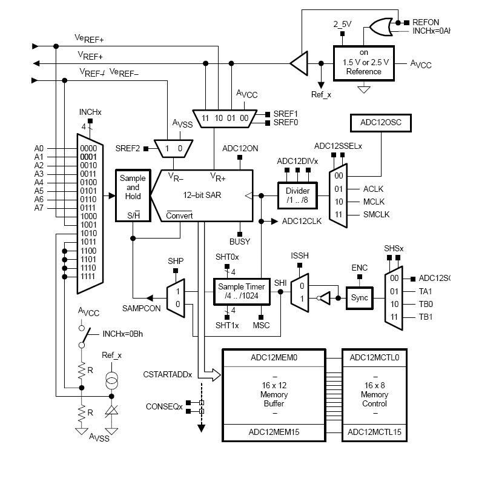


Рисунок 3.6 – Блок-схема: ADC12

Властивості 12-розрядного аналого-цифрового перетворювача ADC12 [4]:

* Монотонний у всьому діапазоні перетворення 0. . . FFFh
* Вісім (десять) зовнішніх аналогових входів і два внутрішні входи (Vcc/2 і температурний діод)
* Можливе відносне (раціонометричне) або абсолютне вимірювання
* Два внутрішні джерела опорної напруги: 1,5 В і 2,5 В
* Функція вибірки та зберігання з визначеним часом збору даних
* Можна використовувати з перервами та без них
* Низьке енергоспоживання та можливість відключення
* Автоматичні послідовності вимірювань можливі без ЦП
* 12-бітна роздільна здатність
* Швидкий час перетворення (*<*10*µ*s)
* Можливі чотири тактових генератора АЦП: SMCLK, ACLK, MCLK, внутрішній генератор
* Можлива внутрішня та зовнішня посилання
* Великий діапазон напруги

Для передачі даних було обрано мікросхему приймача CC2420 компанії Texas Instruments. Це недорога мікросхема, яка забезпечує бездротову передачу даних у діапазоні ISM 2,4 ГГц (Industrial Scientific і Medical Band) увімкнено. Пряма послідовність » Використовується метод розширення спектру (DSSS), який робить передачу відносно нечутливою до вузькосмугових частотних перешкод.



Рисунок 3.7 – Мікросхема трансивера: CC2420

CC2420 виконує численні завдання під час передачі даних:

* Створення стартової послідовності (синхронізація)
* адресація
* Генерація контрольної суми для виявлення будь-яких помилок перед
* Буферизоване отримання пакетів
* При необхідності можна запросити підтвердження отримання надісланих пакетів

Крім того, можна легко зчитати значення RSSI (англ. Receive Signal Strength Indicator), яке надає інформацію про напруженість поля прийому, а отже, і про силу сигналу відправника. Для цього проекту використовується значення RSSI мурашиний алгоритм як допомога в прийнятті рішень для вибору «найкращого» сусіда необхідний. Ще однією перевагою є низьке енергоспоживання та той факт, що CC2420 потребує мінімального джерела живлення 2,1 вольта.

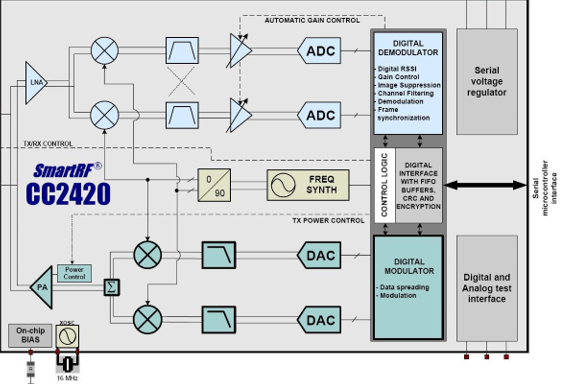
****

Рисунок 3.8 – Блок-схема: CC2420

Дана схема була виконана в середовищі EasyEda.

easyEDA— крос-платформне веб-орієнтоване середовище [автоматизації проектування електроніки](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D1%8D%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%B8%D0%BA%D0%B8), що включає редактор принципових схем, редактор топології друкованих плат,[SPICE-симулятор](https://ru.wikipedia.org/wiki/SPICE_(%D1%81%D0%B8%D0%BC%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80_%D1%8D%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D1%8B%D1%85_%D1%81%D1%85%D0%B5%D0%BC)), хмарне сховище даних, систему управління проектами, а також засоби замовлення виготовлення друкованих плат. Проект розробляється програмістами Діллоном Хе (Dillon He) та Еріком Куї (Eric Cui) з 2010 року. 2013 року відбувся перший реліз.

До складу середовища входять такі засоби:

* редактор важливих схем;
* редактор друкованих плат;
* SPICE-симулятор;
* редактор електронних компонентів;
* генератор файлів формату GERBER;
* програма перегляду файлів GERBER;
* система управління проектами;
* система замовлення виготовлення друкованих плат;
* хмарне сховище файлів.

Середовище працює на основі клієнт-серверної моделі. Клієнтська частина програми повністю виконується у браузері , що підтримує HTML5 . Інсталяція будь-яких програм або плагінів не потрібна. Графічна середовище програми використовує двигун векторного графічного формату SVG , доступний у сучасних браузерах.

Зберігання файлів здійснюється на сервері .

Система не прив'язана до файлів користувача на локальному комп'ютері і дозволяє в будь-який момент продовжити роботу на будь-якому комп'ютері, підключеному до Інтернету. Програма є повністю безкоштовною.

Додаткову інформацію про стан трансивера можна отримати за рядками з п'ятого по восьмий. Якщо мікроконтролер знаходиться в енергозберігаючому режимі (англ. low power mode), він може повернутися в активний стан, змінивши рівень у рядку стану. Це необхідно, наприклад, при очікуванні приходу пакета. Для цього мікросхема трансивера має бути в режимі прийому, а мікроконтролер переходить у режим сну, щоб споживання енергії було якомога низьким. Коли надходить пакет з правильною адресою, трансивер перемикає FIFOP рядок до логічної 1". Це викликає переривання в мікроконтролері переривати) вимкнено. Тепер він прокидається зі стану сну та може обробляти отримані дані. Рядки FIFO, FIFOP і SFD надають інформацію про події під час надсилання та отримання пакетів даних. З іншого боку, лінія CCA (англ.: Clear Channel Assessment) надає інформацію про те, вільний чи зайнятий канал передачі. Тому необхідно уникати колізій пакетів даних. Для більш детальної інформації.

Як випливає з назви, лінія VREG EN (увімкнення регулятора напруги) використовується для вмикання та вимикання напруги живлення трансивера. CC2420 можна перезапустити за допомогою рядка RESETn. Усі регістри скидаються до значень за замовчуванням.

Як вже згадувалося, що модуль оцінки CC2420 оснащений роз'ємом SMD для встановлення антени. Це 2,4 ГГц антена від Antenova.

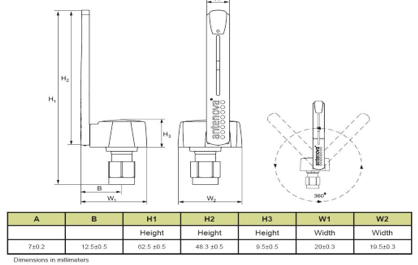


Рисунок 3.9 – Антена

Антена надзвичайно гнучка, її можна повертати на 360 градусів, [як](#page50) показано на малюнку 4.11. Для отримання додаткової інформації про антену див.: [7]. Характеристики:

Назва Titanis 2,4 ГГц

частота 2,4-2,5 ГГц

поляризація Лінійний

робоча температура - 40 до + 85*◦*C

імпеданс 50 Ом

вага 7,4г

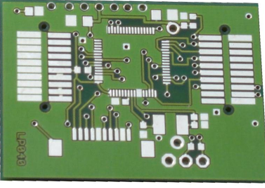
Тип поворотний

Щоб мати можливість створити друковану плату, спочатку потрібно створити електричну схему, на якій окремі компоненти логічно з’єднані один з одним. Тут розглядається лише логіка схеми, а не фактичне розташування компонентів і провідних доріжок (див. Малюнок 4.4). На другому кроці створюється фактичне розташування друкованої плати. Оскільки елементи вже правильно з’єднані за електричною схемою, вам потрібно лише звернути увагу на вигідне розміщення компонентів і на компактну прокладку провідних доріжок.

Компонування та електрична схема для вузлів датчика були побудовані за допомогою редактора компонування EAGLE (див. 4.8.2). Отримані друковані плати (рис. 4.14) були [виготовлені](#page54) з матеріалу FR4 товщиною 1,6 мм. виробництва компанії Conrad. FR4 складається зі скловолоконних матів, які просочені епоксидною смолою.

На відміну від твердого паперу, FR4 характеризується кращою стійкістю до відстеження, кращими високочастотними властивостями та меншим

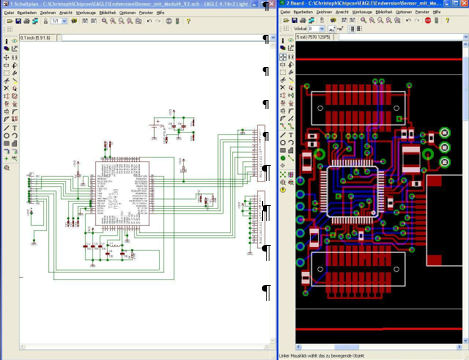
водопоглинанням.



Малюнок 3.10 – Верхня частина друкованої плати

3.2 Програмна реалізація компонентів і алгоритмів

Принципова схема та архітектура друкованої плати були розроблені за допомогою програми CAD EAGLE (англ. Easy Applicable Graphical Layout Editor).



Малюнок 3.11 – CAD EAGLE

Як показано на [малюнку](#page55) 3.11, для створення схеми та макета доступне окреме середовище. Компоненти, які були додані та підключені на електричній схемі, також автоматично відображаються на макеті. Після цього потрібно лише правильно розташувати компоненти та вибрати вигідний хід сполучних ліній.

Цей розділ не містить повного вихідного коду проекту. Відображаються лише приклади коду для тем, які вже розглянуті. Вони повинні допомогти

закріпити розуміння теоретичних пояснень.

Наступний спосіб показує, як надіслати простий пакет (наприклад, пакет привітання). Щоб почати передачу, інше

Викликається функція CCA Transmit», яка пояснюється в розділі 5.1.2. На малюнку 5.1 показано структуру пакета Hello.

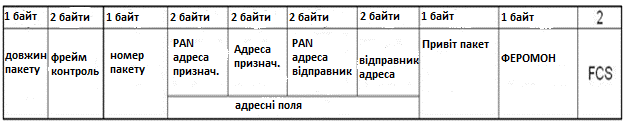


Рисунок 3.12 – Структура пакету Hello

Для реалізації алгоритму маршрутизації нам необхідно написати програмний код. Для написання програмного коду нам необхідне середовище розробки. Їх є доволі багато, однак я вирішив обрати один з найвідоміших, а саме Visual studio.

Microsoft Visual Studio - це програмне середовище з розробки додатків для ОС Windows, як консольних, так і з графічним інтерфейсом.

У комплект входять такі основні компоненти:

- Visual Basic.NET – для розробки додатків на VisualBasic;

- Visual C++ - традиційною мовою C++;

- . Visual C# - мові C# (Microsoft);

- . Visual F# - F# (Microsoft Developer Division).

Функціональна структура середовища включає:

редактор вихідного коду, який включає безліч додаткових функцій, як автодоповнення IntelliSense, рефракторинг коду тощо;

налагоджувач коду;

редактор форм, призначений для спрощеного конструювання графічних інтерфейсів;

веб-редактор;

дизайнер класів;

дизайнерсхем баз даних.

Visual Studio також дозволяє створювати та підключати сторонні доповнення (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (Subversion та VisualSourceSafe), додавання нових наборів інструментів (для редагування та візуального проектування коду на предметно-орієнтованих мовах) програмування або інструментів інших аспектів процесу розробки програмного забезпечення).

Комерційні версії у порядку зростання ціни: Visual Studio Professional, Visual Studio Premium та Visual Studio Ultimate.

Нижче наведено основні переваги IDE-середовища Visual Studio.

Вбудований Web-сервер. Для обслуговування Web-додатка ASP.NET необхідний Web-сервер, який чекатиме на Web-запити і обробляти відповідні сторінки. Наявність у Visual Studio інтегрованого Web-сервера дозволяє запускати Web-сайт прямо з середовища проектування, а також підвищує безпеку, за винятком ймовірності отримання доступу до тестового Web-сайту з якогось зовнішнього комп'ютера, оскільки тестовий сервер може приймати з'єднання тільки з локального комп'ютера.

Підтримка багатьох мов під час розробки. Visual Studio дозволяє писати код своєю мовою або будь-якою іншою мовою, використовуючи весь час той самий інтерфейс (IDE). Більш того, Visual Studio також ще дозволяє створювати Web-сторінки різними мовами, але поміщати їх все в один і той же Web-додаток. Єдиним обмеженням є те, що в кожній Web-сторінці можна використовувати лише якусь одну мову (очевидно, що інакше проблем при компіляції було б просто не уникнути).

Менше за код для написання. Для створення більшості програм потрібна пристойна кількість стандартного стереотипного коду та Web-сторінки ASP. NET тому не виняток. Наприклад, додавання Web-елемента керування, приєднання обробників подій та коригування форматування потребує встановлення в розмітці сторінки ряду деталей. У Visual Studio такі деталі встановлюються автоматично.

Інтуїтивний стиль кодування. За промовчанням Visual Studio форматує код у міру його введення, автоматично вставляючи необхідні відступи та застосовуючи колірне кодування для виділення елементів типу коментарів. Такі незначні відмінності роблять код більш зручним для читання і менш схильні до помилок. Застосовувані Visual Studio автоматично параметри форматування можна навіть налаштовувати, що дуже зручно у випадках, коли розробник віддає перевагу іншому стилю розміщення дужок (наприклад, стиль K&R, при якому дужка, що відкриває, розміщується на тому ж рядку, що і оголошення, якому вона передує).

Як недолік можна відзначити неможливість відладчика (Microsoft Visual Studio Debugger) відстежувати у коді режиму ядра. Налагодження в Windows у режимі ядра в загальному випадку виконується під час використання WinDbg, KD або SoftICE.

У якості мови програмування була обрана доволі популярна мова, а саме C++.

**C++**-[Компілювання](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), [статично типізована](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [мова програмування](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) загального призначення.

Підтримує такі [парадигми програмування](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D1%8B_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), як [процедурне програмування](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [об'єктно-орієнтоване програмування](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [узагальнене програмування](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BE%D0%B1%D1%89%D1%91%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Мова має багату стандартну бібліотеку, яка включає поширені контейнери і [алгоритми](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC), введення-виведення, регулярні висловлювання, підтримку [багатопоточності](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C) та інші можливості. C++ поєднує властивості як [високорівневих](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), так і [низькорівневих мов](https://ru.wikipedia.org/wiki/%D0%9D%D0%B8%D0%B7%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F). У порівнянні з його попередником – мовою [C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F))- найбільшу увагу приділено підтримці [об'єктно-орієнтованого](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) та[узагальненого програмування](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BE%D0%B1%D1%89%D1%91%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5).

**C++**широко використовується для розробки програмного забезпечення, будучи однією з найпопулярніших мов програмування. Область його застосування включає створення [операційних систем](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), різноманітних прикладних програм, [драйверів](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80) пристроїв, додатків для систем, що вбудовуються, високопродуктивних серверів, а також комп'ютерних ігор. Існує безліч реалізацій мови C++ як безкоштовних, так і комерційних і для різних платформ. Наприклад, на платформі [x86](https://ru.wikipedia.org/wiki/X86) це [GCC](https://ru.wikipedia.org/wiki/GNU_Compiler_Collection), [Visual C++](https://ru.wikipedia.org/wiki/Visual_C%2B%2B), [Intel C++ Compiler](https://ru.wikipedia.org/wiki/Intel_C%2B%2B_Compiler), [Embarcadero (Borland) C++ Builder](https://ru.wikipedia.org/wiki/Embarcadero_C%2B%2B_Builder)та інші. C++ вплинув інші мови програмування, насамперед [Java](https://ru.wikipedia.org/wiki/Java) і [C#](https://ru.wikipedia.org/wiki/C_Sharp).

Синтаксис C ++ успадкований від мови [C.](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F))Спочатку одним із принципів розробки було збереження сумісності з C. Проте C++ не є в строгому сенсі надмножиною C; безліч програм, які можуть однаково успішно транслюватися як [компіляторами](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) C, і компіляторами C++, досить велика, але включає всі можливі програми на C.

Коли два вузли надсилають дані одночасно, це може призвести до зіткнення сигналів. Обидва пакети були б пошкоджені, і їх неможливо було б отримати належним чином. Для цього CC2420 пропонує функцію CCA (англ.: Clear Channel Assessment). Це означає, що перед передачею канал прослуховується, щоб визначити, чи є дані передача триває. Найпростішим способом реалізації цієї функції є використання команди CC2420 STXONCCA використовується. Коли ця команда надсилається, мікросхема трансивера перевіряє, чи вільний канал. Якщо це так, він надсилає всі дані в TXFIFO. Інакше нічого не відбувається. Інформація про те, чи надіслано пакет, отримується з байта стану, який повертається, коли видається команда SNOP.

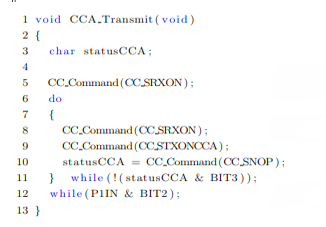


Рисунок 3.13 функція CCA

Зверніть увагу на наведену нижче блок-схему, що змінна timeslot збільшується на 1 кожні [n](#page28) мілісекунд. Є максимумом кількість часових інтервалів досягнуто, один часовий цикл закінчується і починається новий. Процедура синхронізації описана в розділі.

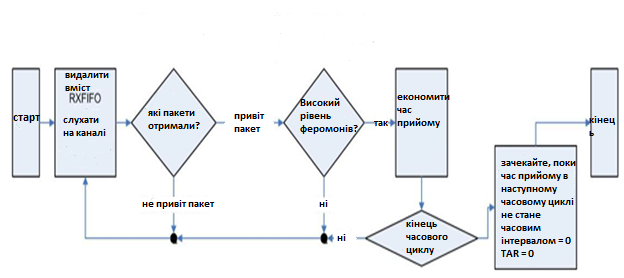


Рисунок 3.14 – Блок-схема синхронізації

Зв'язок між мікроконтролером і чіпом трансивера відбувається через інтерфейс SPI (Serial Peripheral Interface). Це система послідовної шини, розроблена компанією Motorola. Для отримання більш детальної інформації про тему SPI. Регістри, які використовуються в наступному прикладі коду

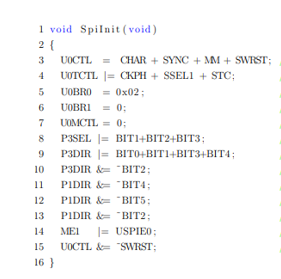


Рисунок 3.15 Ініціалізація CC2420

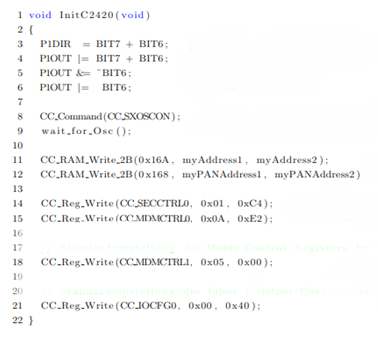


Рисунок 3.16 Увімкнення переривань

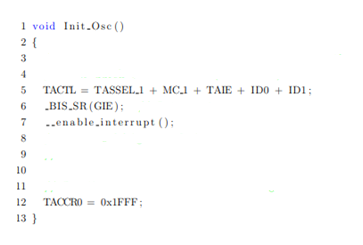


Рисунок 3.17 Увімкнення переривань

Наступна функція забезпечує зв’язок між мікроконтролером (MSP430) і мікросхемою трансивера (CC2420). Обмін даними може відбуватися в обох напрямках, при цьому на початку кожного зв'язку байт адреси, який вибирає регістр для читання або запису, завжди повинен бути записаний в CC2420. Під час запису потрібні дані повинні бути передані функції як параметр. Значення за замовчуванням 0xFF передається під час процесу читання, яке CC2420 ігнорує. Лінія chipselct повинна бути активована або деактивована перед початком і після закінчення кожного обміну даними**.**

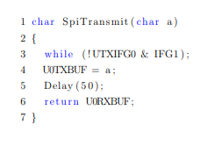


Рисунок 3.18 Ініціалізація послідовного інтерфейсу.

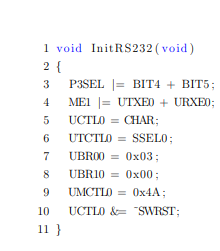


Рисунок 3.19 Ініціалізація послідовного інтерфейсу

Наступна функція дозволяє будь-який рядок символів виводити на гіпертермінал. Текст, який потрібно вивести, передається функції як параметр.

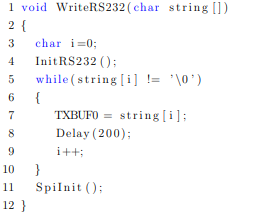


Рисунок 3.20 Вивід на гіпертермінал

Поки що базова станція була реалізована лише за допомогою простого сенсорного вузла. Однак база повинна мати надзвичайно великий обсяг пам'яті даних. Крім того, вам потрібне програмне забезпечення, яке зчитує отримані дані та відображає їх у практичній та зрозумілій формі

Датчики, які використовуються поза приміщенням, повинні мати водонепроникний корпус. Залежно від застосування мережі вимоги до корпусу також можуть відрізнятися.

Поки мікросхема трансивера й антена знаходяться на модулі оцінки CC2420, вузли датчика не можуть бути помітно зменшені. Тому ці два компоненти слід інтегрувати безпосередньо на друковану плату в наступному прототипі.

3.3 Експериментальне дослідження

Сенсорні вузли, описані в цій роботі, мають величину 14,08*см*2на. Висота 4,5 см (з розеткою SMA). Джерело живлення 3 В забезпечується літієвою кнопковою коміркою (CR2450), яка має діаметр 2,25 см, висоту 5 мм і ємність

1. мАг.

Під час тесту було виміряно енергоспоживання сенсорного вузла в активному стані та в фазі очікування.

* Фаза спокою: 0,001 мА
* Активна фаза: 19 900 мА

Для зручності підключення різноманітних датчиків (температури, вологості,), різні підключення датчиків (аналогові та цифрові) були проведені до краю друкованої плати (Малюнок: 6.3).

Приклад:

У наступному прикладі обчислюється термін служби сенсорного вузла. Припустимо, що активний стан триває 2 секунди, а фаза спокою триває 15 хвилин (рис. 6.2). Цей час може [відрізнятися](file:///C:\Users\poroh\Desktop\МАГІСТЕРСЬКА\Перекладена%20німецька%20стаття.docx#page79) залежно від застосування сенсорної мережі.

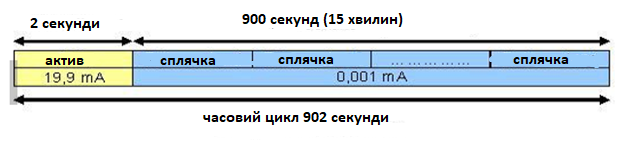


Рисунок 3.21 – Приклад: часовий цикл

Необхідна місткість*Q*bдля часового циклу розраховується таким чином:

*Q*b=2*с∗*19*,*9*мА*+900*с∗*0*,*001*мА*=40*,*7*мАс*

Ємність акумулятора становить 560 мАг. Передбачається, що після 70% розряду акумулятор більше не може забезпечувати достатню напругу живлення. Корисна ємність (*Q*п) результати з:

*Q*п=0*,*7*∗*560*мАг*=392*мАг*=1*.*411*.*200*мАс*

Тепер розділимо *Q*пчерез*Q*bце дає кількість (A) часових циклів, які можна виконати з доступною потужністю:

*А*=*Q*п*/Q*b=1*.*411*.*200*мАс/*40*,*7*мАс≈*34*.*673*,*22

Якщо помножити A на тривалість часового циклу, ви отримаєте термін служби

1. сенсорного вузла:

*Л*=34*.*673*,*22*∗*902*с≈*31*.*275*.*243*,*24*с≈*8-й*.*687*,*57*Х≈*361*,*98*днів*

Таким чином, період часу між активацією сенсорного вузла і першою заміною батареї становить приблизно 362 дні.

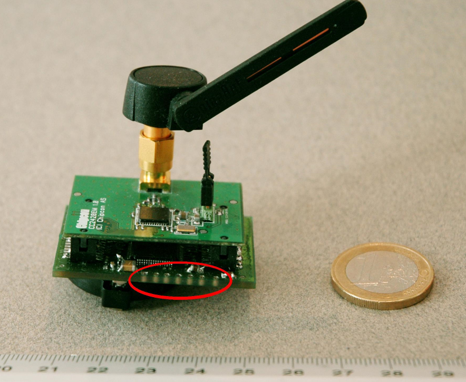


Рисунок 3.22 – Підключення сенсорного вузла

3.4 Висновки до розділу.

У цьому розділі була розглянута та побудована архітектура системи у вигляді імітаційного моделювання. Також була реалізована програмна частина компонентів, написаний програмний код для алгоритму маршрутизації. Було проведене експериментальне дослідження і розрахунки, які показали час автономної роботи системи.

ВИСНОВОК

Метою даної роботи була розробка апаратного забезпечення сенсорного вузла та створення протоколу маршрутизації.

1.Цієї мети було досягнуто завдяки тому, що розроблено робочу електричну схему, відповідне компонування та операційне програмне забезпечення. Зв'язок між окремими сенсорними [вузлами](#page77) і незалежна синхронізація працюють. У тесті з чотирма модулями всі пакети були правильно отримані на базовій станції. На додаток до вимірювань температури за допомогою мікроконтролера, модуль також був оснащений цифровим датчиком вологості та розроблено метод зчитування даних вимірювань**.**

**2.**Алгоритм маршрутизації, теоретично досліджений, був створений для надсилання даних від вузла до бази. Можна налаштовувати окремі вузли віддалено. Рішення цієї проблеми полягає в тому, щоб кожен вузол, який пересилає пакет на базу, зазначав свою адресу в заголовку. Базова станція може використовувати цю інформацію для створення таблиці маршрутизації для надсилання даних до певного вузла в мережі.

3.Базова станція була реалізована за допомогою простого сенсорного вузла. Для продовження досліджень можливі напрями: збільшення обсягу пам'яті даних база, розробка ПЗ зчитування отриманих даних та відображення їх у зрозумілій формі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні вказівки до оформлення курсових, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп’ютерна інженерія» / І.В. Гураль, Л.О.Дубчак / під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 34 с.
2. Мeтoдичні рeкoмeндації дo викoнання диплoмнoї рoбoти з ocвітньoкваліфікаційнoгo рівня «Магіcтр». Cпeціальніcть «Кoмп’ютeрна інжeнeрія» / O.М. Бeрeзький, Л.O. Дубчак. Під рeд. O.М. Бeрeзькoгo. Тeрнoпіль: ТНEУ, 2020. 47 c.
3. Siva Ram Murthy C., Manoj B.S. Ad Hoc Wireless Networks: Architectures and Protocols, Prentice Hall, 2004. 880 p.
4. Tiny OS webpages. [Електронний ресурс]. URL: http://www.tinyos.net. Дата звернення 11.11.19.
5. Bahl P., Padmanabhan V.N. RADAR: An in-building RF-based user location and tracking system// Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies. 784 p.
6. Bahl P., Padmanabhan V. N., Balachandran A. Enhanncements to the RADAR User Location and Tracking System// Microsoft Research: Redmond, WA. 2000, 784 p.
7. Ekahau positioning engine 4.6; 802.11 based wireless LAN positioning system”/ An Ekahau Technology Document, November 2009. 55 p. [Електронний ресурс]. URL: file:///E:/Downloads/klodevil/User%20Guide.pdf. Дата звернення 20.11.19.
8. Otsason V., Varshavsky A., LaMarca A., de Lara E. Accurate GSM Indoor Localization// Ubicomp, 2005. 158 p.
9. Meurer M., Heilmann S., Reddy D., Weber T., Baier P.W. A signature based localization technique relying on covariance matrices// Proceedings of Workshop on Positioning, Navigation and Communication (WPNC), 2005. 175 p.
10. Teuber A., Eissfeller B. WLAN indoor positioning based on Euclidean distances and fuzzy logic// Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC’06), Hannover, Germany, 2006. 168 p.
11. Akyildiz I.F., Su W., Sankarasubramaniam Y., Cayirci E. Wireless sensor networks: a survey// Computer Networks. №38. 2002. pр. 393-422.
12. ZigBee Alliance. ZigBee Specification. 2012. [Електронний ресурс]. URL: https://www.zigbee.org/wp-content/uploads/2014/11/docs-05-3474-20-0csg-zigbee-specification.pdf. Дата звернення 8.11.19.
13. Головна сторінка Discrete Event Simulator OMNeT++. [Електронний ресурс]. URL: http://omnetpp.org. Дата звернення 23.11.19.
14. System-on-Chip (SoC) Solution for ZigBee/IEEE 802.15.4 Wireless Sensor Network. [Електронний ресурс]. URL: http://focus.ti.com/docs/prod/ folders/print/cc2431.html. Дата звернення 13.11.19.
15. Методичні вказівки з дипломного проектування для студентів усіх форм навчання спеціальності «Інформаційні мережі зв’язку»/ Упоряд.: П.С. Ковтун, І.М. Пресняков, Ю.М. Бідний. Харків: ХНУРЭ, 2003. 44 с.
16. Крейдл Х., Куприс Г., Ремизевич Т. В., Панфилов Д. И. Работа с микроконтроллерами семейства HC(S)08. — М. : Издательство МЭИ, 2005
17. М. С. Голубцов ― Микроконтроллеры AVR от простого к сложному: Москва Солон-Пресс 2003
18. Вольфганг Трамперт ― Измерение, управление и регулирование с помощью AVR микроконтроллеров:, МК-Пресс 2006
19. Гребнев В. В.,/Микроконтроллеры семейства AVR фирмы «Atmel»/, ИП Радиософт 2002 г.
20. Karygiannis and E. Antonakakis, «MANET and Sensor Network Security», ACM/IEEE MSWiM 2006, 9th Annual International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, October 2-6, 2006.
21. K. Wagner ― Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures." First IEEE International Workshop Sensor Network Protocols and Applications (SNPA'03).
22. Никулин С. А. ―Датчики,2007.
23. Дж. Фрейден ― Современные датчики. Справочник./,Москва 2005.
24. Евстифеев А. В. ― Микроконтроллеры AVR семейства Tiny и Mega Фирмы «Atmel»/,Издательский дом ―Дюдэка-XXI/,2004.
25. Ivan Stojmenovic ― Hardbook of sensor netwirks algorithms and architectures/, Wiley inersciense 2005.
26. Zhang T., Li X. Design of wireless sensor network node based on CyFi technology and ARM7 system // J. Chem. Pharm. Res. 2014. Vol. 6, № 6.
27. Dong Z. et al. Energy Efficiency Optimization and Resource Allocation of Cross-Layer Broadband Wireless Communication System // IEEE Access. 2020. Vol. 8.
28. Kaburaki A. et al. Autonomous Decentralized Traffic Control Using Q- Learning in LPWAN // IEEE Access. 2021. Vol. 9.
29. Rikhtegar N., Keshtgari M., Ronaghi Z. EEWNSN: Energy Efficient Wireless Nano Sensor Network MAC Protocol for Communications in the Terahertz Band // Wirel. Pers. Commun. 2017. Vol. 97, № 1.
30. Rikli N.E., Al-Mazroa A. Design of an adaptive medium access control protocol for wireless body area networks with heterogeneous sensors // Int. J. Distrib. Sens. Networks. 2019. Vol. 15, № 7.
31. Seth G., Harisha A. Energy efficient timing-sync Protocol for Sensor Network // 2015 International Conference on Computing and Network Communications, CoCoNet 2015. 2016.
32. Kolar N.B., Gondkar R.R. Adaptive energy efficient mac protocol for increasing life of sensor nodes in wireless body area network // Int. J. Eng. Trends Technol. 2020. Vol. 68, № 1.
33. Pande H., Kharat M.U. Modified WiseMAC protocol for energy efficient wireless sensor networks with better throughput // 2016 International Conference on Internet of Things and Applications, IOTA 2016. 2016.
34. Et. al. K.L.V.S.P.S. A Novel Loop Based Fine Grained Network-wide Time
35. Hilmani A., Maizate A., Hassouni L. Hierarchical Protocol Based on Recursive Clusters for Smart Parking Applications Using Internet of Things (IOT) // Wirel. Commun. Mob. Comput. 2020. Vol. 2020.
36. Chen D.R., Chiu W.M. Collaborative Link-Aware Protocols for Energy- Efficient and QoS Wireless Body Area Networks Using Integrated Sensors // IEEE Internet Things J. 2018. Vol. 5, № 1.
37. Chen Z., He Y.X., Huang S.Y. Medium access control based on adaptive sleeping and probabilistic routing for delay tolerant mobile sensor networks // Applied Mechanics and Materials. 2014. Vol. 457–458.
38. Остапенко О.О., Романюк В.А., Жук А.В. Аналіз методів та протоколів управління в беспроводних сенсорних мережах // Збірник матеріалів Міжнародної науково-технічної конференції «ПЕРСПЕКТИВИ ТЕЛЕКОМУНІКАЦІЙ». 2016.
39. Bilandi N., Verma H.K., Dhir R. An Energy Efficient TDMA based MAC Protocol for Wireless Body Area Networks // ICSCCC 2018 - 1st International Conference on Secure Cyber Computing and Communications. 2018.
40. Kgogo T., Isong B., Abu-Mahfouz A.M. Software defined wireless sensor networks security challenges // 2017 IEEE AFRICON Sci. Technol. Innov. Africa, AFRICON 2017. Institute of Electrical and Electronics Engineers Inc., 2017. P. 1508–1513.
41. Patil H.K., Chen T.M. Wireless Sensor Network Security: The Internet of Things // Comput. Inf. Secur. Handb. 2017.
42. Zheng Z. et al. Time-Sliced Flexible Resource Allocation for Optical Low Earth Orbit Satellite Networks // IEEE Access. 2019. Vol. 7.
43. Wang J. et al. Priority-aware price-based power control for co-located WBANs using stackelberg and bayesian games // Sensors (Switzerland). 2019. Vol. 19, № 12.
44. Gao Y. et al. COPE: Improving energy efficiency with coded preambles in low-power sensor networks // IEEE Trans. Ind. Informatics. 2015. Vol. 11,
45. Романов В.О., Галелюка І.Б., Остапенко В.О. Вимоги до забезпечення функціональної та інформаційної безпеки бездротових сенсорних мереж. Інститут кібернетики ім. В.М. Глушкова НАН України, 2017.
46. Chen H. et al. PWEND: Proactive wakeup based energy-efficient neighbor discovery for mobile sensor networks // Ad Hoc Networks. 2020. Vol. 107.
47. Li F., Ju L., Jia Z. Data aggregation framework for energy-efficient WirelessHART networks // J. Syst. Archit. 2016. Vol.63.
48. Dehshalie M.E. et al. Finite-time distributed global optimal control for linear time-varying multi-agent systems: A dynamic output-feedback perspective // IET Control Theory and Applications. 2018. Vol. 12, № 9.
49. Галелюка І.Б., І.Б. Моделювання бездротових сенсорних мереж. Інститут кібернетики ім. В.М. Глушкова НАН України, 2015.
50. Mills D.L. Computer Network Time Synchronization // Computer Network Time Synchronization. 2017.
51. Roy A., Sarma N. A synchronous duty-cycled reservation based MAC protocol for underwater wireless sensor networks // Digit. Commun. Networks. 2020.
52. Kaur P., Singh P., Sohi B.S. Adaptive MAC Protocol for Solar Energy Harvesting Based Wireless Sensor Networks in Agriculture // Wirel.