МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної роботи бакалавра

на тему: «Система управління “Розумний дім” на базі Arduino»

Виконала студентка 4 курсу групи ПІ-54

спеціальності 121 «Інженерія програмного забезпечення»

Раздобудова Валерія Олексіївна

(прізвище, ім’я та по-батькові)

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,

,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,

Національна шкала\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сугоняк І.І.

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Власенко О.В. .

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кравченко С.М..

(підпис) (прізвище та ініціали)

Житомир 2020

ЗМІСТ

[ВСТУП 3](#_Toc26911392)

[РОЗДІЛ 1. ТЕХНОЛОГІЯ “РОЗУМНИЙ” БУДИНОК 6](#_Toc26911393)

[1.1 Принципи роботи системи 6](#_Toc26911394)

[1.2 Існуючі концепції. Переваги та недоліки. 8](#_Toc26911395)

[1.3 Власний варіант реалізації 14](#_Toc26911396)

[РОЗДІЛ 2. АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ 16](#_Toc26911397)

[2.1 Arduino та Arduino Uno 16](#_Toc26911399)

[2.2 Розробка моделі програмного комплексу 18](#_Toc26911400)

[2.3 Arduino Mega2560 20](#_Toc26911401)

[РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 22](#_Toc26911402)

[3.1 Програмне забезпечення Arduino 22](#_Toc26911404)

[3.2 Програмне забезпечення бази даних 24](#_Toc26911405)

[3.3 Генератор файлів конфігурацій 27](#_Toc26911406)

[ВИСНОВКИ 29](#_Toc26911407)

[ЛІТЕРАТУРА 31](#_Toc26911408)

# ВСТУП

У наші дні спостерігається бурхливий ріст та розвиток інформаційних технологій. Те, що здавалось фантастикою декілька десятиліть назад, зараз є буденною звичайністю. Швидкість зростання обсягів даних та обчислювальних потужностей надзвичайно стрімкий. Сучасні мобільні пристрої зазвичай мають технічні характеристики в рази кращі, ніж комп’ютери минулого, а їхня собівартість зменшується з кожним роком. За такого рівня технічного прогресу у більшості звичайних побутових пристроїв вже є власний процесор, що керує ним та дозволяє автоматизувати роботу. В результаті таких досягнень виникає питання, а чи можливо створити єдину систему контролю пристроїв, що нас оточують? Чи можна автоматизувати більшу частину побутових процесів? Відповіддю на ці питання стали розробки системи "розумного" будинку. Багато різноманітних відомих та не дуже компаній займаються розробкою та створенням таких систем.

Основними перевагами системи "розумного" будинку є можливість автоматизації роботи пристроїв у домі. Це дозволяє автоматично контролювати параметри мікроклімату у приміщенні, вмикати та вимикати різного роду мультимедійні пристрої. Іншою корисною функцією даної системи є реалізація можливості енергозбереження. Якщо система контролює охолоджувальні та нагрівальні установки, то вона здатна ефективно витрачати енергію, адже очевидним є той факт, що немає ніякої необхідності підтримувати комфортну температуру у приміщені за умови відсутності господаря. Для нашої країни енергозбереження є дуже актуальним питанням, адже ціни на енергоносії суттєво зросли за останній час. При всіх позитивних моментах використання системи "розумного" будинку вона має і недоліки. Основним з них є вартість готової системи, адже комплексні технічні рішення від виробників зазвичай коштують занадто багато, щоб створити масовий інтерес до продукту. Іншим недоліком є фактична відсутність стандартизації на даний момент. Кожен з виробників реалізує концепцію "розумного" будинку виключно у своєму баченні. Він створює свою специфікацію, стандартизацію і модулі, які зазвичай не будуть працювати в системі від іншого виробника. Це створює складності для користувачів, адже при необхідності модернізації чи розширення системи вони будуть змушені використовувати тільки певні фіксовані елементи. Також проблемою готових рішень під ключ є неможливість модернізації їх користувачами власноруч. Очевидно, що потенційними користувачами систем концепції "розумного" будинку є люди з технічною освітою, адже вони як ніхто інший розуміють усі її переваги. Відсутність можливості власноруч щось змінити позбавляє ці рішення однієї великої складової - товариства користувачів. Ці організації фанатів зазвичай і займаються власноручною модернізацією, покращенням та оптимізацію проектів. Усім чудово відомі товариства користувачів Linux та Unix, які займаються розвитком проектів, їх модернізацією та покращенням. Якби у людей була можливість приймати участь у створені системи, то це б підвищило інтерес та прискорило процес її становлення. Враховуючи усі недоліки готових систем, які наведені вище, було прийнято рішення розробити власну відкриту концепцію "розумного" будинку та виконати її реалізацію.

Метою курсової роботи є дослідження та розробка функціональної схеми системи управління, вибір протоколів зв’язку елементів системи та розробка програми управління виконавчими приладами. Дані дослідження будуть в основному сфокусовані на певному об'єктно-орієнтованому методі CASE-технологій, методі та засобах проектування програмного забезпечення та уніфікації процесів проектування, який називається UML (Unified Modeling Language - уніфікована мова моделювання), що і являє собою об’єкт досліджень курсової.

UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи (UML-моделлю). UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація. Відповідно до своїх можливостей, UML дозволяє нам сповна виконати завдання курсової роботи.

# РОЗДІЛ 1. ТЕХНОЛОГІЯ “РОЗУМНИЙ” БУДИНОК

## Принципи роботи системи

Термін "розумний" будинок не має чіткого визначення, а тому під це поняття підпадає будь-яка система з автоматизованим керуванням приладами, яка спрощує життя людини та підвищує рівень її комфорту. Через нечіткі рамки виникло багато реалізацій з різним рівнем інтеграції та принципом роботи. Їх можна умовно поділити на три групи:

• Вбудовані системи з центральним контролером;

• Вбудовані системи без центрального контролера;

• Системи з інтеграцією, що налаштовується.

Перша група представляє з себе повністю налаштовану і встановлену виробником систему, яка керується центральним обчислювальним пристроєм і не передбачає прямої взаємодії своїх компонентів між собою. Всі призначені для користувача налаштування зберігаються на центральному пристрої (сервері), а периферія лише виконує отримані від нього інструкції і зазвичай не має вбудованої пам'яті і обчислювальних потужностей. Принцип роботи системи зображений на рис. 1.1.



Рисунок 1.1 – Діаграма, що представляє принцип роботи вбудованої системи з центральним контролером



Рисунок 1.3 – Діаграма, що представляє принцип роботи системи з інтеграцією, що налаштовується

## Існуючі концепції. Переваги та недоліки

В ідеальному варіанті "розумний" будинок є системою, в якій кожен прилад інтегрований в загальну екосистему, проте, зважаючи на різні протоколи зв'язку приладів, необхідності у здешевленні системи виробником та налаштування під конкретні завдання відбувається поділ на три основні групи за призначенням:

* Мультимедійний простір;

• Система розумного контролю мікрокліматичних параметрів приміщення;

* Змішана система.

Мультимедійне простір або мультимедійна система "розумний" будинок - це цілісна екосистема з мультимедійних пристроїв з загальним хабом пам'яті для контенту і розширеними можливостями взаємодії.

Ця концепція заснована на переважаючій функції контенту і складається з таких компонентів як Smart TV з доступом до Інтернет ресурсів і додатків, мультимедіа хаб, система об'ємного звуку, просунуті функції енергозбереження для техніки, віддалений контроль пристроїв. З появою готових пристроїв Smart TV і хмарних сервісів концепція може бути легко реалізована будь-яким користувачем без необхідності докуповувати щось окрім самих пристроїв. Є найбільш популярною в США і Канаді.

З появою доповненої реальності концепція мультимедійного "розумного" будинку знайшла нове життя у вигляді так званих шоломів віртуальної реальності. До основних функцій була додана модуляція віртуальних об'єктів на реальність і можливість взаємодіяти з результатом як з цілісною середовищем. Варто відзначити, що тепер для побудови системи потрібно всього лише один пристрій без додаткових аксесуарів і функціональність системи залежить тільки від вбудованого в шолом ПЗ. Приклад мультимедійної системи розумного будинку з доповненою реальністю на рис. 1.4.



Рисунок 1.4 – Приклад роботи Microsoft HoloLens

Порівняння аналогів наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння існуючих варіантів реалізації системи "розумний" будинок.

|  |  |  |  |
| --- | --- | --- | --- |
| Параметр | Meizu | Allone | Clipsal |
| Вартість | Середня | Висока (разом з модулями) | Невелика за модуль, середня за систему в цілому |
| Установка | Проста | Проста | Потребує попереднього налаштування |
| Налаштування | Не потребує | Через веб-сервіс | Через прошивку |
| Готові модулі | Мало | Багато | Не потребує |
| Масштабованість | Масштабується | Масштабується, але залежить від розміру приміщення | Масштабується |
| Взаємодія компонентів | Через смартфон | Через сам пристрій | Через хмарний додаток |
| Функціонал | Базовий | Майже необмежений | Майже необмежений |

Для створення UML-моделі буде використано один з поширених редакторів, але в першу чергу потрібно детальніше розглянути, що таке UML та порівняти можливості декількох редакторів, а потім запинитись на більш доцільному.

Отже, UML - це мова, призначена для візуалізації, специфікації, конструювання й документування програмних систем. Слово «уніфікований» у назві мови означає, що UML може використовуватися для моделювання широкого кола додатків від вбудованих систем і систем реального часу до розподілених веб-додатків. Виразні засоби мови дозволяють описати систему зі всіх точок зору, що мають відношення до розробки й розгортання.

Мова UML одночасно є простим і потужним засобом моделювання, який може бути ефективно використаний для побудови концептуальних, логічних і графічних моделей складних систем самого різного цільового призначення. Конструктивне використання мови UML ґрунтується на розумінні загальних принципів моделювання складних систем та особливостей процесу об'єктно-орієнтованого проектування (ООП) зокрема. Вибір засобів для побудови моделей складних систем зумовлює ті завдання, які можуть бути вирішені з використанням даних моделей.

Тому для порівняння було обрано такі засоби для створення UML-моделей:

* StarUML;
* ArgoUML;
* Umbrello.

StarUML – редактор який доступний на Windows і Linux, а також має як безкоштовну так і комерційну версію. Крім того, StarUML дозволяє нам розширити наші можливості за допомогою плагінів, які дозволяють додавати в програму додаткові мови програмування (PHP, С ++, C #, VB, тощо), автоматично створювати шаблони завдяки вбудованому помічникові і, врешті-решт, підтримує створення більшості UML-діаграм. Редактор підтримує усі стандарти UML і оновлюється відповідно до нових (2.0).

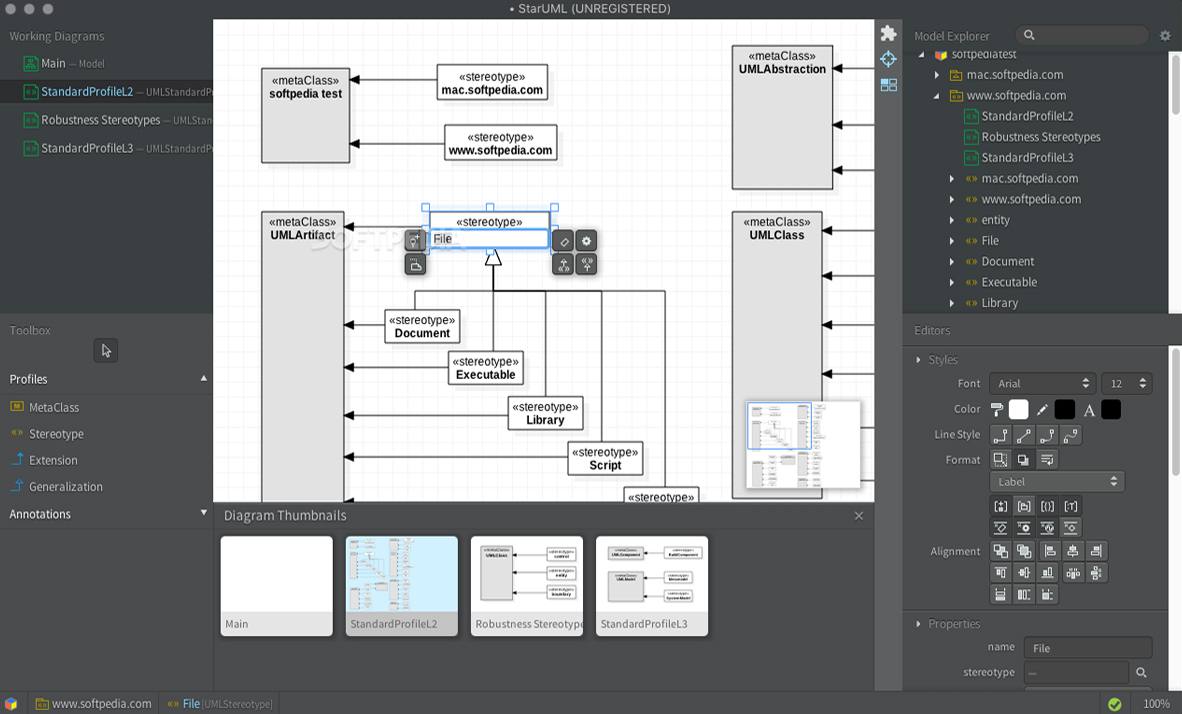


Рисунок 1.5 – Інтерфейс програми StarUML

ArgoUML – є відкритим програмним забезпеченням і розповсюджується під ліцензією EPL. Функціональність ArgoUML включає в себе: підтримку специфікацій UML (1.3, 1.4, XMI 1.0, 1.1, 1.2), 9 видів діаграм UML (діаграми класів, станів, кооперації, послідовності, діяльності, прецедентів, об'єктів, компонентів, розгортання), підтримку OCL для класів, генерацію вихідного коду (Java, C ++, C # і PHP).

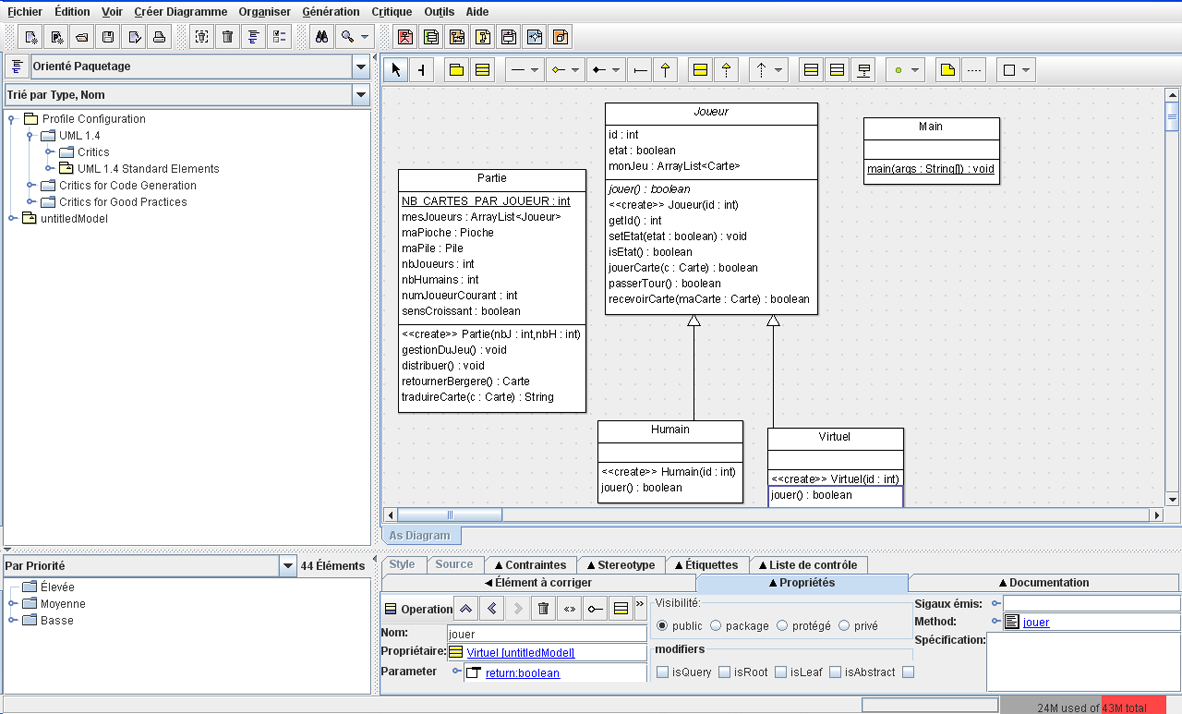


Рисунок 1.6 – Інтерфейс програми ArgoUML

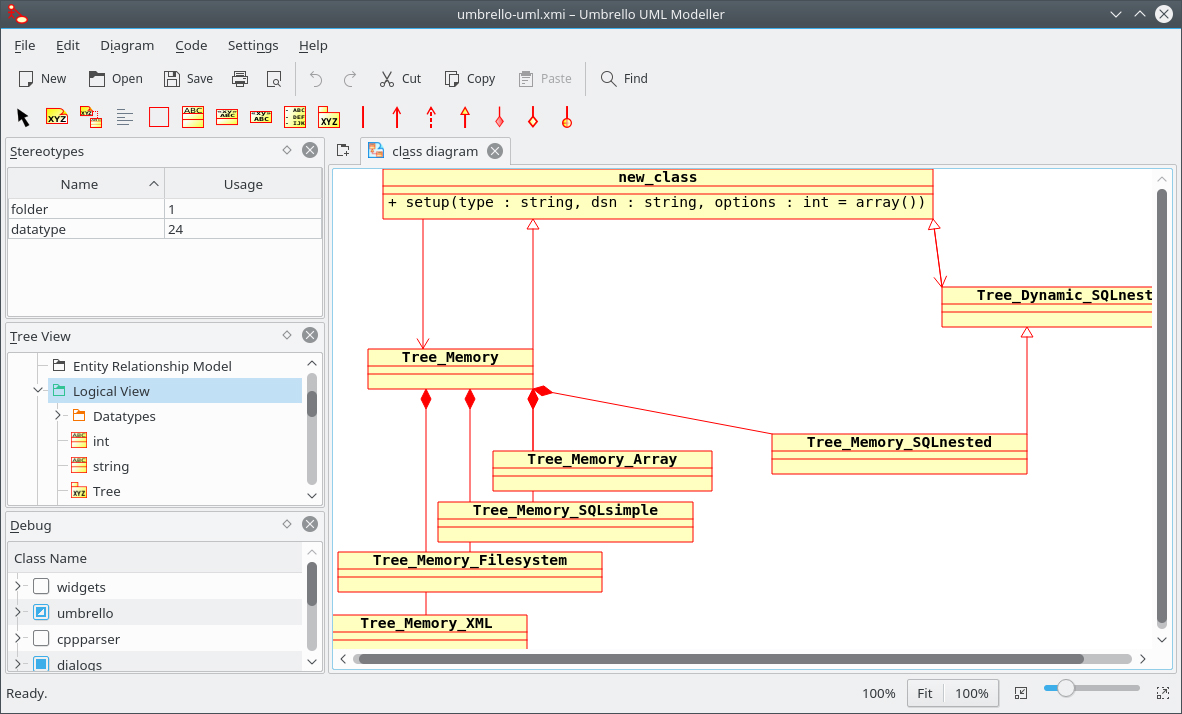


Рисунок 1.3 – Інтерфейс програми Umbrello

Umbrello – ця програма є вільним програмним забезпеченням, призначеним для побудови UML діаграм на платформі Unix. Umbrello підтримує всі стандартні типи UML-діаграм. Також підтримується імпорт з C++, PHP, IDL, Pascal/Delphi, Ada, Python, Java, Perl та експорт діаграм у різні мови програмування. Umbrello дозволяє зберігати дані моделі в форматах DocBook і XHTML. Це було зроблено з метою підтримки моделі спільної розробки коли не всі розробники мають доступ до Umbrello. Ця можливість також незамінна при необхідності розміщення контенту моделі на веб-сайті.

Для більш зручного порівняння виокремимо основні характеристики засобів у таблиці 1.2.

Таблиця 1.2 – Порівняння UML редакторів

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика | StarUML | ArgoUML | Umbrello |
| Графічні інструменти | + | +- | + |
| Простота використання | + | +- | +- |
| Генерація коду | + | +- | + |
| Експорт | + | - | + |
| Імпорт | + | - | + |
| Вибір діаграм | + | +- | +- |
| Підтримка та оновлення редактору | + | - | +- |

Відповідно до наведеного вище порівняння було обрано StarUML, оскільки він надає максимальні можливості для користувача, пропонуючи налаштування параметрів, які визначають методологію створення програмного забезпечення, проектну платформу і мову розробки, підтримується розробником та має регулярні оновлення, і найбільш відповідає стандартам UML.

## Власний варіант реалізації

Варіант реалізації SHome представляє з себе систему розумного контролю мікрокліматичних параметрів з вбудованими функціями енергозбереження та прогнозу енергоспоживання. Система відноситься до класу з вбудованим центральним контролером. Архітектура модульного типу. Розширюваність забезпечується за рахунок сумісних c Arduino модулів, однак для підхоплення нових сенсорів необхідно завантажити нові налаштування на Arduino. Система складається з трьох частин: модулі керування кліматотехніки на основі Arduino, центральний контролер і мобільний додаток для управління системою та відображення інформації у візуально-зрозумілому вигляді. Відмінність від інших рішень полягає у використанні тільки вільних компонентів під вільною ліцензією як у випадку з апаратним забезпеченням, так і з програмним. Абсолютно будь-який користувач може налаштувати систему під себе, використовуючи недорогі компоненти для Arduino і малопотужний пристрій в якості центрального контролера, не написавши при цьому жодного рядка коду.

Основна мета проекту - надати користувачам недорогу систему на безоплатній основі з просунутими можливостями в налаштуванні енергетичних планів і кастомізації. Необхідне для розгортання реалізації від SHome на рис. 1.5.



Рисунок 1.5 – Все необхідне для розгортання системи SHome

Основні характеристики:

• Взаємодія

• Масштабованість

• Віддалений доступ

Основні риси власної системи SHome:

• Енергозбереження

• Енергопрогнозування

• Компоненти під вільною ліцензією

• Кастомізація

# РОЗДІЛ 2. АПАРАТНЕ ЗАБЕСПЕЧЕННЯ



## Arduino та Arduino Uno

Реалізація концепції "розумного" будинку досить складна задача. Для побудови та тестування такої системи необхідна значна кількість різних датчиків, модулів та пристроїв. Розглянемо компоненти, необхідні для виконання поставленої задачі.

Мікроконтролер або мікрокомп’ютер - один з центральних елементів системи "розумного" будинку. Він відповідає за керування різними пристроями навантаження, такими як обігрівачі, освітлення, установки зволоження повітря та інші, а також знімає показники з різноманітніших датчиків. Розглянемо деякі варіанти мікроконтролерів та мікрокомп’ютерів.

Arduino – апаратна обчислювальна платформа, основними компонентами якої є плата вводу/виводу та середовище розробки на мові Processing/Wiring. Arduino може використовуватися як для створення автономних інтерактивних об'єктів, так і підключатися до програмного забезпечення, яке виконується на комп'ютері (наприклад: Adobe Flash, Processing, Max/MSP, Pure Data, SuperCollider).

Arduino Uno є плата мікроконтролера на базі ATmega328. Його зображення наведено на рис. 2.1 та рис. 2.2, характеристики у таблиці 2.1

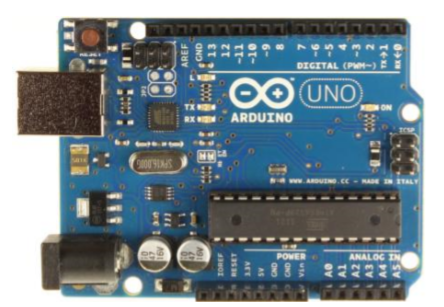


Рисунок 2.1 – Arduino UNO вид зверху

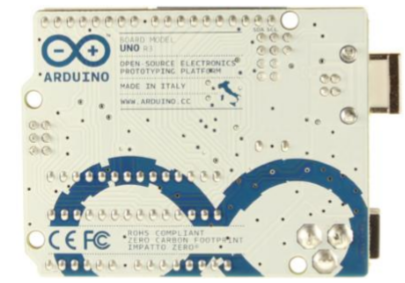


Рисунок 2.2 – Arduino UNO вид знизу

Таблиця 2.1 – Характеристики Arduino UNO

|  |  |
| --- | --- |
| **Параметр** | **Значення** |
| Мікроконтролер | ATmega328 |
| Робоча напруга | 5В |
| Вхідна напруга (рекомендований) | 7-12В |
| Вхідна напруга (граничне) | 6-20В |
| Цифрові Входи / Виходи | 14 (з них 6 можуть використовуватися в якості ШІМвиходів) |
| Аналогові входи | 6 |
| Постійний струм через вхід / вихід | 40 mA |
| Постійний струм для виведення 3.3 В | 50 mA |
| Флеш-пам'ять | 32 КБ з яких 0.5 КБ використовуються завантажувачем |
| ОЗУ | 8 KB |
| Незалежна пам'ять | 4 KB |
| Тактова частота | 16 MHz |

Вартість плати на сайті Arduino.ua **1560.00 UAH**

## Розробка моделі програмного комплексу

UML діаграмі класів використовуються такі позначення специфікаторів доступу для атрибутів та операцій:

* Публічні (public) “+”;
* Захищені (protected) “#”;
* Приватні (private) “-“.

Класи пов’язуються між собою одним з нижче наведених відношень:

* Узагальнення - це наслідування. Це відношення добро розглядається в кожному підручнику будь-якої ООП мови. Позначається лінією з стрілкою-трикутником;
* Асоціація показує, що об'єкти однієї сутності (класу) пов'язані з об'єктами іншої сутності таким чином, що можна переміщатися від об'єктів одного класу до іншого. Є загальним випадком композиції і агрегації. Зображується у вигляді непереривної стрілки;
* Агрегація - форма асоціації, що описує відношення частина - ціле (між цілим і його частинами). Агрегація представляє собою сильну взаємодію між об'єктами класів. Агрегація зображується в UML у вигляді безперервної лінії, що з'єднує класи. Кінець лінії, що підходить до класу-цілому, зображується у вигляді не зафарбованого ромба;
* Залежність - показує відношення між клієнтом і постачальником, при цьому клієнт не знає про існування постачальника. Залежність зображується в UML у виді пунктирної стрілки, спрямованої від клієнта до постачальника;
* Реалізація, як і узагальнення має відношення до ООП: оголошення інтерфейсу и можливість його реалізації будь-якім класом. Графічно реалізація представляється так само, як і успадкування, але з пунктирною лінією;
* Композиція - більш суворий варіант агрегації. Відома також як агрегація за значенням. Композиція має жорстку залежність існування примірників класу контейнера і примірників агрегованих класів. Якщо контейнер буде знищений, то весь його вміст буде також знищено. Графічно представляється як і агрегація, але з зафарбованим ромбом.

Виходячи з даних відомостей, було побудовано діаграму класів, зображену на рисунку 2.2.

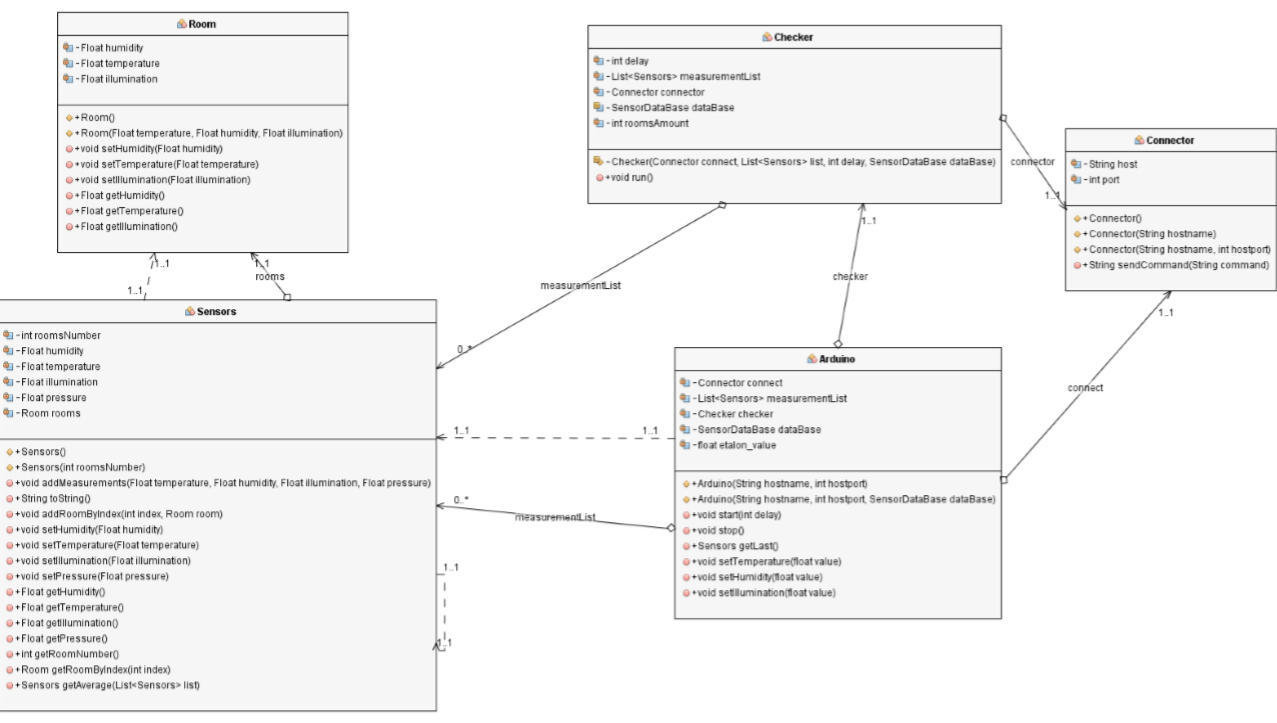


Рисунок 2.2 - Діаграма класів з патернами проектування

Як було згадано раніше при проектуванні були використані патерни. Патерн проектування — це типовий спосіб вирішення певної проблеми, що часто зустрічається при проектуванні архітектури програм.

На відміну від готових функцій чи бібліотек, патерн не можна просто взяти й скопіювати в програму. Патерн являє собою не якийсь конкретний код, а загальний принцип вирішення певної проблеми, який майже завжди треба підлаштовувати для потреб тієї чи іншої програми.

Ви можете цілком успішно працювати, не знаючи жодного патерну. Більше того, ви могли вже не раз реалізувати який-небудь з патернів, навіть не підозрюючи про це. Отже, навіщо ж знати патерни?

* Перевірені рішення;
* Стандартизація коду;
* Загальний словник програмістів.

Патерни відрізняються за рівнем складності, деталізації та охоплення проектованої системи. Розглянемо основні групи патернів:

* Породжуючі патерни піклуються про гнучке створення об’єктів без внесення в програму зайвих залежностей;
* Структурні патерни показують різні способи побудови зв’язків між об’єктами;
* Поведінкові патерни піклуються про ефективну комунікацію між об’єктами.

## Arduino Mega 2560

Arduino Mega 2560 мікроконтролер на основі процесора ATmega2560. Його зображення наведено на рис. 2.3 та рис. 2.4, характеристики у таблиці 2.

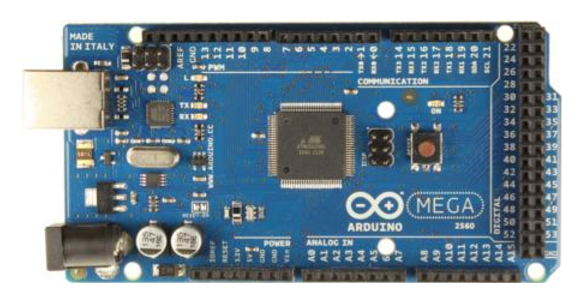


Рисунок 2.3 – Arduino Mega 2560 R3 вид зверху

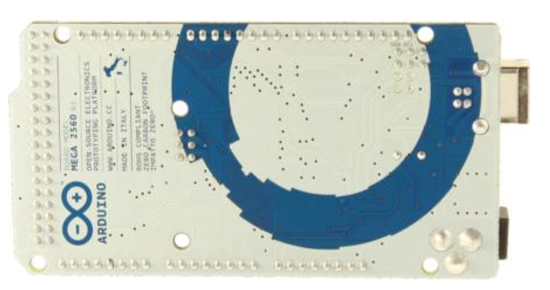


Рисунок 2.4 – Arduino Mega2560 R3 вид знизу

Таблиця 2.2 – Характеристики Arduino Mega2560 R3

|  |  |
| --- | --- |
| **Параметр** | **Значення** |
| Мікроконтролер | ATmega2560 |
| Робоча напруга | 5В |
| Вхідна напруга (рекомендований) | 7-12В |
| Вхідна напруга (граничне) | 6-20В |
| Цифрові Входи / Виходи | 54 (з них 14 можуть використовуватися в якості ШІМ виходів) |
| Аналогові входи | 16 |
| Постійний струм через вхід / вихід | 40 mA |
| Постійний струм для виведення 3.3 В | 50 mA |
| Флеш-пам'ять | 256 КБ з яких 8 КБ використовуються завантажувачем |
| ОЗУ | 8 KB |
| Незалежна пам'ять | 4 KB |
| Тактова частота | 16 MHz |

# РОЗДІЛ 3. Розробка програмного забезпечення



## Програмне забезпечення Arduino

Для реалізація роботи мікроконтролера а також взаємодії з ним необхідно розробити відповідне програмне забезпечення. Потрібно створити:

* + Програмне забезпечення, яке буде виконуватися на Arduino
  + Програмне забезпечення для комунікації між Android та Arduino
  + Базу даних та методи роботи з нею
  + Файл для створення та редагування файлів конфігурацій для мікроконтролера Розглянемо поетапно ці завдання.

Для роботи мікроконтролера необхідно розробити ПЗ, яке буде виконувати наступні функції:

* + Слідкувати за значеннями сенсорів
  + Включати та виключати пристрої
  + Взаємодіяти з Сервером через протокол TCP/IP
  + Зчитувати файл конфігурацій з карти пам’яті та виконувати відповідні налаштування

Для розробки ПЗ скористаємось офіційним середовищем розробки від Arduino. Нам знадобляться наступні бібліотеки:

* + SPI.h
  + SD.h
  + Ethernet.h
  + DHT.h
  + stdio.h
  + string.h
  + SFE\_BMP180.h
  + Wire.h
  + OneWire.h
  + DallasTemperature.h

Бібліотека SPI.h необхідна для роботи з SD Card Reader та W5100 Ethernet Shield.

Бібліотека SD.h відповідає за роботу з SD Card Reader.

Бібліотека Ethernet.h використовується для взаємодії з W5100 Ethernet Shield.

Бібліотека DHT.h необхідна для зв’язку з датчиками сімейства DHT. Бібліотеки stdio.h та string.h використовуються для роботи зі строками та стандартними функціями.

Бібліотека SFE\_BMP180.h відповідає за взаємодію з датчиком барометричного тиску BMP180.

Бібліотеки OneWire.h та DallasTemperature.h необхідні для роботи з датчиками DS1820/DS18B20/DS18S20.

Передача команд мікроконтролеру та його відповідей буде відбуватися через протокол TELNET, що дозволить легко тестувати роботу системи без необхідності використовувати додаткове ПЗ.

Призначення протоколу TELNET у наданні достатньо спільного, двонаправленого, восьмибітового байт-орієнтованого засобу зв'язку. Його основне завдання полягає в тому, щоб дозволити термінальним пристроям і термінальним процесам взаємодіяти один з одним. Передбачається, що цей протокол може бути використаний для зв'язку виду термінал-термінал ("зв'язування") або для зв'язку процес-процес ("розподілені обчислення"). Програмний код скетча включає всього 4 функції:

* run\_config()
* rewrite\_etalon()
* setup()
* loop()

Функції будемо намагатися використовувати менше, щоб отримати максимальну швидкість роботи мікроконтролера. З цієї ж самою причини відмовимось від використання ООП, на скільки це можливо.

## Програмне забезпечення бази даних

Для роботи систем прогнозування, а також для оптимізації збереження інформацій, спрощення доступу до неї необхідно реалізувати базу даних. Це дозволить легко виконувати збереження даних у хмарі, імпорт та експорт даних. Для реалізації цієї задачі доцільним буде скористатися можливостями програмування для Android, а саме базою даних на мові SQLite.

Використання SQLite дозволяє зберігати базу даних локально у директорії програми. Для експорту та імпорту достатньо буде просто отримати доступ до файлу и виконати переписування або зчитування файлу. Для реалізації роботи з базо даних на Android було створено пакет SensorDB. Він складається з наступних класів:

* SensorDataBaseHelper
* SensorDataBase

UML діаграма класів наведена на рис. 3.2

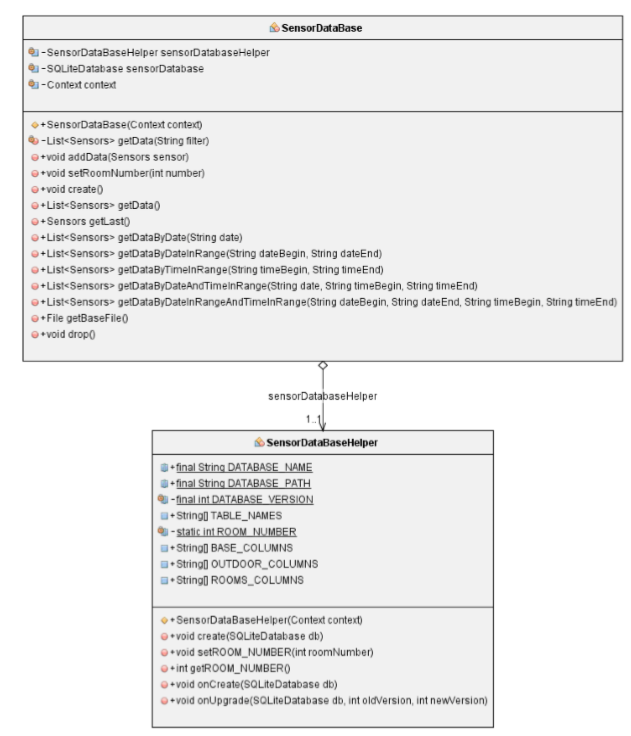


Рисунок 3.2 – UML діаграма класів для роботи з БД

Для зручності роботи з даними необхідно розробити власне саму базу даних. Створимо базу даних 3 нормальної форми. У відповідності до 3NF сформуємо по таблиці на кожну кімнату, таблицю для вуличних вимірювань та базову таблицю. При цьому у базовій таблиці будемо мати посилання на таблиці кімнат та зовнішніх вимірів. Діаграма бази даних для випадку 1 кімнати наведена на рис 3.3

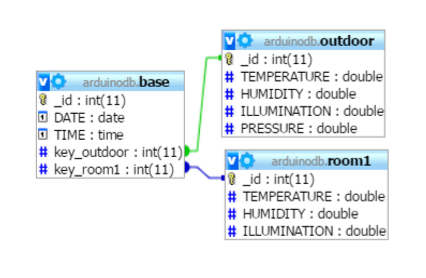


Рисунок 3.3 – Схема БД

Базова таблиця outdoor складається з таких стовбців:

* \_id
* DATE
* TIME
* key\_outdoor
* key\_room1

Стовпець \_id є унікальним ідентифікатором запису у таблиці.

У полі DATE будемо зберігати дату, коли було зроблено запис.

У полі TIME зберігаємо час запису.

Поле key\_outdoor є зовнішнім посиланням на запис у таблиці outdoor.

Поле key\_room є зовнішнім посиланням на запис у таблиці room1.

Таблиця outdoor складається з наступних стовпців:

* \_id
* TEMPERATURE
* HUMIDITY
* ILLUMINATION
* PRESSURE

Стовпець \_id є унікальним ідентифікатором запису у таблиці. У полі TEMPERATURE зберігається значення температури на вулиці. У полі HUMIDITY зберігається значення вологості на вулиці. У полі ILLUMINATION зберігається значення освітленості на вулиці. У полі PRESSURE зберігається значення атмосферного тиску на вулиці. Таблиця room1 має такі стовпці:

* \_id
* TEMPERATURE
* HUMIDITY
* ILLUMINATION

Стовпець \_id є унікальним ідентифікатором запису у таблиці.

У полі TEMPERATURE зберігається значення температури.

У полі HUMIDITY зберігається значення вологості.

У полі ILLUMINATION зберігається значення освітленості.

## [Генерування програмного коду для прототипу програми](#_3rdcrjn)

На основі діаграми класів, яка була побудована раніше можна виконати кодогенерацію, яка згенерує файли класів спроектованого веб-сайту. Зазначимо, що кодогенерація – це процес автоматичного перетворення діаграм UML у код програми.

Для проведення генерації коду програми необхідна наявність діаграми класів та діаграми компонентів, у якій компоненти мають мати визначену мову програмного коду, бо від мови реалізації залежить послідовність кроків генерації коду.

Перед проведення генерації коду необхідно адаптувати усі елементи під обрану мову програмування, наприклад змінити усі використані типи даних на відповідні до обраної мови програмування. У нашому випадку було використано мову програмування C#.

Для генерації C# у StarUML необхідно виконати такий порядок дій:

1. Відкрити файл з створеною раніше діаграмою класів;
2. Перейти у StarUML – Tools – Extension Manager та встановити розширення C#;
3. Перезавантажити StarUML;
4. Перейти у StarUML – Tools – C# – Generate code та обрати класи код яких потрібно згенерувати;
5. Згенерований код з’явиться у обраній вами папці, яку запропонує обрати редактор, на цьому генерація коду завершена.

Також порядок генерації коду наведено на рисунку 3.4, 3.5 та 3.6.

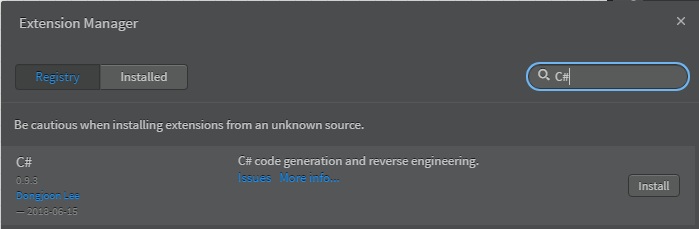


Рисунок 3.4 – Встановлення розширення C# у StarUML

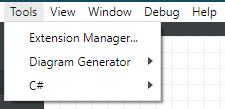


Рисунок 3.5 – Перехід до генерації коду в C#

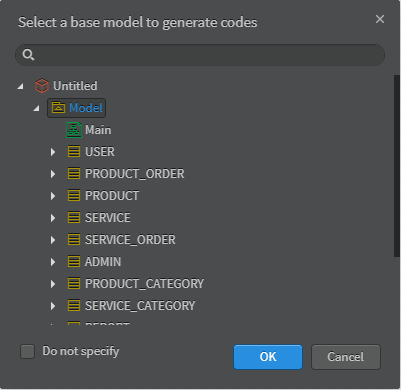


Рисунок 3.6 – Вибір класів код яких потрібно згенерувати

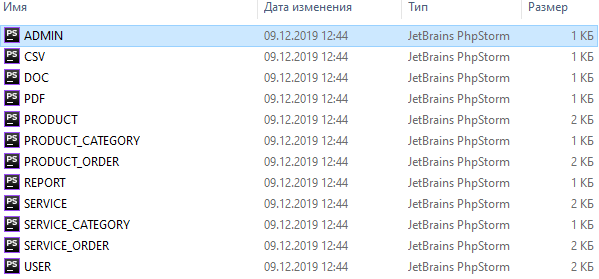


Рисунок 3.7 – Файли згенерованих класів

# ВИСНОВКИ

На сьогоднішній день концепція "розумного" будинку є дуже актуальна. Вона дозволяє користувачам економити енергоносії та створювати для себе комфортний мікроклімат у приміщені, тому важливість цього питання є дуже високою, особливо для нашою країни в її складному економічному та енергетично залежному становищі.

Для контролю параметрів навколишнього середовища, а також мікроклімату приміщення був визначений перелік розповсюджених сенсорів. Також було наведені характеристики цих сенсорів. Такий підхід дозволяє користувачеві самому визначити параметри для контролю, та побудувати систему відповідно до його вподобань. Для реалізації системи управління виконавчими пристроями був розроблений ПЗ для мікроконтролера. Головною його особливістю є гнучка можливість керування мікроконтролером, а також можливість зміни конфігурації системи без переписування коду. Для цього достатньо лише завантажити новий файл конфігурації на карту пам’яті та перезавантажити мікроконтролер.

У процесі виконання досліджень була розроблена UML модель веб-сайту для “розумного” будинку.

В ході її розробки було створено діаграми, що входять до мови моделювання UML. А саме: діаграма прецедентів, що відображає відносини між акторами та прецедентами які вони можуть виконувати, діаграма діяльності яка моделює процес виконання операцій, діаграма класів та компоненти, котрі відображають взаємодію класів та компонентів у цілому та дозволяють згенерувати код який буде основою програмної системи, та діаграма послідовності на якій відображено взаємодію об’єктів системи.

Також було застосовано патерни проектування які можна назвати типовими способами вирішення однієї і тієї ж проблеми, що часто повторюються з невеликими варіаціями під час проектування та розробки програмного забезпечення.

Відповідно виконаних завдань було вивчено особливості моделювання та аналізу програмних комплексів, що і було метою досліджень курсової роботи.

# ЛІТЕРАТУРА

1. ABOUT THE UNIFIED MODELING LANGUAGE SPECIFICATION VERSION 2. URL: https://www.omg.org/spec/UML/2.0/;
2. Рефакторинг.Гуру присвячений темним матеріям програмування — рефакторингу, патернам проектування, принципам SOLID та іншим важливим темам зі світу програмування. URL: https://refactoring.guru/uk;
3. UML Activity Diagram Tutorial. URL: https://www.lucidchart.com/pages/uml-activity-diagram;
4. Нотація і семантика мови UML: Інформація. URL: https://www.intuit.ru/studies/courses/32/32/info;
5. UML: історія, Специфікація, Бібліографія. УРЛ: http://it.ridne.net/node/265;
6. UML Sequence Diagram Tutorial: URL: https://www.lucidchart.com/pages /uml-sequence-diagram;
7. Діаграма варіантів використання (USE case diagram). URL: https://studopedia.ru/19\_284009\_diagrama-variantiv-vikoristannya-USE-case-diagram.html;
8. Відносини класів - від UML до коду. URL: https://habr.com/post/150041/;
9. CASE-засоби. URL: http://easy-code.com.ua/2012/08/case-zasobi-zagalna-xarakteristika-i-klasifikaciya-komerciya-rizne-statti/;
10. Уніфікована мова моделювання UML. Загальна характеристика. URL: https://svitppt.com.ua/informatika/unifikovana-mova-modelyuvannya-uml-zagalna-harakteristika.html;
11. Проектування інформаційних системи на основі уніфікованої мови моделювання. URL: https://elearning.sumdu.edu.ua/free\_content/lectured: de1c9452f2a161439391120eef364dd8ce4d8e5e/20151203140326/204841/index.html;
12. StarUML documentation: Introduction. URL: https://docs.staruml.io.
13. Lutron Electronics, Inc. - Dimmers And Lighting Controls. – Режим доступу : http://www.lutron.com/en-US/Company-Info/Pages/AboutUS/OurStory.aspx .
14. Pico Electronics. – Режим доступу : <http://www.picodenshi.com/about-pico30238.html>.
15. HomeKit - Apple Developer.
16. Products - Clipsal by Schneider Electric. – Режим доступу : <https://www.clipsal.com/Home-Owner>
17. Arduino Uno. – Режим доступу : http://arduino.ua/ru/hardware/Uno .
18. Arduino Mega 2560. – Режим доступу http://arduino.ua/ru/hardware/Mega2560 .