

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ ТА ТЕЛЕКОМУНІКАЦІЙ

УДК 621.37:621.391

«Допущено до захисту»

Завідувач кафедри біомедичної
інженерії та телекомунікацій
к.т.н., доц. Ніятчук Т.М.

«13» 12 2021 р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему: Дослідження можливості побудови портативного вимірювача
ємності на базі платформи Arduino

за спеціальністю 172 «Телекомунікації та радіотехніка»
освітній ступінь «магістр»

Виконавець роботи
студент 2 курсу, гр. ТРм-20-1

13.12.2021

(дата, підпис)

Літинський Д.С.

Керівник роботи
к.т.н., доц.

13.12.2021

(дата, підпис)

Чухов В.В.

Житомир
2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ ТА ТЕЛЕКОМУНІКАЦІЙ

Пояснювальна записка
до кваліфікаційної магістерської роботи

на тему: Дослідження можливості побудови портативного вимірювача
ємності на базі платформи Arduino

Виконав: студент 2 курсу,
групи ТРм-20-1 спеціальності
172 «Телекомунікації та радіотехніка» _____

Літинський Д.С. _____

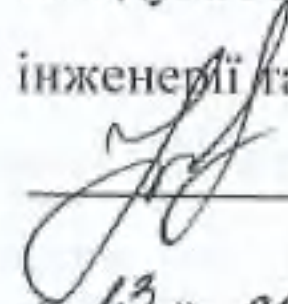
Керівник _____ Чухов В.В. _____

Рецензент _____ Ципоренко В.В. _____

Житомир
2021

«Затверджено»

Завідувач кафедри біомедичної
інженерії та телекомунікацій

 Т.М. Нікітчук

«13» січня 2021 р.

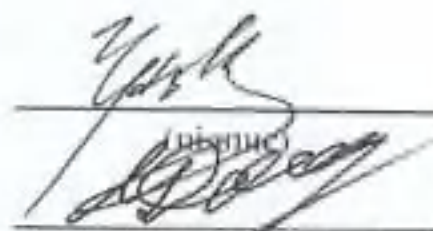
Завдання на кваліфікаційну магістерську роботу

Літинського Дмитра Сергійовича

1. Тема роботи Дослідження можливості побудови портативного вимірювача ємності на базі платформи Arduino
затверджено наказом по університету від «08» 11 2021 р. № 546-с
2. Мета дослідження оцінка можливості побудови портативного вимірювача ємності на базі платформи Arduino
3. Вихідні дані (характеристики об'єкта, умов дослідження та ін. типові моделі конденсаторів, вимірювання на низьких частотах за кімнатної температури
4. Основні задачі дослідження аналіз літературних джерел, розробка математичної моделі об'єкта дослідження, аналіз його точнісних можливостей
5. Термін подання роботи до захисту грудень 2021 р.
6. Дата видачі завдання «01» 01 2021 р.

Науковий керівник

Завдання прийняв
до виконання


(підпис)

Чухов В.В.
(прізвище, ініціали)
Літинський Д.С.
(прізвище, ініціали)

АНОТАЦІЯ

Кваліфікаційну роботу магістра присвячено дослідженню можливості побудови портативного вимірювача ємності на базі платформи Arduino.

Підтверджено можливість побудови такого вимірювача, підтвердженням чого діючий прототип моделі під управлінням плати Arduino UNO R3 для вимірювання ємності конденсатора. Виміряне значення ємності на LCD дисплеї 16x2.

ABSTRACT

The master's thesis is devoted to the study of the possibility of building a portable capacity meter based on the Arduino platform.

The possibility of constructing such a meter has been confirmed, as evidenced by the current prototype of the model under the control of the Arduino UNO R3 board for measuring the capacitance of the capacitor. Measured capacity value on 16x2 LCD display.

РЕФЕРАТ

Кваліфікаційну роботу магістра присвячено дослідженню можливості побудови портативного вимірювача ємності на базі платформи Arduino.

Підтверджено можливість побудови такого вимірювача, підтвердженням чого діючий прототип моделі під управлінням плати Arduino UNO R3 для вимірювання ємності конденсатора. Виміряне значення ємності на LCD дисплеї 16x2.

Робота складається з вступу, п'ятих розділів, висновків, переліку використаних джерел та двох додатків.

У вступі обґрунтовано актуальність проекту та сформульовано мету, об'єкт та предмет проекту.

У першому розділі були наведені загальні відомості про конденсатор, загальні формули та методи вимірювання ємностей.

У другому розділі розглянуто алгоритм роботи портативного вимірювача ємності.

У третьому розділі була розглянута мікроконтролерна частина випускної кваліфікаційної роботи та модулі які будуть використані для проектування.

У четвертому розділі розглянуті програмні засоби для створення системи на основі Arduino.

У п'ятому розділі написано скетч який вимірює ємність конденсатора та виводить результат на дисплей.

ЗМІСТ

ВСТУП	8
РОЗДІЛ 1 Аналіз конденсаторів та використаних елементів, методи та формули вимірювання ємності	9
1.1 Загальні відомості про конденсатор та формули вимірювання ємності .	9
1.2 Характеристики та властивості конденсаторів	12
1.3 Класифікація конденсаторів	14
1.4 Мікроконтролерні системи	16
1.5 Методи вимірювання ємностей конденсаторів.....	18
Висновки	20
РОЗДІЛ 2 Алгоритм роботи портативного вимірювача ємності	21
2.1 Блок схема алгоритму роботи вимірювача ємності	21
2.2 Структурна схема вимірювача ємності на arduino	22
2.3 Функціональна схема вимірювача ємності на Arduino	22
2.4 Принципова схема вимірювача ємності на arduino	23
Висновки	24
РОЗДІЛ 3 Розробка портативного вимірювача ємності.....	25
3.1 Вибір платформи Arduino Uno R3.....	25
3.2 Загальні відомості ЖК-контролер Hitachi HD44780	32
3.3 Генератор прямокутних імпульсів на мікросхемі таймера 555.....	34
3.4 Тригер Шмітта.....	36
Висновки	38
РОЗДІЛ 4 Програмні засоби для написання коду вимірювача ємності на базі платформи arduino	39
4.1 Arduino IDE.....	39
4.2 Структура програми в Arduino IDE.....	43
4.3 Як Arduino вимірює ємність, функція pulseIn()	45
Висновки	46
РОЗДІЛ 5 Програмування arduino uno r3 та перегляд результатів	47

5.1 Написання коду скетчу для вимірювача ємності.....	47
5.2 Написання коду скетчу для відображення результату на ЖК-контролер Hitachi HD44780	48
5.3 Тестування та результати роботи портативного вимірювача ємності ..	49
Висновки	50
ВИСНОВОК	51
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ	52
ДОДАТОК А	53
ДОДАТОК В	54

ВСТУП

Актуальність теми. Для проектування та створення будь яких аналогових або цифрових пристроїв використовуються різні електронні елементи із схемотехніки, наприклад резистори, транзистори, індикатори, світлодіоди, конденсатори та багато інших елементів.

Усі вони мають різні характеристики, класифікації та параметри які залежать від їхнього призначення та особливостей використання у різних принципових електричних схемах.

У даній роботі досліджується можливість побудови портативного вимірювача ємності на базі платформи Arduino.

Об'єктом дослідження є процес вимірювання ємності конденсаторів.

Предметом дослідження є метод і засіб вимірювання ємності конденсаторів..

Мета дослідження – оцінка можливості побудови портативного вимірювача ємності на базі платформи Arduino.

Для досягнення мети необхідно розв'язати такі задачі та виконати наступні роботи:

1. Провести аналіз літературних джерел та визначити основні методи вимірювання ємності конденсаторів, їх переваги та недоліки.
2. Провести розробку математичної моделі роботи портативного вимірювача ємності.
3. Провести дослідження характеристик портативного вимірювача ємності.

Практичні результати:

1. Розроблено математичний апарат моделювання оптичного нахиломіра.
2. Проведено дослідження точності вимірювань нахиломіра, що працює на основі алгоритму зсуву фази.

РОЗДІЛ 1

АНАЛІЗ КОНДЕНСАТОРІВ ТА ВИКОРИСТАНИХ ЕЛЕМЕНТІВ, МЕТОДИ ТА ФОРМУЛИ ВИМІРЮВАННЯ ЄМНОСТІ

1.1 Загальні відомості про конденсатор та формули вимірювання ємності

Конденсатор — система з двох чи більше електродів (обкладок), які розділені діелектриком, товщина якого менша у порівнянні з розміром обкладок. Така система має взаємну електричну ємність і здатна зберігати електричний заряд.

Конденсатор є пасивним електронним компонентом і широко застосовується в електронних схемах для блокування постійного струму, пропускаючи змінний струм.

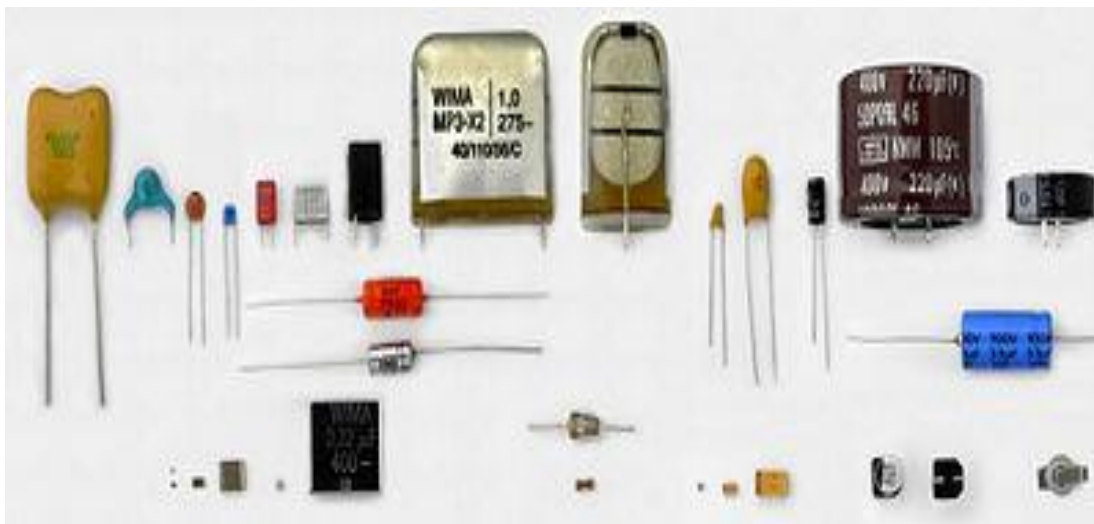


Рисунок 1.1 - Різні види конденсаторів.

Прикладання електричної напруги до обкладок конденсатора спричиняє накопичення на них електричного заряду. Після відключення від джерела напруги, заряд утримується на обкладках силами електростатики.

Якщо конденсатор, як цілісний елемент, не є наелектризованим, то заряд,

що накопичений на обох обкладках є однаковим за величиною і протилежний за знаком.

Здатність конденсатора накопичувати заряд характеризує його електрична ємність:

$$C = \frac{Q}{U} \quad (1.1)$$

де: C — ємність конденсатора у фарадах;

Q — електричний заряд, що накопичений на одній з обкладок в кулонах;

U — електрична напруга між обкладками у вольтах.

Ємність виражається у фарадах. Одна фарада є досить значною одиницею, тому на практиці ємність конденсаторів виражається у піко-, нано-, мікро- та міліфарадах.

У загальному випадку, напруга U_C і електричний струм I_C конденсатора у момент часу t пов'язані залежністю:

$$U_C = \frac{Q}{C} = \frac{1}{C} \int_{-\infty}^t I_C(\tau) d\tau \quad (1.2)$$

Робота dW , яку слід виконати, щоб перенести елементарний заряд dq з однієї обкладки конденсатора ємності C , на іншу, при допущенні, що одна з обкладок містить заряд з поточним значенням q .

$$dW = U(q)dq = \frac{q}{C}dq \quad (1.3)$$

Енергію, яка накопичена в конденсаторі можна визначити інтегруванням рівняння, записаного вище з отриманням виразу:

$$W = \int_0^Q \frac{q}{C} dq = \frac{1}{2} \frac{Q^2}{C} = \frac{1}{2} C U_C^2, \quad (1.4)$$

де: Q — початкове значення заряду конденсатора.

Зміну величини заряду конденсатора у часі характеризує електричний струм у момент заряджання, на основі чого можна записати:

$$I_c(t) = \frac{dQ}{dt} = C \frac{dU_c}{dt} \quad (1.5)$$

Конденсатор у колі постійної напруги після того, як він зарядиться не проводить струм, оскільки його обкладки розділені діелектриком. У ланцюгу зі змінною напругою він проводить електричний струм, оскільки коливання змінного струму викликають циклічне перезарядження конденсатора, а тому і струм у ланцюгу, що описується рівняннями:

$$\begin{aligned} U_c(t) &= U_0 \sin(\omega t) \\ I_c &= C \frac{dU_c}{dt} = CU_0 \omega \cos(\omega t) \end{aligned} \quad (1.6)$$

Величина, що пов'язує струм і напругу на конденсаторі, називається реактивним опором, котра є тим меншою, чим більшою є ємність конденсатора і частота струму. Для конденсатора характерним є те, що для синусоїдального закону зміни струму, зміна напруги відстає за фазою на $\frac{\pi}{2}$ (тобто струм випереджає напругу за фазою на кут $\frac{\pi}{2}$).

З цієї точки зору імпеданс конденсатора є комплексним числом і описується рівнянням:

$$Z = \frac{1}{i\omega C} = \frac{-i}{\omega C} = \frac{-i}{2\pi f C} \quad (1.7)$$

де: ω — кутова частота;

f — частота в герцах;

i — уявна одиниця

Реактивний опір ємнісного опору записується рівнянням:

$$X_c = \frac{-1}{\omega C} = \frac{-1}{2\pi f C} \quad (1.8)$$

Відповідно, для постійного струму частота дорівнює нулю, а опір конденсатора — нескінченна величина (в ідеальному випадку).

При зміні частоти змінюється діелектрична проникність діелектрика і рівень впливу паразитних параметрів — власної індуктивності і опору втрат.

На високих частотах будь-який конденсатор можна розглядати як

послідовний коливальний контур, утворений ємністю C , власною індуктивністю L_C і опором втрат R_n .

$$f_p = 1/2\pi\sqrt{L_c C} \quad (1.9)$$

При $f > f_p$ конденсатор в колі змінного струму поводить себе як котушка індуктивності. Відповідно, конденсатор доцільно використовувати лише на частотах $f < f_p$, на яких його опір має ємнісний характер.

1.2 Характеристики та властивості конденсаторів

Основною характеристикою конденсатора є його електрична ємність (точніше номінальна ємність), яка визначає накопичений заряд. Типові значення ємності конденсаторів складають від одиниць пікофарад до сотень мікрофарад. Але існують конденсатори з ємністю десятків фарад.

Ємність плоского конденсатора, яка складається з двох паралельних металічних пластин площиною S кожна, які розташовані на відстані d одна від одної, в системі СІ виражена формулою $C = \frac{\varepsilon\varepsilon_0 S}{d}$, де ε — відносна діелектрична проникність середовища, яке заповнює простір між пластинами. Ця формула справедлива лише при малих d .

Для отримання великих ємностей конденсатори з'єднують паралельно. Загальна ємність батареї паралельно з'єднаних конденсаторів дорівнює сумі ємностей всіх конденсаторів, які входять у батарею.

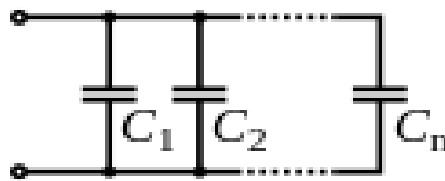


Рисунок 1.2 - Паралельне з'єднання конденсаторів.

$$C = \sum_{i=1}^n C_i \quad (1.10)$$

При послідовному з'єднанні конденсаторів заряди усіх конденсаторів однакові. Загальна ємність батареї послідовно з'єднаних конденсаторів дорівнює

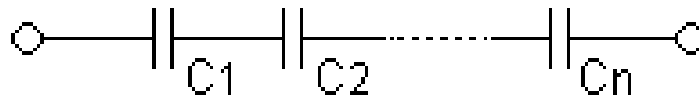


Рисунок 1.3 - Послідовне з'єднання конденсаторів.

$$C = \frac{1}{\sum_{i=1}^n 1/C_i} \quad (1.11)$$

Ця ємність завжди менша мінімальної ємності конденсатора, який входить в батарею. Але при послідовному з'єднанні зменшується загроза пробоем конденсаторів, оскільки на кожний конденсатор надходить лише частина різниці потенціалів джерела напруги.

Конденсатори також характеризуються питомою ємністю — відношення ємності до об'єму (або маси) конденсатора.

Ємність конденсатора можна виразити у Ампер·годинах виходячи з визначення Фаради:

$$\Phi = Кл/В = А \cdot с/В$$

приймавши $А \cdot год = 3600 А \cdot с$, отримуємо:

$$\Phi = 3600 \cdot А \cdot год/В$$

звідси, при напрузі в 1В і ємності конденсатора в 1Ф ємність в $А \cdot год$ буде:

$$А \cdot год = (1/3600) \cdot В \cdot \Phi$$

Іншою не менш важливою характеристикою конденсаторів є номінальна

напруга — значення електричної напруги, яке позначається на конденсаторі, при якому він може працювати у заданих умовах під час строку служби із зберіганням параметрів у допустимих межах.

Номінальна напруга залежить від конструкції конденсатора і властивостей застосованих матеріалів. При експлуатації напруга на конденсаторі не повинна перевищувати допустимої. Для більшості типів конденсаторів із збільшенням температури допустима напруга знижується.

Напругу, при якій впродовж 1-5 с виникає пробій, називають пробивною. Допустиму робочу напругу обирають у 3-10 разів меншою за пробивну.

Більшість конденсаторів із оксидним діелектриком (електролітичні) мають уніполярну провідність, внаслідок чого їх експлуатація можлива тільки при позитивному потенціалі аноду.

Електричний опір ізоляції — це опір конденсатора постійному струму, який визначається співвідношенням $R_{із} = U/I_{віт}$, де U — напруга, що спрямована на конденсатор, $I_{віт}$ — струм витоку.

ТКЄ — це параметр, який характеризує залежність ємності конденсатора від температури. Практично ТКЄ визначають як відношення зміни ємності конденсатора при зміні температури на 1°C . Але ТКЄ визначається не для всіх типів конденсаторів.

1.3 Класифікація конденсаторів

Основна класифікація конденсаторів проводиться за типом діелектрика в конденсаторі. Тип діелектрика визначає основні електричні параметри конденсаторів: опір ізоляції, стабільність ємності, величину втрат тощо.

- За видом діелектрика розрізняють:
- Вакуумні конденсатори (обкладки без діелектрика розташовані у вакуумі);
- Конденсатори з газоподібним діелектриком;
- Конденсатори з рідким діелектриком;
- Конденсатори з твердим неорганічним діелектриком: скляні, слюдяні,

керамічні, тонкошарові із неорганічних плівок (K10, K15, K26, K32,);

- Конденсатори з твердим органічним діелектриком: паперові, металопаперові, плівкові, комбіновані (K41, K42, K71, K72);

- Електролітичні та оксидо-напівпровідникові конденсатори. Як діелектрик використовується шар оксиду металу. Наприклад для конденсаторів оксидно-алюмінієвих (K50) це Al_2O_3 , а для оксидно-танталових (K51) — Ta_2O_5 . Однією обкладинкою слугує металева фольга (анод), а друга (катод) — це або електроліт (в електролітичних конденсаторах) або шар напівпровідника (в оксидно-напівпровідникових), нанесений безпосередньо на оксидний шар. Анод виготовляється, в залежності від типу конденсатора, з алюмінієвої, ніобієвої чи танталової фольги. Такі конденсатори відрізняються від інших типів перш за все своєю великою питомою ємністю, але здатні працювати при відносно низьких напругах і мають значні діелектричні втрати.

Крім того, конденсатори розрізняються за можливістю зміни своєї ємності:

- Постійні конденсатори — основний клас конденсаторів, який має сталу ємність (окрім як зменшення з часом використання);

- Змінні конденсатори — конденсатори, які дозволяють зміни ємності в процесі функціонування апаратури. Керування ємністю може відбуватися механічно, електричною напругою (варіконди) та температурою (термоконденсатори). Використовуються, наприклад, у радіоприймачах для налаштування частоти резонансного контуру.

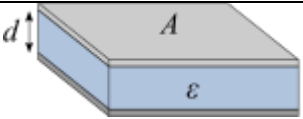
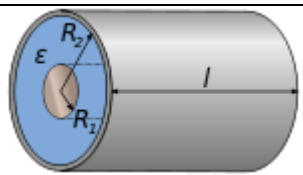
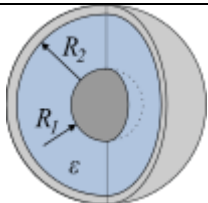
- Конденсатори підлаштування — конденсатори, ємність яких змінюється при разовому чи періодичному регулюванні і не змінюється в процесі функціонування апаратури. Їх використовують для підлаштування та вирівнювання початкових ємностей сполучених контурів, для періодичного підлаштування та регулювання ланцюгів схем, де потрібна незначна зміна ємності.

В залежності від призначення конденсатори можна умовно розділити на конденсатори загального та спеціального призначення. Конденсатори загального призначення використовуються практично у більшості видів і класів

апаратури. Традиційно до них відносять найбільш розповсюджені низьковольтні конденсатори, до яких не висуваються особливі вимоги. Решта конденсаторів є спеціальними. До них належать високовольтні, імпульсні, дозиметричні, пускові та інші конденсатори.

За формою обкладок конденсатори бувають: плоскі, циліндричні, сферичні, рулонні та інші.

Таблиця 1.1

Плоский конденсатор	$C = \varepsilon_0 \varepsilon_r \cdot \frac{A}{d}$	$E = \frac{Q}{\varepsilon_0 \varepsilon_r A}$	
Циліндричний конденсатор	$C = 2\pi\varepsilon_0\varepsilon_r \frac{l}{\ln(R_2/R_1)}$	$E(r) = \frac{Q}{2\pi r l \varepsilon_0 \varepsilon_r}$	
Сферичний конденсатор	$C = 4\pi\varepsilon_0\varepsilon_r \left(\frac{1}{R_1} - \frac{1}{R_2} \right)^{-1}$	$E(r) = \frac{Q}{4\pi r^2 \varepsilon_0 \varepsilon_r}$	
Сфера	$C = 4\pi\varepsilon_0 \varepsilon_r R_1$		

За способом монтажу конденсатори поділяються на елементи навісного монтажу і поверхневого (друкованого), а також для використання у складі мікросхем та мікромодулів. Виводи конденсаторів для навісного монтажу можуть бути жорсткими або м'якими, аксіальними або радіальними з дроту чи стрічки, у вигляді пелюсток, кабельного вводу, шпильок чи опорних гвинтів. У більшості конденсаторів одна з обкладок сполучається з корпусом, який служить другим виводом.

1.4 Мікроконтролерні системи

Мікроконтролер (MCU) - мікросхема, призначена для керування електронними пристроями. Типовий мікроконтролер поєднує в собі функції процесора і периферійних пристроїв, може містити ОЗУ і ПЗУ. По суті, це однокристальний комп'ютер, здатний виконувати прості завдання.

Використання однієї мікросхеми, замість цілого набору, як у випадку звичайних процесорів, що застосовуються в персональних комп'ютерах, значно знижує розміри, енергоспоживання і вартість пристроїв, побудованих на базі мікроконтролерів. Мікроконтролери є основою для побудови вбудованих систем, їх можна зустріти в багатьох сучасних приладах, таких, як телефони, пральні машини і т. п. Якщо представити всі типи сучасних мікроконтролерів (МК), то можна дивуватися величезною кількістю різноманітних приладів цього класу, доступних споживачеві. Проте всі ці пріори можна розділити на наступні основні типи:

- Вбудовувані (embedded) 8-розрядні МК;
- 16 і 32-розрядні МК;
- Цифрові сигнальні процесори.

Промисловістю випускаються дуже широка номенклатура вбудованих МК. У них всі необхідні ресурси (пам'ять, пристрої введення-виведення і т.д.) розташовуються на одному кристалі з процесорним ядром. Якщо подати харчування і тактові імпульси на відповідні входи МК, то можна сказати, що він як би «оживе» і з ним можна буде працювати. Зазвичай МК містять значну кількість допоміжних пристроїв, завдяки чому забезпечується їх включення в реальну систему з використанням мінімальної кількості додаткових компонентів. До складу цих МК входять:

- Схема початкового запуску процесора (Reset);
- Генератор тактових імпульсів;
- Центральний процесор;
- Пам'ять програм (E(E)PROM) і програмний інтерфейс;
- Засоби введення/виводу даних;
- Таймери, що фіксують кількість командних циклів.

Більш складні вбудовані МК можуть додатково реалізовувати наступні можливості:

- Вбудований монітор/налагодження програм;
- Внутрішні засоби програмування пам'яті програм (ROM);
- Обробка переривань від різних джерел;

- Аналоговий введення/виводу;
- Послідовний ввід/вивід (синхронний та асинхронний);
- Паралельний ввід/вивід (включаючи інтерфейс з комп'ютером);
- Підключення зовнішньої пам'яті (мікропроцесорний режим).

Всі ці можливості значно збільшують гнучкість застосування МК і роблять більш простим процес розробки систем на основі. Деякі МК (особливо 16 - і 32 біт) використовують тільки зовнішню пам'ять, яка включає в себе як пам'ять програм (ROM), так і деякий обсяг пам'яті даних (RAM), необхідний для даного застосування. Вони застосовуються в системах, де потрібен великий обсяг пам'яті і відносно не велика кількість пристроїв (портів введення/виводу. Типовим прикладом застосування такого МК з зовнішньої пам'яттю є контролер жорсткого диску (HDD) з буферної кеш-пам'яттю, який забезпечує проміжне зберігання і розподіл великих обсягів даних (порядку декількох мегабайт). Зовнішня пам'ять дає можливість такого мікроконтролеру працювати з більш високою швидкістю, ніж вбудований МК. Цифрові сигнальні процесори (DSP) - відносно нова категорія процесорів. Призначення DSP полягає в тому, щоб отримувати поточні дані від аналогової системи, обробляти дані і формувати відповідний відгук у реальному масштабі часу. Вони зазвичай входять до складу систем, використовуючись як пристроїв керування зовнішнім обладнанням, і не призначені для автономного використання.

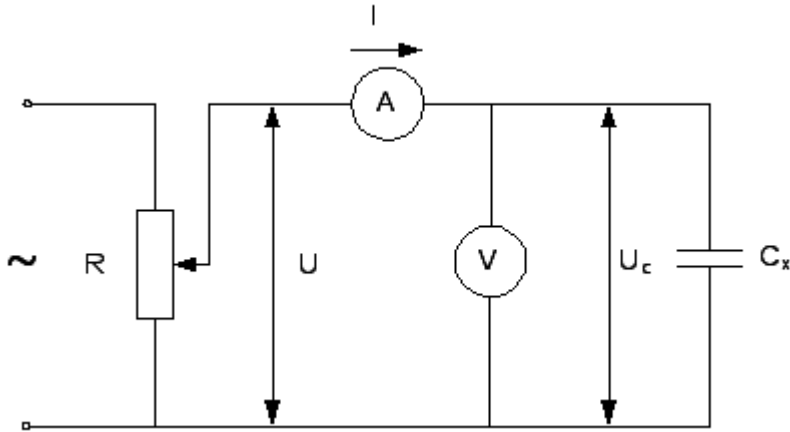
1.5 Методи вимірювання ємностей конденсаторів

Залежно від об'єму вимірювання, необхідної точності результату, діапазону робочих частот використовують різні методи для вимірювання ємності. Найчастіше при вимірюванні параметрів застосовуються методи:

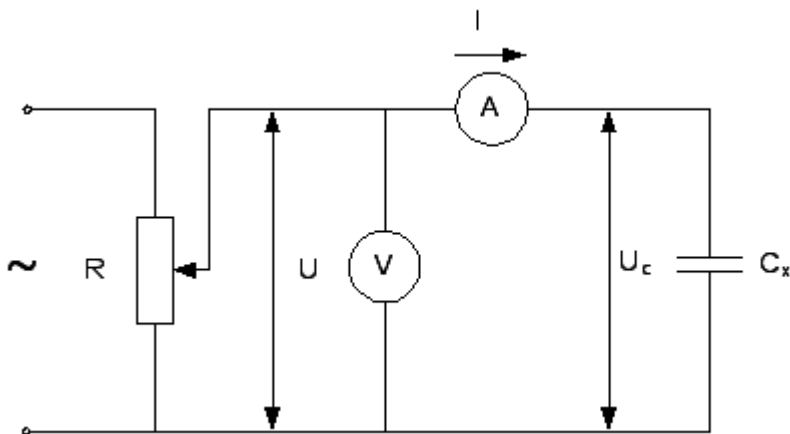
- вольтметра-амперметра;
- безпосередньої оцінки;
- мостовий;
- резонансний;

- дискретного відліку.

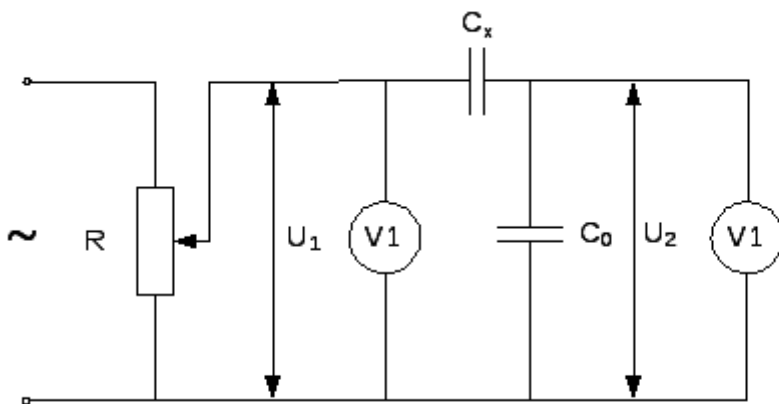
Розглянемо вимірювання ємності на основі електролітичних конденсаторів за методом вольтметра-амперметра, схеми якого приведені нижче:



а)



б)



в)

Рисунок 1.4 - Схеми вимірювання ємності конденсатора методом вольтметра-амперметра

Звідси випливає, що для вимірювання ємності цим методом необхідно знати частоту джерела живлення. При великих ємностях, тобто малому ємнісному опорі, рекомендується схема, зображена на рисунку 4 а), а при малих ємностях – рисунок 1.4 б).

Одним із різновидів методу вольтметра-амперметра є метод двох вольтметрів, що застосовується для вимірювання малих ємностей за схемою зображеною на рисунку 1.4 в).

Метод двох вольтметрів дає змогу вимірювати ємності від часток пікофарад. Частотний діапазон і точність вимірювання визначаються відповідними характеристиками вольтметрів. Похибка вимірювання вказаним методом становить 5 – 10 %.

Висновки

В даному розділі були наведені загальні відомості про конденсатор, загальні формули та методи вимірювання ємностей.

РОЗДІЛ 2

АЛГОРИТМ РОБОТИ ПОРТАТИВНОГО ВИМІРЮВАЧА ЄМНОСТІ

2.1 Блок схема алгоритму роботи вимірювача ємності

Блок-схема - представлення алгоритму розв'язування або аналізу задачі за допомогою геометричних елементів (блоків), які позначають операції, потік, дані тощо. Блок схема для нашого макету буде мати наступний вигляд:

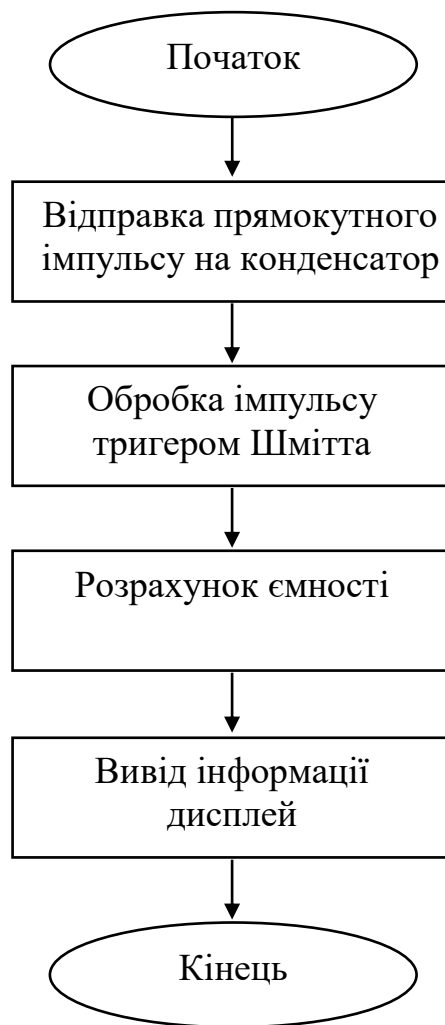


Рисунок 2.1 - Блок схема роботи програми для вимірювання ємності.

2.2 Структурна схема вимірювача ємності на arduino

Структурна схема — схема, яка визначає основні функціональні частини виробу, їх взаємозв'язки та призначення. Під функціональною частиною розуміють складову частину схеми: елемент, пристрій, функціональну групу, функціональну ланку.

Структурна схема призначена для відображення загальної структури пристрою, тобто його основних блоків, вузлів, частин та головних зв'язків між ними. Із структурної схеми повинно бути зрозуміло, навіщо потрібний даний пристрій і як він працює в основних режимах роботи, як взаємодіють його частини. Позначення елементів структурної схеми можуть обиратись довільно, хоча загальноприйнятих правил виконання схем слід дотримуватись.

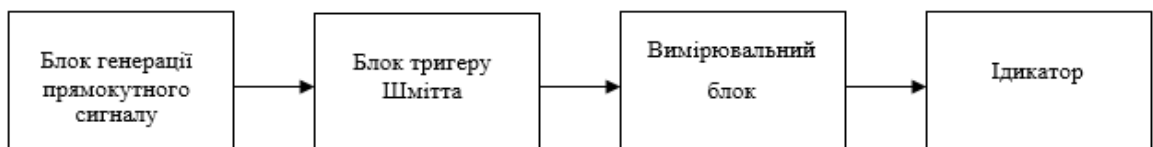


Рисунок 2.2 - Структурна схема портативного вимірювача ємності.

2.3 Функціональна схема вимірювача ємності на Arduino

Функціональна схема містить інформацію про способи реалізації пристроєм заданих функцій. За такою схемою можна визначити, як здійснюються перетворення і які для цього необхідні функціональні елементи. Функціональна схема— схема, що роз'яснює певні процеси, що відбуваються у певних функціональних ланцюгах виробу (устаткування) чи у виробі (устаткуванні) в цілому. На основі функціональної та структурної схем розробляється принципова схема. Такими схемами користуються для вивчення принципу роботи виробів(устаткуванні), а також при їх налагодженні контролі чи устаткуванні.

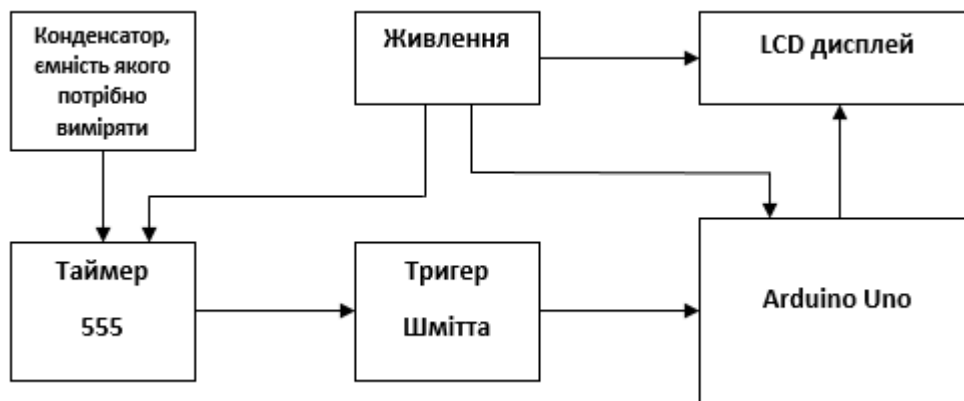


Рисунок 2.3 - Функціональна схема портативного вимірювача ємності.

2.4 Принципова схема вимірювача ємності на arduino

Електрична принципова схема (ГОСТ 2.702-75) є видом електричної схеми виробу, що дає найповніше уявлення про склад і принцип його роботи. При виконанні схем цифрової обчислювальної техніки керуються ГОСТ 2.708-81. Цей вид кресленика не враховує габаритних розмірів і реального розташування деталей об'єкта. За рівнем абстракції принципові електричні схеми займають середню позицію між функціональними і монтажними схемами.

Напринциповій електричній схемі (ГОСТ 2.709-89) зображуються всі складові частини виробу і зв'язки між ними, та елементи, якими закінчуються вхідні та вихідні ланки електричних кіл(роз'єми, затискачі тощо). Принципові електричні схеми призначенні для повного відображення взаємозв'язків пристроїв з урахуванням принципів їх дії і послідовності роботи. На принципових електричних схемах електричні елементи зображують за допомогою умовних позначень, а також вказують лінії зв'язків між ними, блоками та модулями. На схемі, також, розміщується наступна інформація: умовне зображення принципу роботи функціональних вузлів, пояснювальні написи, частини окремих елементів, діаграми переключення контактів, а також перелік використовуваних в даній схемі пристроїв.

Принципова схема вимірювача ємності на Arduino наведена Додатку А. Схема проста, для відображення виміряної ємності конденсатора до Arduino підключений LCD дисплей. До Arduino підключено схему генератора прямокутних імпульсів (555 в астабільному режимі), до якої ми повинні підключити конденсатор, ємність якого необхідно виміряти. Тригер Шмітта (мікросхема 74LS14) використовується для подачі на Arduino тільки прямокутного сигналу. Для фільтрації шуму ми додали кілька конденсаторів на лінію живлення.

Висновки

В даному розділі розглянуто алгоритм роботи портативного вимірювача ємності. Створено структурну, функціональну та принципову схеми.

РОЗДІЛ 3

РОЗРОБКА ПОРТАТИВНОГО ВИМІРЮВАЧА ЄМНОСТІ

3.1 Вибір платформи Arduino Uno R3

Arduino Uno - це пристрій на основі мікроконтролера ATmega328. У його склад входить все необхідне для зручної роботи з мікро контролером: 14 цифрових входів та виходів (з них 6 можуть використовуватися в якості ШІМ-виходів), 6 аналогових входів, кварцовий резонатор на 16 МГц, роз'єм USB, роз'єм живлення, роз'єм для внутрисхемного програмування (ICSP) і кнопка скидання. Для початку роботи з пристроєм досить просто подати живлення від AC / DC-адаптера або батарейки, або підключити його до комп'ютера за допомогою USB-кабелю.

На відміну від всіх попередніх плат Ардуіно, Uno в якості перетворювача інтерфейсів USB-UART використовує мікроконтролер ATmega16U2 (ATmega8U2 до версії R2) замість мікросхеми FTDI.

На платі Arduino Uno версії R2 для спрощення процесу оновлення прошивки доданий резистор, що підтягує до землі лінію HWB мікроконтролера 8U2.



Рисунок 3.1 - Arduino Uno R3 - вигляд спереду.

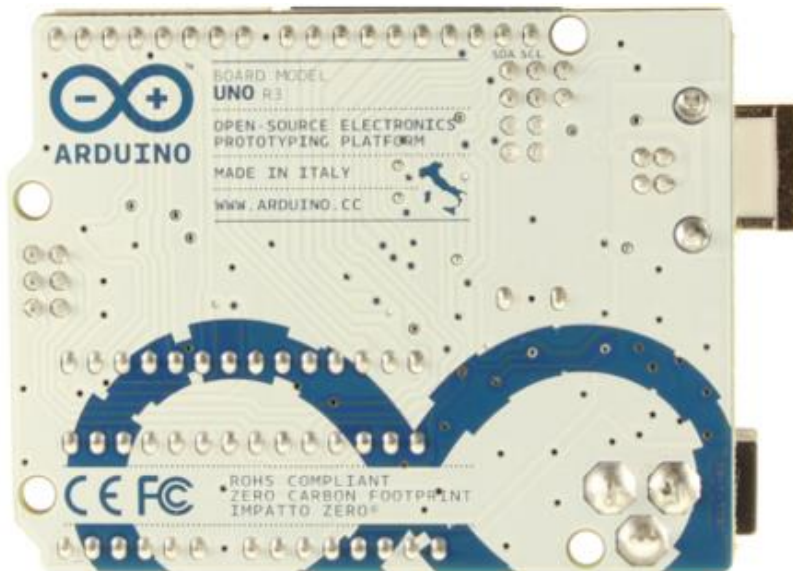


Рисунок 3.2 - Arduino Uno R3 - вигляд ззаду.

Зміни на платі версії R3 перераховані нижче:

- Терморегулятори 1.0: додані висновки SDA і SCL (біля виведення AREF), а також два нових виведення, розташованих біля виведення RESET.

Перший - IOREF - дозволяє платам розширення підлаштовуватися під робочу напругу Ардуіно. Даний вихід передбачений для сумісності плат розширення як з 5В-Ардуіно на базі мікроконтролерів AVR, так і з 3.3В-платами Arduino Due.

Другий вихід ні до чого не приєднаний і зарезервований для майбутніх цілей;

- Покращена стійкість ланцюга скидання;
- Мікроконтролер ATmega8U2 замінений на ATmega16U2.

"Uno" (в перекладі з італійської - "один") названий з нагоди майбутнього випуску Arduino 1.0. Спільно з Arduino 1.0 дані пристрої будуть базовими версіями Ардуіно. Uno - еталонна модель платформи Arduino і є останньою в серії USB-лат; для порівняння з попередніми версіями, см. список лат Arduino.

Arduino Uno може бути живиться від USB або від зовнішнього джерела живлення - тип джерела вибирається автоматично.

В якості зовнішнього джерела живлення (НЕ USB) може

поза незалежності від того, як живиться пристрій: від адаптера (7 - 12В), від USB (5В) або через вивід VIN (7 - 12В). Живити пристрій через висновки 5В або 3В3 не рекомендується, оскільки в цьому випадку не використовується стабілізатор напруги, що може привести до виходу плати з ладу.

- 3V3. 3.3В, що надходять від стабілізатора напруги на платі. Максимальний струм, споживаний від цього висновку, становить 50 мА.
- GND. Виходи землі.
- IOREF. Цей вихід надає платам розширення інформацію про робочій напрузі мікроконтролера Ардуіно. Залежно від напруги, ліченого з виведення IOREF, плата розширення може переключитися на відповідний джерело живлення або задіяти перетворювачі рівнів, що дозволить їй працювати як з 5В, так і з 3.3В-пристроями.

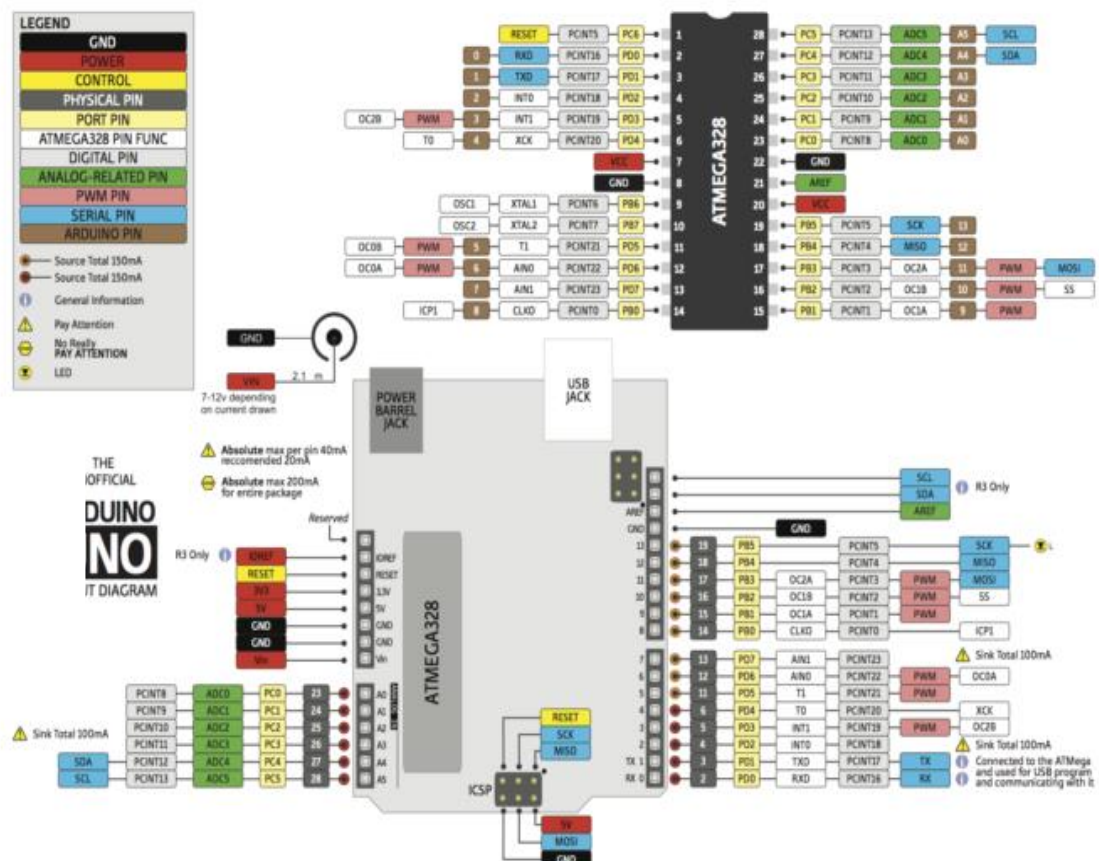


Рисунок 3.4 - Розпіновка плати Arduino Uno R3 на базі ATMEGA 328.

Обсяг флеш-пам'яті ATmega328 становить 32 КБ (з яких 0.5 КБ

використовуються загрузчиком). Мікроконтролер також має 2 КБ пам'яті SRAM і 1 КБ EEPROM (з якої можна зчитувати або записувати інформацію за допомогою бібліотеки EEPROM).

Піни плати, з використанням функцій `pinMode ()`, `digitalWrite ()` і `digitalRead ()` кожен з 14 цифрових висновків може працювати в якості входу або виходу. Рівень напруги на висновках обмежений 5В. Максимальний струм, який може віддавати або споживати один вихід, становить 40 мА. Всі висновки пов'язані з внутрішніми підтягуються резисторами (за замовчуванням відключеними) номіналом 20-50 кОм.

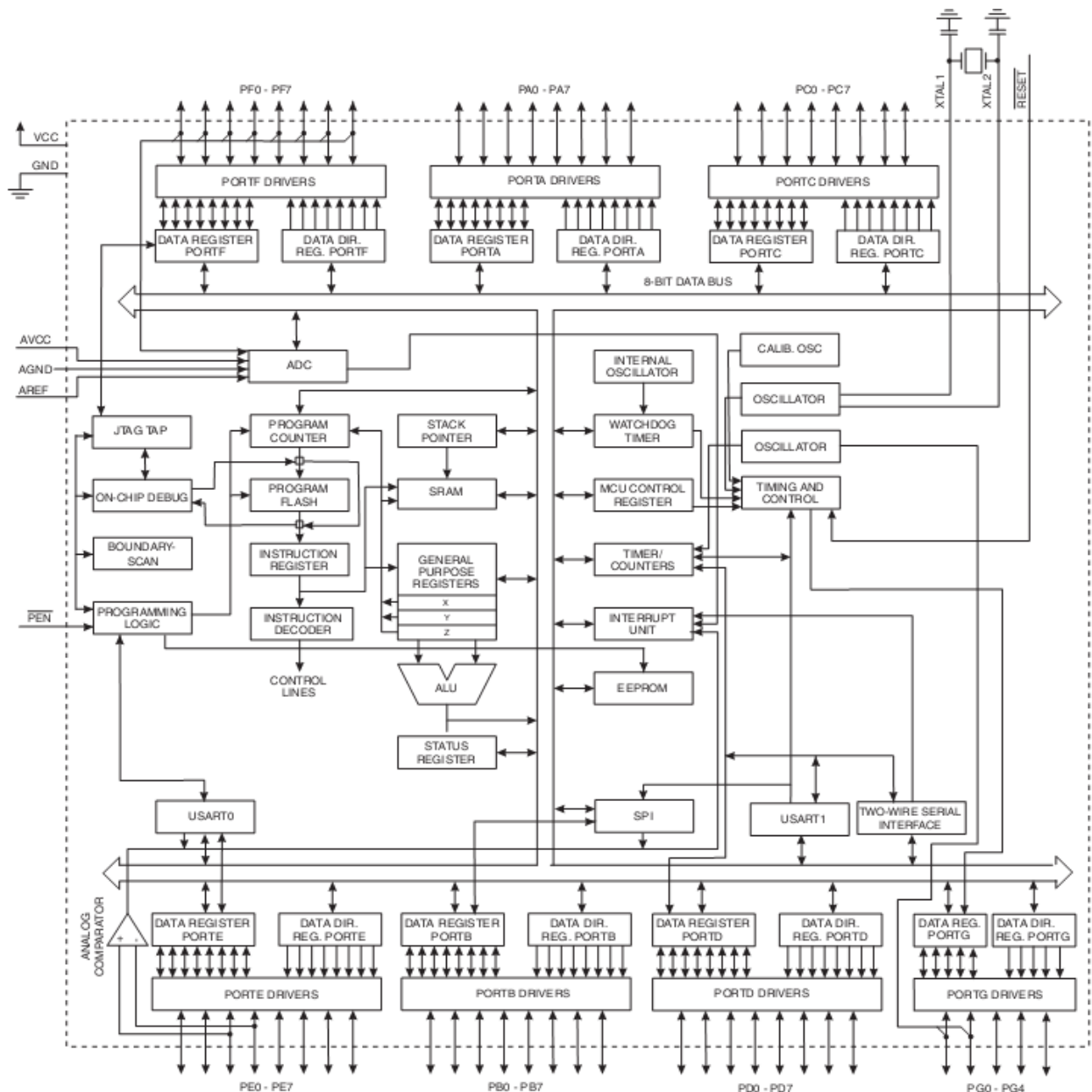


Рисунок 3.5 - Структурна схема Arduino Uno R3.

Крім цього, деякі висновки Ардуіно можуть виконувати додаткові функції:

- Послідовний інтерфейс: висновки 0 (RX) і 1 (TX). Використовуються для отримання (RX) і передачі (TX) даних по послідовному інтерфейсу. Ці висновки з'єднані з відповідними висновками мікросхеми ATmega8U2, яка виконує роль перетворювача USB-UART.

- Зовнішні переривання: висновки 2 і 3. Чи можуть служити джерелами переривань, що виникають при фронті, спаді або при низькому рівні сигналу на цих висновках. Для отримання додаткової інформації див. Функцію `attachInterrupt ()`.

- ШІМ: виходи 3, 5, 6, 9, 10 і 11. За допомогою функції `analogWrite ()` можуть виводити 8-бітові аналогові значення в вигляді ШІМ-сигналу.

- Інтерфейс SPI: висновки 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Із застосуванням бібліотеки SPI дані висновки можуть здійснювати зв'язок по інтерфейсу SPI.

- Світлодіод: 13. Вбудований світлодіод, приєднаний до висновку 13. При відправці значення HIGH світлодіод включається, при відправці LOW - вимикається.

В Arduino Uno є 6 аналогових входів (A0 - A5), кожен з яких може уявити аналогову напругу у вигляді 10-бітного числа (1024 різних значення). За замовчуванням, вимір напруги здійснюється щодо діапазону від 0 до 5 В. Проте, верхню межу цього діапазону можна змінити, використовуючи вихід AREF і функцію `analogReference ()`. Крім цього, деякі з аналогових входів мають додаткові функції:

- TWI: вихід A4 або SDA і вихід A5 або SCL. З використанням бібліотеки Wire дані висновки можуть здійснювати зв'язок по інтерфейсу TWI.

Крім перерахованих на платі існує ще кілька виходів:

- AREF. Опорна напруга для аналогових входів. Може бути задіяний функцією `analogReference ()`.

- Reset. Формування низького рівня (LOW) на цьому висновку призведе до перезавантаження мікроконтролера. Зазвичай цей вихід служить для

функціонування кнопки скидання на платах розширення.

Arduino Uno надає ряд можливостей для здійснення зв'язку з комп'ютером, ще одним Ардуіно або іншими мікроконтроллерами. У ATmega328 є приймач UART, що дозволяє здійснювати послідовну зв'язок за допомогою цифрових висновків 0 (RX) і 1 (TX). Мікроконтролер ATmega16U2 на платі забезпечує зв'язок цього приймача з USB-портом комп'ютера, і при підключенні до ПК дозволяє Ардуіно визначатися як віртуальний COM-порт. Прошивка мікросхеми 16U2 використовує стандартні драйвера USB-COM, тому установка зовнішніх драйверів не потрібно.

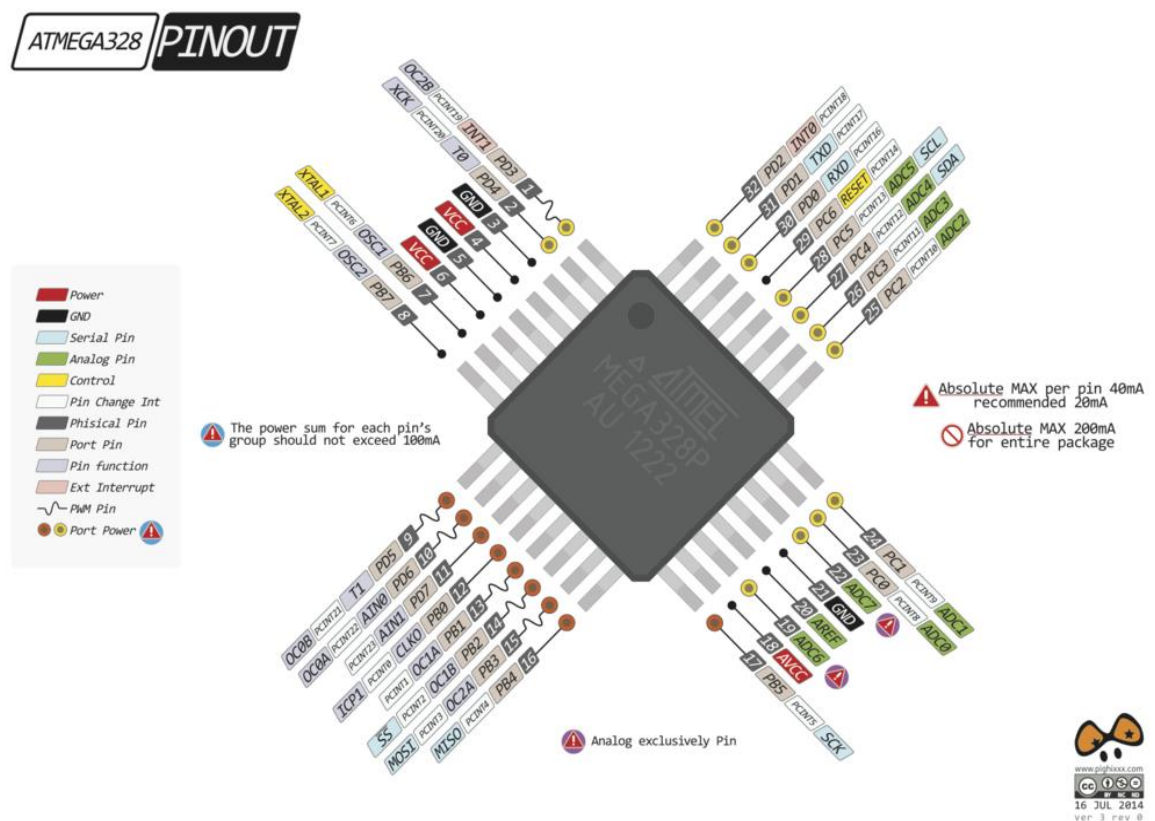


Рисунок 3.6 - Розпіновка мікропроцесора ATMEGA 328.

На платформі Windows необхідний тільки відповідний .inf-файл. У пакет програмного забезпечення Ардуіно входить спеціальна програма, що дозволяє зчитувати і відправляти на Ардуіно прості текстові дані. При передачі даних через мікросхему-перетворювач USB-UART під час USB-з'єднання з комп'ютером, на платі будуть мигати світлодіоди RX і TX. (При послідовній передачі даних за допомогою висновків 0 і 1, без використання USB-

перетворювача, дані світлодіоди задіюються).

Бібліотека SoftwareSerial дозволяє реалізувати послідовну зв'язок на будь-яких цифрових висновках Arduino Uno.

У мікроконтролері ATmega328 також реалізована підтримка послідовних інтерфейсів I2C (TWI) і SPI. У програмне забезпечення Ардуіно входить бібліотека Wire, що дозволяє спростити роботу з шиною I2C; для отримання більш докладної інформації див. документацію. Для роботи з інтерфейсом SPI використовуйте бібліотеку SPI.

3.2 Загальні відомості ЖК-контролер Hitachi HD44780

ЖК контролер Hitachi HD44780 є буквено - цифровий матричний рідкокристалічний дисплей (LCD) , контролер , розроблений Hitachi . Набір символів контролера включає символи ASCII , японські символи Kana та деякі символи у двох 28 символних рядках. Використовуючи драйвер розширення, пристрій може відображати до 80 символів.

HD44780 - це один з найпопулярніших ЖК-дисплеїв, який коли-небудь робився, з численними сторонніми дисплеями, використовуючи 16-контактний інтерфейс та набір інструкцій щодо сумісності.

ЖК-контролер Hitachi HD44780 обмежений монохромними текстовими дисплеями і часто використовується в копіювальних апаратах, факсах, лазерних принтерах, промисловому тестовому обладнанні, мережевому обладнанні, таких як маршрутизатори та пристрої зберігання даних.

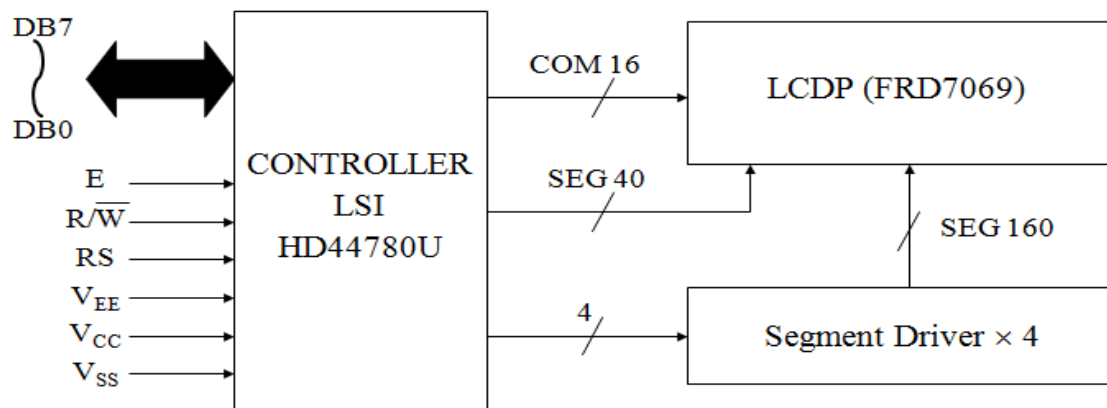


Рисунок 3.7 - Структурна схема ЖК-модуля на основі HD44780U.

Сумісні ЖК-екрани випускаються в декількох стандартних конфігураціях. Загальні розміри - це один ряд з восьми символів (8x1) та форматів 16×2 , 20×2 та 20×4 . Більші спеціальні розміри складаються з 32, 40 та 80 символів та з 1, 2, 4 або 8 рядками. Найчастіше виготовляється більша конфігурація - 40x4 символів, для чого потрібні два індивідуально адресовані контролери HD44780 з мікросхемами розширення, оскільки один чіп HD44780 може містити до 80 символів.

ЖК-дисплеї символів можуть мати підсвітку, яка може бути світлодіодною, флуоресцентною або електролюмінесцентною.

ЖК-дисплеї символів використовують 16 контактний інтерфейс, як правило, використовуючи штифти або крайові з'єднання картки на центрах розміром 0,1 дюйма (2,54 мм). Ті, хто не має підсвічування, можуть мати лише 14 контактів, опускаючи два штифти, що живлять світло.

Цей інтерфейс був розроблений для того, щоб легко підключитися до інтерфейсу XRAM Intel MCS-51, використовуючи лише два адресних штифти, що дозволяло відображати текст на РК за допомогою простих команд MOVX,

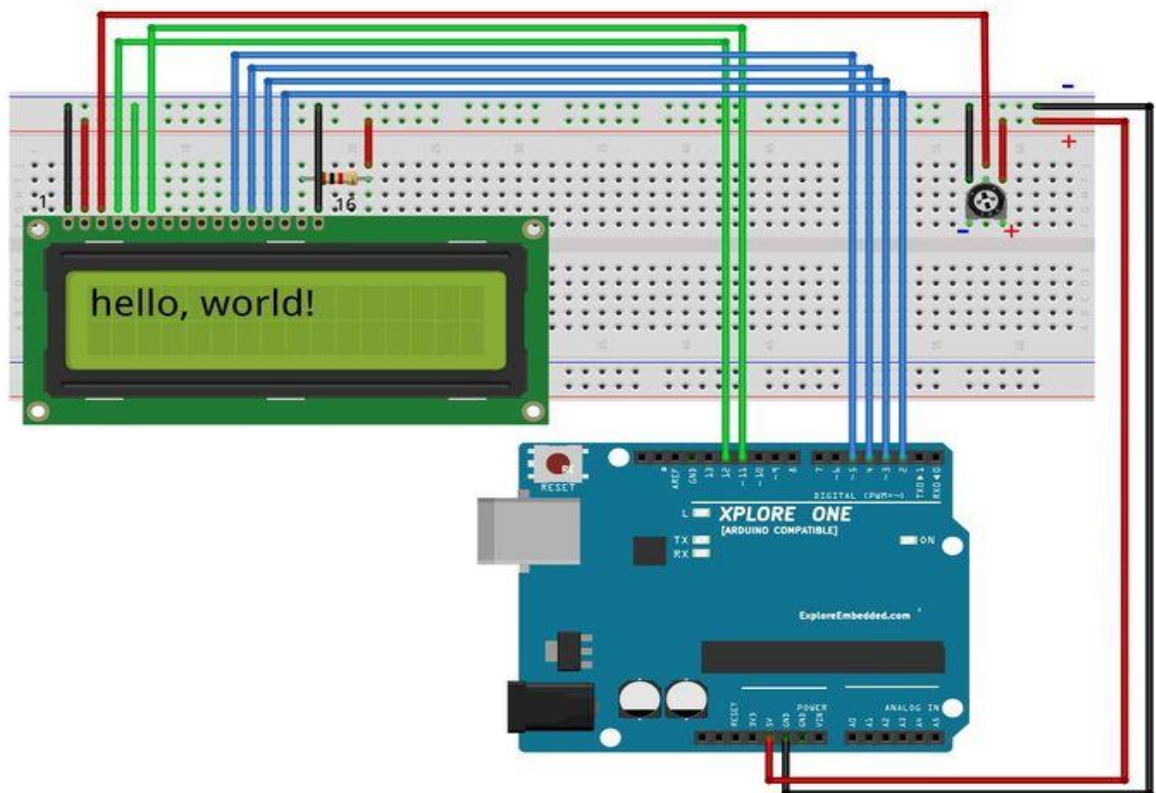


Рисунок 3.8 - Схема підключення ЖК-модуля на основі HD44780U до Arduino.

3.3 Генератор прямокутних імпульсів на мікросхемі таймера 555

Перш за все, ми поговоримо про генератор прямокутного сигналу на мікросхемі таймера 555, або астабільному мультивібраторі на 555. Ми знаємо, що ємність конденсатора не може бути виміряна цифровою схемою безпосередньо, іншими словами, Arduino Uno має справу з цифровими сигналами і не може вимірювати ємність безпосередньо. Тому для зв'язку конденсатора з «цифровим світом» використовуємо схему генератора прямокутного сигналу на 555.

Простіше кажучи, таймер забезпечує прямокутний вихідний сигнал, частота якого безпосередньо залежить від підключеного до нього ємності. Отже, спочатку ми отримуємо сигнал прямокутної форми, частота якого є показником ємності невідомого конденсатора, і передаємо цей сигнал на Arduino Uno для отримання відповідного значення.

Типова схема на таймері 555 в астабільному режимі показана нижче.

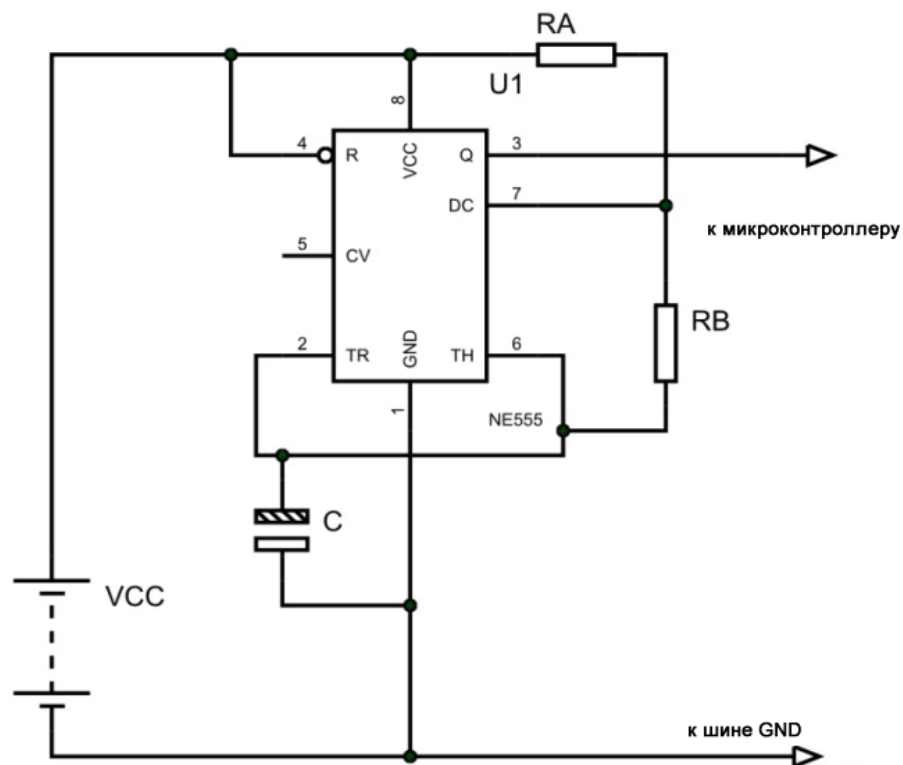


Рисунок 3.9 - Типова схема на таймері 555 в автоколебальному режимі.

Коли ланцюг під'єднано, як показано на рисунку (під'єднані контакти 2 і 6), він спрацьовує сам і вільно працює як мультивібратор. Зовнішній

конденсатор заряджається через R_A і R_B і розряджається тільки через R_B . Таким чином, робочий цикл можна точно встановити, регулюючи співвідношення цих двох резисторів.

У нестабільному режимі роботи C заряджається та розряджається між $1/3 V_{CC}$ і $2/3 V_{CC}$. Як і в режимі спрацьовування, час заряду та розряду, а отже, частота є незалежно від напруги живлення.

Час зарядки (вихід HIGH) визначається як:

$$t_1 = 0.693(R_A + R_B)C \quad (3.1)$$

і час розряду (вихід LOW) за:

$$t_2 = 0.693(R_B)C \quad (3.2)$$

Таким чином, загальний період T визначається як:

$$T = t_1 + t_2 = 0.693(R_A + 2R_B)C \quad (3.3)$$

Тоді частота коливань буде:

$$F = \frac{1}{T} = \frac{1.44}{(R_A + 2R_B)C} \quad (3.4)$$

Робочий цикл визначається так:

$$\frac{t_1}{(t_1 + t_2)} = \frac{(R_A + R_B)}{(R_A + 2R_B)} = 1 - \left[\frac{R_B}{(R_A + R_B)} \right] \quad (3.5)$$

Виведем ємність:

$$C = \frac{1.44}{(R_A + 2R_B)F} \quad (3.6)$$

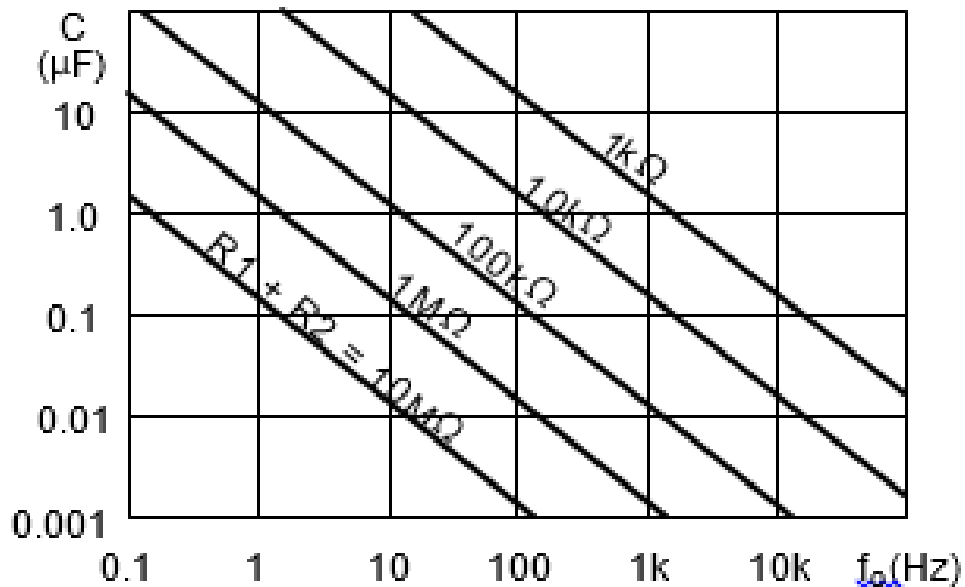


Рисунок 3.10 – Частота вільної роботи в порівнянні з R_A , R_B , C .

3.4 Тригер Шмітта

Сигнали, що генеруються схемою на таймері, не зовсім безпечні для безпосередньої передачі на Arduino Uno. З огляду на чутливість Arduino Uno, ми використовуємо елемент тригера Шмітта. Тригер Шмітта є цифровий логічний елемент, призначений для арифметичних і логічних операцій.

Цей елемент забезпечує вихідний сигнал (OUTPUT) на основі рівня напруги вхідного сигналу (INPUT). Тригер Шмітта має пороговий рівень напруги (THRESHOLD): коли рівень вхідного сигналу вище порогового рівня елемента, рівень сигналу на виході буде дорівнює високому логічному рівню. Якщо рівень вхідного сигналу нижче порога, на виході буде низький логічний рівень. Зазвичай у нас немає окремого тригера Шмітта, за ним завжди слід елемент НЕ.

Використовувати мікросхему 74LS14, яка містить 6 тригерів Шмітта. Ці шість елементів всередині підключені, як показано на рисунку нижче.

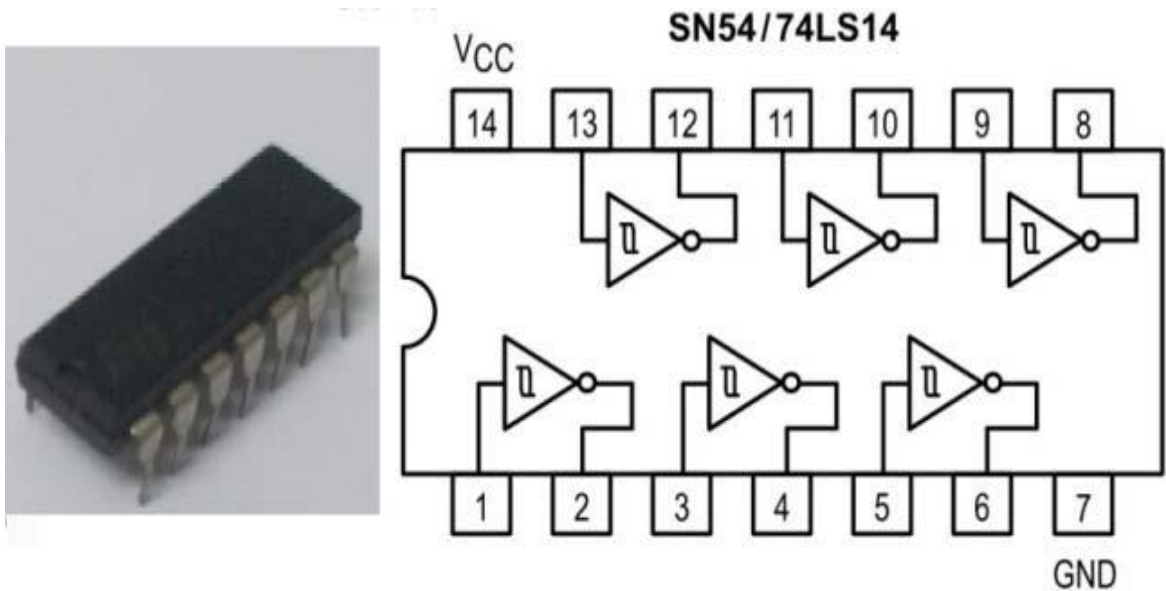


Рисунок 3.11 - Мікросхема 74LS14, що містить шість тригерів Шмітта.
Терморегулятори.

Таблиця істинності інвертованого тригера Шмітта показана нижче, відповідно до неї ми повинні запрограмувати Arduino Uno для інвертування позитивних і негативних періодів часу на її висновках.

Таблиця 3.1

Вхід	Вихід
A	Y
L	H
H	L

- H - високий логічний рівень;
- L - низький логічний рівень.

Тепер, коли ми подамо сигнал будь-якого типу на елемент тригера Шмітта, у нас на виході буде прямокутний сигнал з інвертованими тимчасовими періодами, і цей сигнал ми подамо на Arduino Uno.

Висновки

В даному розділі була розглянута мікроконтролерна частина випускної кваліфікаційної роботи та модулі які будуть використані для проектування.

РОЗДІЛ 4

ПРОГРАМНІ ЗАСОБИ ДЛЯ НАПИСАННЯ КОДУ ВИМІРЮВАЧА ЄМНОСТІ НА БАЗІ ПЛАТФОРМИ ARDUINO

4.1 Arduino IDE

Arduino IDE це середовище розробки з відкритим вихідним кодом, написане спеціально для створення скетчів під Arduino платформу. Воно дозволяє легко написати код і завантажити його на плату. Arduino IDE працює на Windows, Mac OS X, Linux і. Середовище написане в Java на основі Processing та іншого програмного забезпечення з відкритим вихідним кодом.

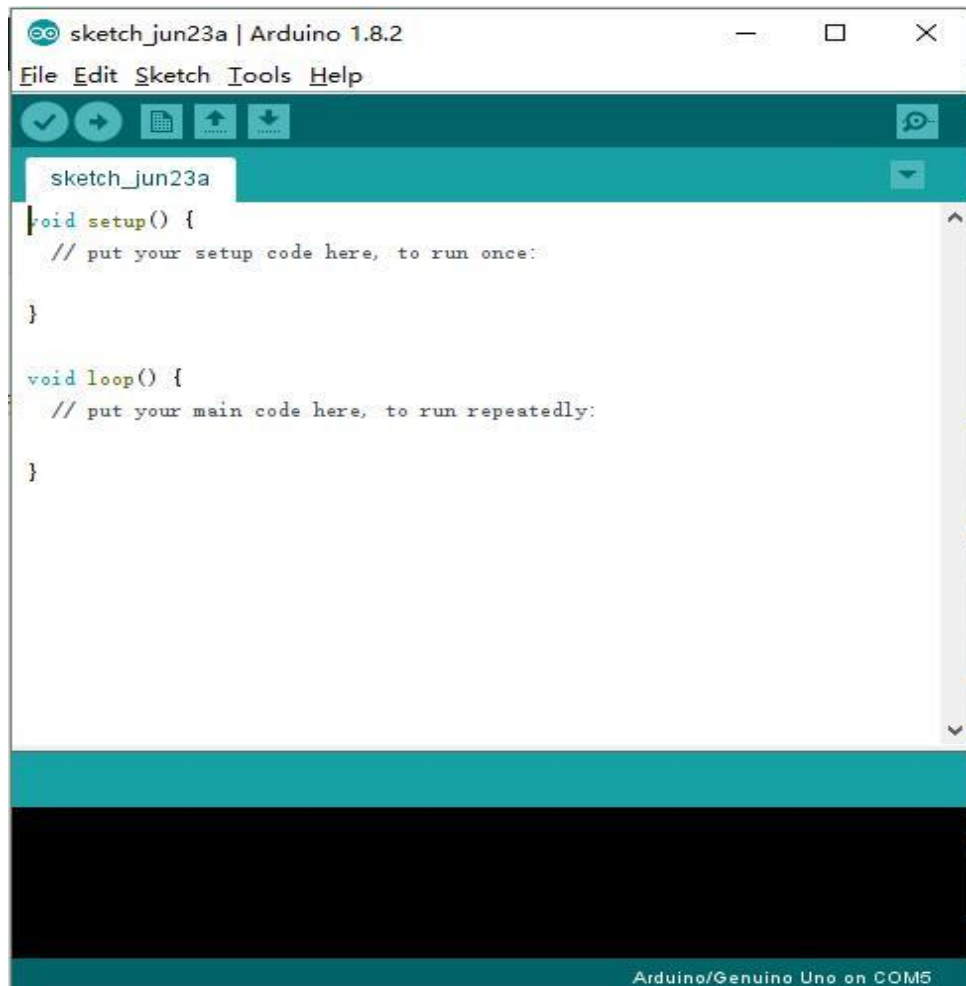


Рисунок 4.1 - Вікно програми.

Середовище розробки Arduino складається з вбудованого текстового

редактору програмного коду, області повідомлень, вікна виведення тексту (консолі), панелі інструментів з кнопками часто використовуваних команд і декількох меню. Для завантаження програм і зв'язку середовище розробки підключається до апаратної частини Arduino [8]. Програма, написана в середовищі Arduino, називається скетч. Скетч пишеться в текстовому редакторі, що має інструменти вирізки / вставки, пошуку / заміни тексту. Під час збереження та експорту проекту в області повідомлень з'являються пояснення, також можуть відображатися виникли помилки. Вікно виводу тексту (консоль) показує повідомлення Arduino, що включають повні звіти про помилки та іншу інформацію. Кнопки панелі інструментів дозволяють перевірити і записати програму, створити, відкрити і зберегти скетч, відкрити моніторинг послідовної шини:

- «Verify» Перевірка програмного коду на помилки, компіляція;
- «Load» Перевірка програмного коду на помилки, компіляція та загрузка коду на платформу;
- «New» Створення нового скетчу;
- «Open» Відкриття меню доступу до всіх скетчам в блокноті. Відкривається натисканням в поточному вікні.
- «Save» Збереження скетчу;
- «Serial monitor» Відкриття моніторингу послідовної шини (Serial monitor).

Додаткові команди згруповані в п'ять меню: File, Edit, Sketch, Tools, Help. Доступність меню визначається роботою, виконуваною в даний момент.

В розділі проводиться вибір та аналіз апаратних, апаратно-програмних, програмних засобів, які забезпечують реалізацію алгоритму розв'язання поставленої задачі, розробка схеми електричної функціональної з висвітленням особливостей функціонування базових компонентів.

Edit:

- Copy for Discourse - Копіює в буфер обміну відповідний для розміщення на форумі код скетчу з виділенням синтаксису.
- Copy as HTML - Копіює код скетчу в буфер обміну як HTML код, для

розміщення на веб-сторінках.

Sketch:

- Verify - Перевірка скетчу на помилки.
- Import Library - Додає бібліотеку в поточний скетч, вставляючи директиву `#include` в код скетчу. Детальна інформація в описі бібліотек нижче (Libraries).
- Show Sketch Folder - Відкриває папку, яка містить файл скетчу, на робочому столі.
- Add File... - Додає файл в скетч (файл буде скопійований з поточного місця розташування). Новий файл з'являється в новій закладці у вікні скетчу.

Файл може бути видалений з скетчу за допомогою меню закладок

Tools:

- Auto Format - Дана опція оптимізує код, наприклад, вибудовує в одну лінію по вертикалі відкриває і закриває дужки і поміщає між ними твердження.
- Board - Вибір використовуваної платформи. Список з описом платформ наводиться нижче.
- Serial Port - Меню містить список послідовних пристроїв передачі даних (реальних і віртуальних) на комп'ютері. Список оновлюється автоматично кожного разу при відкритті меню Tools.
- Burn Bootloader - Пункти даного меню дозволяють записати Завантажувач (Bootloader) в мікроконтролер на платформі Arduino. Дана дія не потрібно в поточній роботі з Arduino, але стане в нагоді, якщо є новий ATmega (без завантажувача). Перед записом рекомендується перевірити правильність вибору платформи з меню. При використанні AVR ISP необхідно вибрати відповідний програматору порт з меню Serial Port [9].

Блокнот (Sketchbook) - середовищем Arduino використовується принцип блокнота: стандартне місце для зберігання програм (скетчів). Скетчі з блокнота відкриваються через меню `File > Sketchbook` або кнопкою `Open` на панелі інструментів. При першому запуску програми Arduino автоматично створюється директорія для блокнота. Розташування блокнота змінюється через діалогове вікно `Preferences`.

Закладки, Файли та Компіляція дозволяють працювати з декількома файлами скетчів (кожен відкривається в окремій закладці). Файли коду можуть бути стандартними Arduino (без розширення), файлами C (розширення * .c), файлами C ++ (* .cpp) або головними файлами (.h).

Перед завантаженням скетчу потрібно задати необхідні параметри в меню Tools> Board і Tools> Serial Port. Платформи описуються далі за текстом. В ОС Mac послідовний порт може позначатися як dev / tty.usbserial-1B1 (для плати

USB) або /dev/tty.USA19QW1b1P1.1 (для плати послідовної шини, підключеної через адаптер Keyspan USB-to-Serial). В ОС Windows порти можуть позначатися як COM1 або COM2 (для плати послідовної шини) або COM4, COM5, COM7 і вище (для плати USB). Визначення порту USB проводиться в поле Послідовною шини USB Диспетчера пристроїв Windows. В ОС Linux порти можуть позначатися як / dev / ttyUSB0, / dev / ttyUSB1.

Після вибору порту і платформи необхідно натиснути кнопку завантаження на панелі інструментів або вибрати пункт меню File> Upload to I / O Board. Сучасні платформи Arduino перезавантажуються автоматично перед завантаженням. На старих платформах необхідно натиснути кнопку перезавантаження. На більшості плат під час процесу будуть мигати світлодіоди RX і TX. Середовище розробки Arduino виведе повідомлення про закінчення завантаження або про помилки.

При завантаженні скетчу використовується Завантажувач (Bootloader) Arduino, невелика програма, що завантажується в мікроконтролер на платі. Вона дозволяє завантажувати програмний код без використання додаткових апаратних засобів. Завантажувач (Bootloader) активний в перебігу декількох секунд при перезавантаженні платформи і при завантаженні будь-якого з скетчів в мікроконтролер. Робота завантажувач (Bootloader) розпізнається за миготінню світлодіода (13 пін) (наприклад при перезавантаженні плати).

Моніторинг послідовної шини (Serial Monitor), відображає дані посилаються в платформу Arduino (плата USB або плата послідовної шини).

Для відправки даних необхідно ввести текст і натиснути кнопку Send або Enter.

Потім вибирається швидкість передачі зі списку, відповідна значенню `Serial.begin` в скетчі. На ОС Mac або Linux платформа Arduino буде перезавантажена (скетч почнеться спочатку) при підключенні моніторингу послідовної шини.

4.2 Структура програми в Arduino IDE

Структура програми в Arduino IDE може змінюватися програмістом в залежності від завдань і конкретних рішень, але основний її незмінний мінімальний скелет як на цьому прикладі обов'язково зберігається з двох функцій `setup` і `loop`. Без них програма не компілюватися.

```
void setup() {  
  }  
void loop() {  
  }
```

Функція - це фрагмент програмного коду, який має унікальну назву і обмежений фігурними дужками, до якого можна звернутися з іншого місця програми.

`Setup` означає налаштування. Пряме призначення цієї функції полягає в тому, щоб програмісти в ній задавали початкові параметри функціоналу контролера і програмних об'єктів, що використовуються в інших частинах програми.

Слово `loop` означає повторюється цикл - зациклення виконання набору виразів, які перебувають в цій функції. Її призначення в тому, щоб зберігати вираження основного циклу програми контролера.

Після старту контролер приймається обробляти вираження, що знаходяться між фігурними дужками секції `setup`.

Після закінчення виконання всіх завдань першої функції, контролер береться за нескінченне послідовне виконання програмного коду секції `loop`, який за задумом повинен виконувати всю корисну роботу, закладену нами, в

динамічному режимі.

Тепер розглянемо простий приклад програми з подібною структурою, із закладеною можливістю моніторингу її виконання в моніторі порту, а так само по індикації за допомогою світлодіода на платі контролера. На платі Arduino Nano і UNO є світлодіод, підключений до 13-му піну і ним ця програма може блимати.

```
int led_pin = 13;
void setup() {
  Serial.begin(9600);
  pinMode(led_pin, OUTPUT);
  Serial.println("setup");
}
void loop() {
  digitalWrite(led_pin, HIGH);
  delay(1000);
  digitalWrite(led_pin, LOW);
  delay(1000);
  Serial.println("loop");
}
```

Тут до розглянутої нами структури додався рядок коду, який знаходиться попереду і поза фігурних дужок будь-якої функції. У цьому місці прийнято розміщувати рядки підключення бібліотек, оголошення глобальних змінних, констант і примірників класів.

Отже це були уявлення про найпростіші складові проекту програми-скетчів для контролерів Arduino. Охарактеризували що вміст функції setup обробляється контролером один раз після запуску контролера, а вміст функції loop обробляється циклічно без зупинки. Так само, що скетч не компілюватиметься при відсутності цих функцій.

4.3 Як Arduino вимірює ємність, функція pulseIn()

У Arduino є спеціальна функція pulseIn(), яка дозволяє нам визначати тривалість позитивного або негативного стану конкретного прямокутного сигналу:

```
Htime=pulseIn(8,HIGH);
```

```
Ltime = pulseIn(8, LOW);
```

Функція pulseIn() вимірює час, протягом якого високий або низький логічний рівень присутній на виводі 8 Arduino. Функція pulseIn() вимірює час високого логічного рівня (Htime) і час низького логічного рівня (Ltime) в мікросекундах. Коли ми складемо Htime і Ltime разом, то отримаємо тривалість періоду, а інвертовану її, ми отримаємо частоту.

Можливі деякі похибки в вимірювання довгих сигналів. Функція може вимірювати сигнали довжиною від 10 мікросекунд до 3 хвилин.

Загальний синтаксис функції має наступний вигляд

```
pulseIn(pin, value)
```

```
pulseIn(pin, value, timeout)
```

Параметри:

- pin: номер порту вхід / виходу, на якому буде очікуватися сигнал. (Int)
- value: тип очікуваного сигналу - HIGH або LOW .
- timeout (опціонально): час очікування сигналу (таймаут) в мікросекундах; за замовчуванням - одна секунда.

Значення, що повертається це довжина сигналу в мікросекундах або 0, якщо сигнал не отримана до закінчення часу очікування.

Приклад застосування функції:

```
int pin = 7;
```

```
unsigned long duration;
```

```
void setup()
```

```
{
```

```
pinMode(pin, INPUT);  
}  
void loop()  
{  
duration = pulseIn(pin, HIGH);  
}
```

Висновки

У даному розділі розглянуті програмні засоби для створення системи на основі Arduino, а саме програмне середовище та функцію `pulseIn()`.

РОЗДІЛ 5

ПРОГРАМУВАННЯ ARDUINO UNO R3 ТА ПЕРЕГЛЯД РЕЗУЛЬТАТІВ

5.1 Написання коду скетчу для вимірювача ємності

Оголошення змінних в які будемо записувати вихідні та розраховані значення під час вимірювання ємності:

```
int32_t Htime;  
int32_t Ltime;  
float Ttime;  
float frequency;  
float capacitance;
```

Встановимо в функції void setup() сигнальний вхід на 8 пін:

```
void setup()  
{  
    pinMode(8,INPUT);  
}
```

До функції void loop() додамо код для вимірювання ємності, для цього використаємо цикл для усереднення значення, вимірювання проводимо 5 разів та знаходимо середнє значення:

```
void loop()  
{  
    for (int i=0;i<5;i++)    // измеряем время пять раз  
    {  
        Ltime=(pulseIn(8,HIGH)+Ltime)/2;  
        Htime=(pulseIn(8,LOW)+Htime)/2;  
    }  
}
```

Після вимірювання потрібно додати Htime і Ltime для отримання довжини періоду, а інвертувавши її отримаємо частоту:

$$Ttime = Htime + Ltime;$$
$$frequency = 1000000 / Ttime;$$

Для точного отримання значення ємності ми розрахували результат в нФ, помноживши отримані результати (у Фарада) на "1000000000". Отже, якщо ми знаємо частоту прямокутного сигналу, то можемо отримати значення ємності.

$$capacitance = (1.44 * 1000000000) / (20800 * frequency);$$

5.2 Написання коду скетчу для відображення результату на ЖК-контролер Hitachi HD44780

Підключення бібліотеки LiquidCrystal.h яка необхідна для підключення та роботи із дисплеєм із присвоєнням пінів:

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 4, 5, 6, 7);
```

В кінець функції void setup() добавимо наступний код, він встановлює розмір екрану, а саме кількість стовбців та стрічок, після чого встановимо курсор на першу стрічку і першу колонку та виведемо у першу стрічку дисплею напис «capacitance =»:

```
void setup()
{
    lcd.begin(16, 2);
    lcd.setCursor(0,0);
    lcd.print("capacitance =");
}
```

Після чого до функції void loop(), після отримання середнього значення для кожного періоду та розрахунку ємності в нФ, додамо наступний код який

встановлює курсор на другу стрічку і першу колонку куди виведемо результати розрахунку із одиницею вимірювання «nF», після чого призупинимо виконання програми на пів секунди:

```
void loop() {  
    lcd.setCursor(0,1);  
    lcd.print(capacitance);  
    lcd.print(" nF ");  
    delay(500);  
}
```

5.3 Тестування та результати роботи портативного вимірювача ємності

Отже, в підсумку ми підключаємо невідомий конденсатор до схеми на таймері 555, яка генерує прямокутний вихідний сигнал, частота якого безпосередньо пов'язана з ємністю конденсатора. Цей сигнал подається на Arduino Uno через тригер Шмітта. Arduino Uno вимірює частоту. Ми програмуємо Arduino Uno для розрахунку ємності по відомій частоті, використовуючи формулу, яку обговорювали раніше.

Результати, які було отримано під час тестування:

- підключення електролітичного конденсатору на 1 мкФ, результат склав 1091,84 нФ (відносна похибка 9 %), для поліестерового конденсатора 0,1 мкФ результат склав 107,70 нФ (відносна похибка 7 %);



Рисунок 5.1 - Тестування вимірювача ємності на Arduino - електролітичний конденсатор 1 мкФ



Рисунок 5.2 - Тестування вимірювача ємності на Arduino - поліестеровий конденсатор 0,1 мкФ

• керамічний конденсатор 0,1 мкФ, і результат склав 100,25 нФ (відносна похибка 0,25 %), для електролітичного конденсатора 4,7 мкФ результат вимірювань склав 4842,83 нФ (відносна похибка 3 %).



Рисунок 5.3 - Тестування вимірювача ємності на Arduino - керамічний конденсатор 0,1 мкФ



Рисунок 5.4 - Тестування вимірювача ємності на Arduino-електролітичний конденсатор 4,7мкФ

Висновки

В розділі написано скетч який вимірює ємність конденсатора та виводить результат на дисплей. Також проведено тестування приладу.

ВИСНОВОК

У роботі досліджено можливість побудови портативного вимірювача ємності на базі платформи Arduino. Проаналізовано основні методи вимірювання ємності та запропоновано метод і структуру портативного вимірювача ємності на базі платформи Arduino.

Програмну частину реалізовано у кодї для прошивки мікроконтролера (Arduino скетч). Для програмування мікроконтролера використовується мова C. Дана програма визначає ємність конденсатора, вимірюючи частоту коливань на виході генератора, де частотозадаючим елементом і виступає конденсатор, ємність якого вимірюється.

Для візуального відображення вимірюваної відстані використано модуль дисплею, а саме ЖК-контролер Hitachi HD44780

Основним матеріалом для дослідження було взято Arduino UNO R3 на основі якого і було побудовано макет приладу. Результати вимірювань підтвердили працездатність запропонованого вимірювача, а відносна похибка вимірювань для конденсаторів різних типів перебувала у межах 0,25 %... 9 %, що є цілком прийнятно для приладів такого класу.

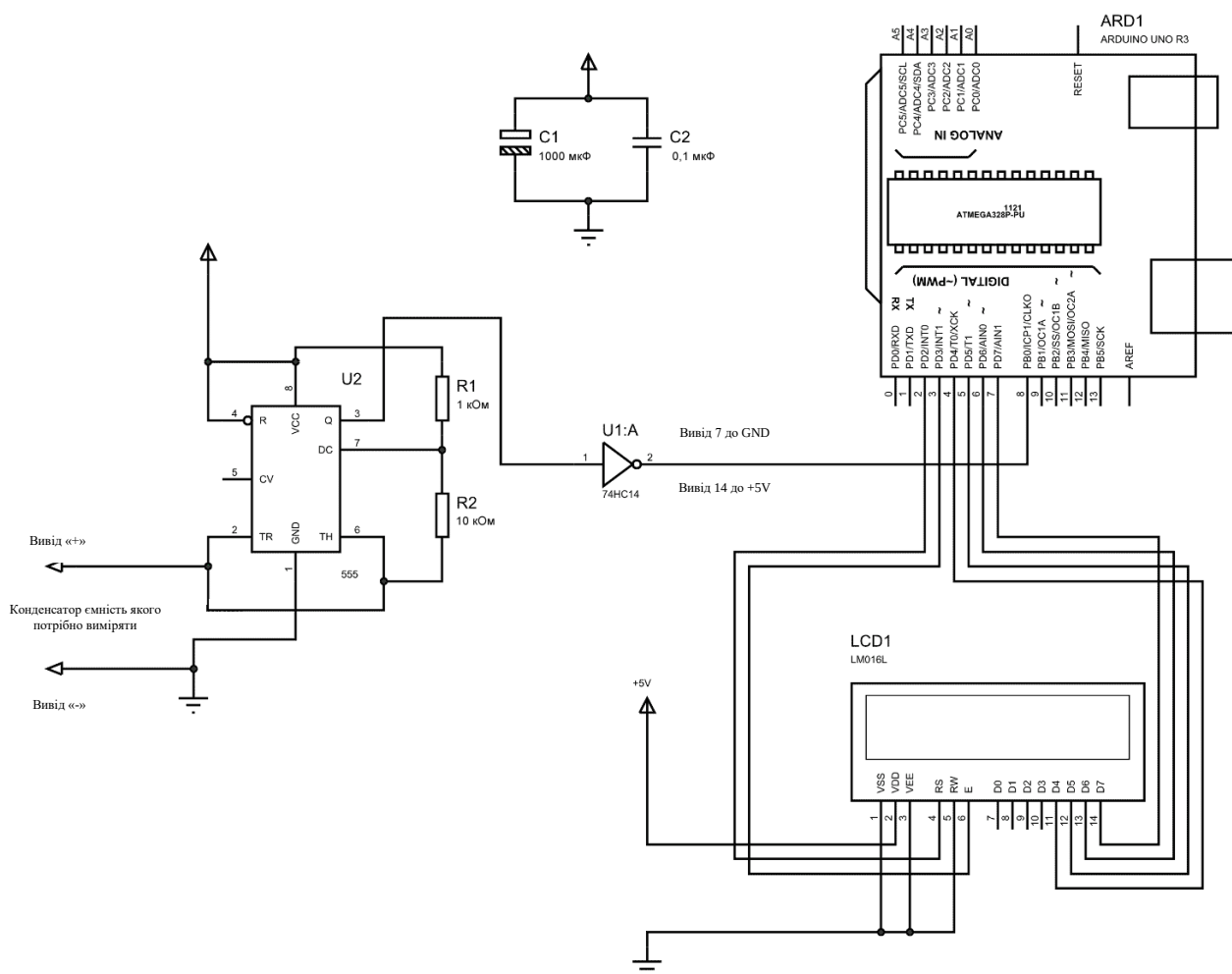
На основі такої вимірювача можуть бути спроектовані та розроблені десятки пристроїв для полегшення роботи інженерам, схемотехнікам, науковцям та ісім, хто займається створенням інноваційних пристроїв.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ

1. Kleeberger K., Bormann R., Kraus W., Huber M. F. "A Survey on Learning-Based Robotic Grasping" // Current Robotics Reports. — 2020. — Vol. 1. — pp. 239-249.
2. Siciliano, B. (Editor) "Springer Handbook of Robotics" — Springer International Publishing, 2016. — 2228 с.
3. Будіщев М. С. Електротехніка, електроніка та мікропроцесорна техніка. Підручник. — Львів : Афіша, 2001. — 424 с.
4. ДСТУ 2843-94. Електротехніка. Основні поняття. Терміни та визначення. Чинний від 1995-01-01. — Київ : Держспоживстандарт України, 1995. — 65 с.
5. Сайт по програмуванню [Електронний ресурс] // Режим доступу: metanit.com
6. Санітарні норми виробничого шуму, ультразвуку та інфразвуку. ДСН 3.3.6.037.99 (затверджено Постановою Головного Державного санітарного лікаря України від 1.12.1999 р. № 37).
7. Corke, P. "Robotics, Vision, and Control: Fundamental Algorithms in MATLAB" — Springer, 2023. — 819 с.
8. Thrun, S., Burgard, W., Fox, D. "Probabilistic Robotics" — MIT Press, 2005. — 667 с.
9. Craig, J. J. "Introduction to Robotics: Mechanics and Control" — Pearson, 2022. — 400 с.
10. Lynch, K. M., Park, F. C. "Modern Robotics: Mechanics, Planning, and Control" — Cambridge University Press, 2017. — 576 с.

ДОДАТОК А

ПРИНЦИПОВА СХЕМА ПОРТАТИВНОГО ВИМІРЮВАЧА ЄМНОСТІ



ДОДАТОК В

ЛІСТИНГ СКЕТЧУ ВИМІРЮВАННЯ

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

int32_t Htime;
int32_t Ltime;
float Ttime;
float frequency;
float capacitance;

void setup()
{
    pinMode(8,INPUT);
    lcd.begin(16, 2);
    lcd.setCursor(0,0);
    lcd.print("capacitance =");
}

void loop()
{
    for (int i=0;i<5;i++)
    {
        Ltime=(pulseIn(8,HIGH)+Ltime)/2;
        Htime=(pulseIn(8,LOW)+Htime)/2;
    }

    Ttime = Htime+Ltime;
    frequency=1000000/Ttime;

    capacitance = (1.44*1000000000)/(21000*frequency);
    lcd.setCursor(0,1);
    lcd.print(capacitance);
    lcd.print(" nF ");
    delay(500);
}
```