Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії
(повне найменування факультету)

Кафедра обчислювальної техніки
(повна назва кафедри)

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему Методи та засіб діагностики якості електролітичних конденсаторів комп'ютерної техніки

Виконав: студент 2 курсу, групи 1КІ-18м
спеціальності:
123 «Комп'ютерна інженерія»
(шифр і назва напряму підготовки, спеціальності)

Кірше А. О.
(прізвище та ініціали)

Керівник к.т.н доц. Роптанов В. І.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

м. Вінниця — 2020 року

Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет: Інформаційних технологій та комп'ютерної інженерії

Кафедра: Обчислювальної техніки

Освітньо-кваліфікаційний рівень: магістр

Спеціальність: 123 «Комп'ютерна інженерія»

(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ, д.т.н., проф.
Т. Б. Мартинюк
«____» 2020 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кірше Андрію Олександровичу

1. Тема роботи: Методи та засіб діагностики якості електролітичних конденсаторів комп'ютерної техніки
керівник проекту: к.т.н., доц. Роптанов В. І.
затверджені наказом вищого навчального закладу від " " 20 року №
2. Строк подання студентом проекту
3. Вихідні дані до роботи. Дослідження методів тестування електролітичних конденсаторів та вибір методу для оперативного тестування; дослідження приладів для вимірювання параметрів конденсаторів; розробка мікропроцесорного пристрою діагностики якості електролітичних конденсаторів; розробка програмного забезпечення для мікропроцесорного пристрою діагностики якості електролітичних конденсаторів
4. Зміст магістерської кваліфікаційної роботи (перелік питань, які потрібно розробити): Вступ. 1 Аналіз методів тестування електролітичних конденсаторів. 2 Засіб діагностики якості електролітичних конденсаторів комп'ютерної техніки. 3 Програмне забезпечення та комп'ютерне моделювання засобу діагностики якості електролітичних конденсаторів комп'ютерної техніки 4. Економічна частина.
5. Графічна частина: схема пристрою діагностики якості електролітичних конденсаторів комп'ютерної техніки, перелік елементів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Технічний	Роптанов Володимир Ілліч, к.т.н., доц. каф. обчислювальної техніки	дата	дата
		підпис	підпис
Економічний	Бальзан Марина Володимирівна, к.е.н., доц. каф. економіки підприємства і виробничого менеджменту	дата	дата
		підпис	підпис
		підпис	підпис

7. Дата видачі завдання: _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Розробка технічного завдання (ТЗ)	10.03.2020 — 13.03.2020	
2	Огляд літературних джерел. Огляд і аналіз існуючих методів та засобів тестування електролітичних конденсаторів. Вибір методу для тестування якості електролітичних конденсаторів	16.03.2020 — 28.03.2020	
3	Огляд та аналіз елементної бази пристрою. Розробка схеми пристрою	30.03.2020 — 10.04.2020	
4	Розробка програмного забезпечення Комп'ютерне моделювання.	13.04.2020 — 24.04.2020	
5	Аналіз економічної ефективності розробки	4.05.2020 — 15.05.2020	
6	Оформлення пояснювальної записки (ПЗ) та графічної частини	18.05.2020 — 29.05.2020	
7	Попередній захист	1.06.2020 — 14.06.2020	
8	Захист	9.06.2020	

Студент _____ Кірше А. О.
(підпис) (прізвище та ініціали

Керівник роботи _____ Роптанов В. І.
(підпис) (прізвище та ініціали)

Методи та засіб діагностики якості електролітичних конденсаторів комп’ютерної техніки

Кірше Андрій

Роптанов Володимир Ілліч

1КІ-18м

Реферат

У даній магістерській кваліфікаційній роботі розглянуті методи діагностики працездатності електролітичних конденсаторів та засіб діагностики якості електролітичних конденсаторів, що дозволить прискорити процес виявлення несправних елементів під час ремонту комп’ютерної техніки, мережевого обладнання, засобів оргтехніки

У першому розділі проведений аналіз параметрів електролітичних конденсаторів та методи вимірювання їх. Обґрунтовано вибір методу вимірювання ESR, як метод для діагностики якості електролітичних конденсаторів.

У другому розділі проведений аналіз методу вимірювання ESR та отримана математична модель засобу діагностики якості електролітичних конденсаторів. Вибрана мікропроцесорна платформа та розроблена схема пристрою діагностики якості електролітичних конденсаторів.

У третьому розділі розглянуті питання програмного забезпечення для мікропроцесорного засобу діагностики якості електролітичних конденсаторів та етапи його розробки. Проведено комп’ютерне імітаційне моделювання пристрою діагностики якості конденсаторів.

У четвертому розділі проведено розрахунок кошторису витрат на виробництво пристрою та ефективність вкладених інвестицій.

Abstract

In this master's qualifying paper, the methods of diagnostics of the health of electrolytic capacitors and diagnostic tool quality electrolytic capacitors that will speed up the process of identifying faulty elements during the repair of computer equipment, network equipment, office equipment

In the first section of the analysis of electrolytic capacitors and methods of measuring them. The choice of the measurement method of ESR as a method for diagnostics of the quality of electrolytic capacitors.

In the second section the analysis of the method of ESR measurement and a mathematical model diagnostics quality electrolytic capacitors. The selected microprocessor platform and developed a diagram of the device of diagnostics of the quality of electrolytic capacitors.

The third section considers the issues of software for microprocessor-based diagnostic tools quality electrolytic capacitors and the stages of its development. A computer simulation of the device diagnostics of the quality of the capacitors.

In the fourth section calculated estimates of production costs of the device and the efficiency of investments.

Зміст

ВСТУП

Актуальність теми полягає в тому, що із-за тривалої роботи імпульсні блоки живлення точок доступу, роутерів Wi-Fi, моніторів виходять з ладу через конденсатори,яківтратилиємність.Електролітичніконденсатори,які працюють у високочастотних імпульсних схемах (блоки живлення, інвертори, перетворювачі, імпульсні стабілізатори) працюють у досить екстремальних умовах та виходять з ладу частіше. Дуже часто при ремонті комп'ютерів та комп'ютерної техніки — в блоках живлення, материнській платі комп'ютера, відеокарти, монітори, принтери та інші пристрої — можна виявити зіпсовані електролітичні конденсатори, в яких витік електроліт.

У будь-якому комп'ютері є безліч електролітичних конденсаторів, які потрібні для фільтрації завад та підтримки стабільної напруги живлення мікросхем, які дуже критичні до рівня та якості напруги живлення.

За рахунок використання електролітичних конденсаторів та котушок індуктивності в схемах вдається забезпечувати необхідні параметри живлення. Після певного відрізка часу експлуатації комп'ютера, ємність електролітичних конденсаторів знижується. Комп'ютерна техніка починає працювати зі збоями, безпричинно зависає, тому треба зробити заміну електролітичних конденсаторів, які несправні.

Погане охолодження негативно позначається не тільки на роботі процесорів та мікросхем, але і на електролітичних конденсаторах.

Аналогічна ситуація спостерігається у блоках живлення персональних комп’ютерів, що несправні — електролітичні конденсатори вздуваються, що призводить до зменшення напруги живлення та підвищення рівня пульсацій.

Мета і задачі дослідження. Метою роботи є дослідження методів та засобу діагностики якості електролітичних конденсаторів комп’ютерної техніки, що дозволяють прискорити процес діагностики працездатності електролітичних конденсаторів та зменшить час виявлення несправних елементів під час ремонту комп’ютерної техніки.

Відповідно до поставленої мети в роботі вирішуються такі задачі:

— аналіз несправностей електролітичних конденсаторів;

— аналіз методів тестування електролітичних конденсаторів;

— обґрунтування методу та засобу діагностики якості електролітичних конденсаторів комп’ютерної техніки;

— удосконалення існуючих пристроїв діагностики працездатності конденсаторів для практичної реалізації пристрою діагностики якості електролітичних конденсаторів комп’ютерної техніки.

Об'єктом дослідження є процес діагностики несправностей елементів комп’ютерної техніки.

Предметом дослідження є методи та засіб діагностики якості електролітичних конденсаторів.

Методи дослідження. Для досягнення поставленої в роботі мети використовуються такі методи дослідження:

— системний аналіз, який застосовується для дослідження методів та засобів тестування електролітичних конденсаторів;

— об'єктно-орієнтовані методи програмування мікроконтролерів AVR;

— формальні методи опису синтаксису мов програмування;

— методи комп’ютерного та натурного моделювання.

Наукова новизна отриманих результатів:

вперше комплексно розглянуто та вдосконалено метод оцінювання ESR як засіб діагностики якості електролітичних конденсаторів.

Практичне значення отриманих результатів:

— удосконалено пристрій тестування електролітичних конденсаторів, який відрізняється застосуванням методу оцінки ESR та сучасної елементної бази, що дозволяє прискорити процес діагностики працездатності електролітичних конденсаторів та достовірність отриманих результатів.

— застосування методу оцінки ESR для практичної реалізації пристрою для діагностики якості електролітичних конденсаторів;

Апробація результатів роботи. Основні положення магістерської роботи опубліковані у міжнародному науковому журналі Polish science journal.

1 АНАЛІЗ МЕТОДІВ ТЕСТУВАННЯ ЕЛЕКТРОЛІТИЧНИХ КОНДЕНСАТОРІВ

У даному розділі розглянуто параметри електролітичних конденсаторів та методи перевірки їх, що дозволило вибрати метод для діагностики якості електролітичних конденсаторів, який необхідний для зменшення часу визначення несправного елементу в комп'ютерній техніці.

1.1 Основні параметри конденсаторів

Основними параметрами, що характеризують конденсатори, є їх електрична ємність та кут втрат.

В електронних пристроях застосовуються конденсатори з ємностями в межах від 1 пФ до 1000 мкФ. Величина похибки визначення ємності конденсатора залежить від сфери застосування останніх. У коливальних системах ємність конденсаторів повинна визначатися з точністю 1%. При виборі конденсаторів блокувальних, розділових, зв'язку допускається значний (до 20-50%) розкид ємностей та вимір їх можна виконувати найпростішими методами.

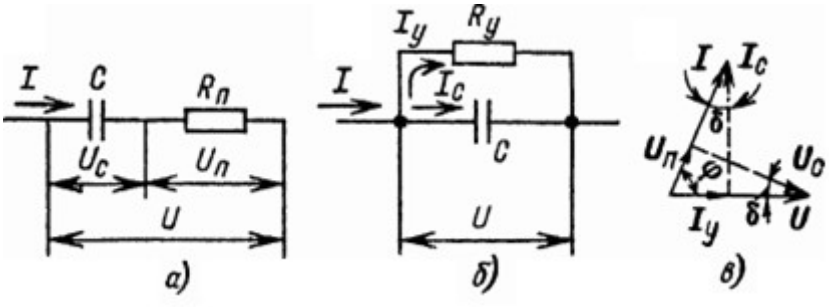


Рисунок 1.1 — Еквівалентні схеми (а, б) і векторна діаграма (в) ланцюга з конденсатором

У кожному конденсаторі, включеному в електричне коло, є втрати енергії, що виникають в матеріалі діелектрика, а також внаслідок недосконалості ізоляції між виводами. З урахуванням втрат еквівалентну схему конденсатора можна уявити в двох варіантах: ємність С, що включена послідовно з опором втрат $R_П$ (рис. 1, а); ємність С, що шунтується опором витоку $R_У$ (рис. 1, б). Між еквівалентними схемами існує формула перерахунку опору втрат в опір витоку:

$$R_У = 1 / ((2 \cdot \pi \cdot f \cdot C)^2 \cdot R_П), \quad (1)$$

де f - частота струму в колі конденсатора.

З векторної діаграми (рис. 1, в), справедливою для обох варіантів еквівалентних схем, що в колі з конденсатором через наявність втрат фазовий зсув φ між струмом I і напругою U завжди менше 90 °. Втрати в конденсаторі зазвичай характеризують кутом втрат $\delta = 90° - \varphi$, що визначаються відповідно до позначень на рис. 1 з формули

$$\operatorname{tg} \delta = U_П / U_C = Iy / Ic = 2 \cdot \pi \cdot f \cdot C \cdot R_п = 1 / (2 \cdot \pi \cdot f \cdot C \cdot R_У). \quad (2)$$

Втрати в конденсаторі іноді характеризують коефіцієнтом потужності $\cos \varphi$ або струмом витоку Iy, що визначаються при стандартних умовах. Для більшості конденсаторів через малі втрати ($\operatorname{tg} \delta < 0{,}001$) можна вважати, що

$$\operatorname{tg} \delta \approx \delta \approx \sin \delta = \sin (90° - \varphi) = \cos \varphi. \quad (3)$$

Найбільші втрати мають місце в електролітичних та паперових конденсаторах, застосування яких в основному обмежується областю низьких частот.

З підвищенням частоти втрати помітно зростають, тоді як ємність С практично не залежить від частоти. Параметри конденсатора (С, $R_П$, $R_У$, δ) залежать від прикладеної до нього напруги та умов експлуатації — температури, вологості, атмосферного тиску.

1.2 Класифікація методів перевірки працездатності конденсаторів

Найпростіші перевірки конденсаторів можна виконувати і без застосування вимірювача параметрів конденсатора. Омметром або пробником легко виявити пробій між обкладинками конденсатора (пробій іноді проявляється тільки при напрузі на конденсаторі, яка близька до його робочій напрузі). Перевірка на обрив неелектролітичних конденсаторів з ємністю більше 0,01 мкФ проводиться вмиканням конденсатора в коло змінного струму, послідовно з будь-яким навантаженням — лампою розжарювання, гучномовцем. Нормальне або трохи ослаблене світіння лампи або звучання

радіопередачі буде свідчити про відсутність обриву [10].

Конденсатор, опір витоку якого великий, утримує тривалий час отриманий ним заряд; це дозволяє оцінити якість конденсаторів з ємністю вище за 0,01 мкФ простими засобами. При перевірці такого конденсатора омметром, його стрілка відхилиться, так як конденсатор заряджається, а потім повернеться у вихідне положення, якщо він має великій опір витоку. Наступні короткочасні підключення до конденсатору омметра, повторювані з інтервалом в декілька секунд, не повинні викликати відхилення стрілки вимірювача. При малому опорі витоку помітне відхилення стрілки буде спостерігатися при кожному підключенні омметра. Для перевірки на витік конденсаторів ємністю більше 100 пФ можна застосувати головні телефони, з'єднані послідовно з низьковольтної батареєю. При малому опорі витоку кожне підключення індикатора до конденсатору викликає клацання в телефонах, тоді як при хорошому конденсаторі клацання прослуховується лише при першому підключенні. Вимірювання значення опору витоку (на постійному струмі) може проводитися індукторними або електронними мегомметрами [8].

Електролітичні конденсатори слід приєднувати до випробувального приладу з урахуванням полярності включення джерела живлення. Під час вимірювання опору витоку таких конденсаторів рекомендується відлік здійснювати через 10 хвилин після їх включення під напругу, коли процес заряду можна вважати завершеним.

Для вимірювання параметрів конденсаторів застосовуються методи вольтметра - амперметра, безпосереднього вимірювання за допомогою вимірювача ємності (мікрофарадометр), порівняння (заміщення), мостовий, резонансний, вимірювання ESR (еквівалентного послідовного опору $R_П$)

1.3 Метод вольтметра - амперметра

Метод вольтметра - амперметра застосовують для вимірювання великих ємностей. Живлення вимірювальної схеми здійснюють від генератора низької частоти: F = 50 ... 1000 Гц, тому можна знехтувати активними втратами в

конденсаторах та впливом вимірювальних приладів і наводками.

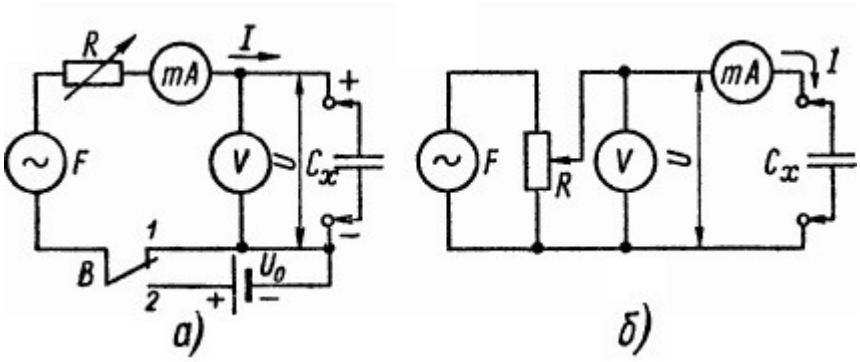


Рисунок 1.2 — Схеми вимірювання ємності методом вольтметра-амперметра

Схема вимірювань представлена в двох варіантах на рис. 1.2. Конденсатор Сх, що перевіряється, включається в коло змінного струму відомої частоти *F*, і потенціометром *R* встановлюють зручну для відліку значення струму *I* або напруги *U*. За показниками приладів змінного струму (вольтметр та міліамперметр) можна розрахувати повний опір конденсатора

$$Z = (R^2 + X^2)^{1/2} = U / I, \quad (4)$$

де R і $X = 1 / (2 \cdot \pi \cdot F \cdot C_X)$ — відповідно його активна і реактивна складові.

Якщо втрати малі $R << X$, то ємність визначається з формули

$$C_X = I / (2 \cdot \pi \cdot F \cdot U). \quad (5)$$

Схема на рис. 1.2, *а*, дає досить точні результати при вимірюванні великих ємностей, опір яких X значно менше вхідного опору вольтметра V. Схема на рис. 1.2, *б*, застосовується для вимірювання ємності, опір яких в десятки і більше разів перевищує опір міліамперметра. Схема на рис. 1.2, а може бути застосована для вимірювання ємності електролітичних конденсаторів. Якщо напруга живлення не перевищує 1-2 В, то вимір допустимо проводити при установці перемикача *В* у положення 1. При великих змінних напругах можливе пошкодження конденсаторів внаслідок розкладання електроліту. Ця небезпека усувається, якщо перемикач *В* встановити в положення 2. При цьому послідовно з джерелом змінного струму частоти F включається джерело постійного струму, напруга на затискачах якого U_0 має перевищувати амплітуду змінної

напруги. Тоді в колі буде діяти пульсуюча напруга, безпечна для конденсатора за умови правильної полярності його включення в схему. Пульсуючу напругу можна отримати при послідовному включенні в вимірювальну схему діода. У всіх випадках вольтметр і міліамперметр повинні вимірювати лише змінні складові напруги та струму, для чого вони виконуються із закритою схемою входу [8].

1.4 Метод мікрофарадометру

На рис. 1.3, *а* представлений один з варіантів паралельної схеми мікрофарадометру. При вільних вхідних затискачах (що еквівалентно ємності $C_X = 0$) регулюванням чутливості вольтметра V домагаються відхилення стрілки його вимірювача до кінця шкали. Включення в схему конденсатора C_X призводить до того, що напруга на вольтметрі з $U_П$, знижується до значення U_X.

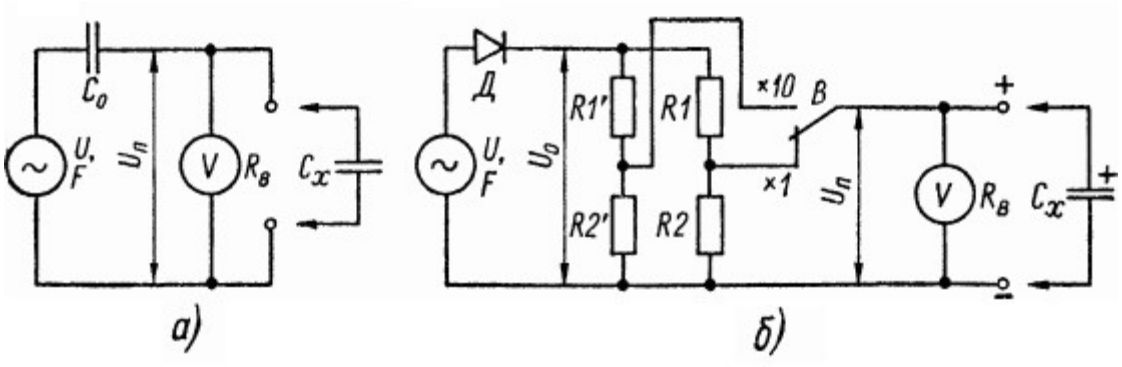


Рисунок 1.3 — Паралельні схеми мікрофарадометру

Характеристика градуювання мікрофарадометру визначається формулою

$$U_X / U_П \approx C_O / (C_O + C_X). \quad (6)$$

Вхідний опір вольтметра R_B та частота струму живлення F обмежують вибір опорної ємності конденсатора C_O , який визначає середнє значення шкали, умовою

$$C_O \geq 1{,}5 / (F \cdot R_B). \quad (7)$$

Для вимірювання ємності електролітичних конденсаторів придатна схема на рис. 1.3, *б*. Завдяки включенню діода Д на дільнику напруги R1, R2 діє пульсуюча напруга U_O. При $C_X = 0$ з резистора R2 на вольтметр V подається напруга повного відхилення $U_П$. Включення конденсатора C_X призводить до

зниження напруги на вольтметрі відповідно до формули (6). При обраному середньому значенні шкали ємності C_O і частоті F = 50 Гц необхідні значення опорів дільника напруги визначаються формулами:

$$R1 = U_O / (U_П \cdot 180 \cdot C_O); R2 = R1 \cdot U_П (U_O — U_П). \quad (8)$$

Похибка вимірювання ємності таким методом складає 5-10% [1, 8]. Тому при градуюванні використовують набори конденсаторів з допусками по ємності не більше 5%.

1.5 Метод вимірювання середнього значення струму розряду

Вимірювач ємності з рівномірною шкалою базується на вимірюванні середнього значення струму заряду або розряду перевіряється конденсатора, що перезаряджається напругою відомої частоти.

На рис. 1.4, наведена схема вимірювального блоку, що живиться імпульсною напругою прямокутної форми. Під час дії імпульсу через діод Д відбувається заряд конденсатора C_X до максимальної напруги U_M. В інтервалі між імпульсами конденсатор розряджається через вимірювач *И* до початкової напруги U_H. У сталому режимі при частоті повторення вхідних імпульсів *f* і їх амплітуді $U_П = U_M — U_H$ середнє значення протікає через вимірювач струму

$$I_X = C_X \cdot U_П \cdot f. \quad (9)$$

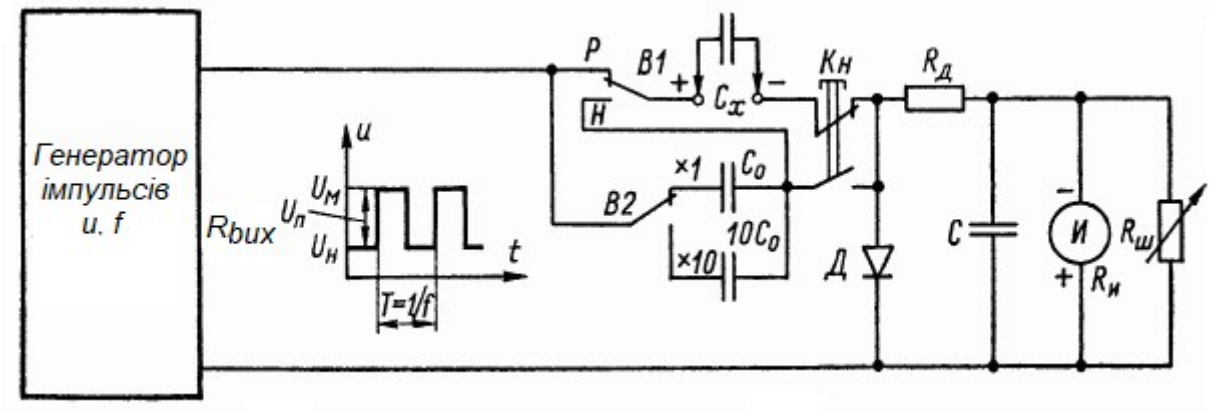


Рисунок 1.4 — Схема вимірювального блоку з рівномірною шкалою

При фіксованих значеннях $U_П$ та *f* вимірювач можна забезпечити рівномірною шкалою з відліком у значеннях C_X відповідно до формули (9)

$$C_X = I_X / (U_П \cdot f). \quad (10)$$

Перед початком вимірювань проводитися калібрування, для чого до нього натисканням кнопки Кн приєднують конденсатор ємністю $C_O = C_П$ (рис. 1.4); при цьому домагаються відхилення стрілки вимірювача до кінця шкали за допомогою плавного регулювання частоти *f*, амплітуди імпульсів $U_П$ або чутливості вимірювача (реостат $R_Ш$). Оскільки шкала приладу рівномірна, то похибка вимірювання ємності в основному визначається похибкою підбору опорної ємності C_O , відхилення якої від необхідного номіналу ($C_П$) не повинно перевищувати 1 ... 5%.

Для отримання правильних результатів вимірювань необхідно, щоб за один період вхідного напруги конденсатор C_X встигав повністю зарядитися і розрядитися. Найлегше це забезпечується при прямокутній формі вхідних імпульсів і належному виборі частоти їх повторення *f*.

Як відомо, в колі, що складається з елементів R і C, тривалість заряду (розряду) конденсатора C до значення прикладеної до цього кола постійної напруги визначається постійної часу $\tau = RC$ і практично не перевершує 5τ. Для того щоб заряд (розряд) закінчувався протягом напівперіоду $T/2$ напруги частоти *f*, необхідно виконання умови

$$5RC = 5\,\tau \le T / 2 = 1 / (2 \cdot f),$$

яке задовольняється при частоті

$$f \le 1 / (10 \cdot RC). \quad (11)$$

Взявши максимально можливий опір кола заряду і розряду R = 10 кОм (з урахуванням вихідного опору Rвих генератора імпульсів), отримуємо практичну формулу для вибору частоти повторення імпульсів (в кілогерцах):

$$f\,[кГц] \le 10^4 / C_П\,[пФ]. \quad (12)$$

Вимірювачі ємності розглянутого типу мають верхні межі вимірювання $C_П$ не менше 100 пФ через труднощі генерування прямокутних імпульсів з високою частотою повторення і впливу паразитного зв’язку. Труднощі

виникають і при розширенні діапазону вимірювань в бік великих ємностей.

Крім рівномірної шкали ємності, такі прилади можуть мати нерівномірну шкалу з діапазоном показань від 0 до ∞, подібну шкалами паралельних схем омметрів. Характер шкали (рівномірна - Р, нерівномірна - Н) в схемі на рис. 1.4, визначається установкою перемикача *B1* . У положенні «Н» випробуваний конденсатор C_X включається послідовно з опорним конденсатором C_O, ємність якого задає межа вимірювань приладу й приблизно відповідає середині його нелінійної шкали.

1.6 Метод порівняння

Даний метод базується на порівнянні режиму роботи вимірювальної схеми, коли підключається вимірювана ємність C_X та зразкова ємність C_O.

Схема вимірювань, в якій порівнюються опір змінному струму ємностей C_X та C_O, наведена на рис. 1.5. При включенні конденсатора C_X потенціометром R встановлюють в колі струм, зручний для відліку або контролю за міліамперметром змінного струму *mA* або іншому низькоомному індикатору. Потім замість конденсатора C_X приєднують до схеми набір конденсаторів і підбором їх ємності C_O домагаються попереднього показання індикатора. Це буде коли $C_O = C_X$. На точність вимірювання впливає чутливість індикатора та похибки визначення ємності C_O ; вона може бути отримана менше 1% [8, 13].

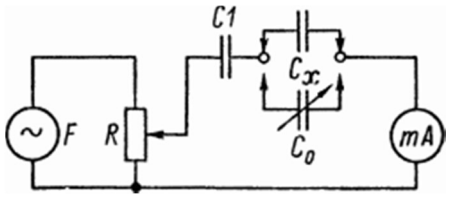


Рисунок 1.5 — Схема вимірювання ємності методом порівняння

Метод порівняння застосовується в резонансних та мостових вимірювачах ємності.

1.7 Мостовий метод

Вимірювальні мости бувають магазинного та лінійного

типу. Найпростіший магазинний міст, придатний для вимірювання ємності в десятки і сотні пФ, може бути складений з чотирьох конденсаторів: вимірюваного, змінного зі шкалою ємності і двох постійних з однаковою ємністю. При використанні в якості індикатора головних телефонів джерелом живлення моста може служити радіотрансляційна мережа. Широкодіапазонні магазинні мости складніше лінійних, однак вони забезпечують меншу похибку вимірювання й можуть мати рівномірні шкали. Діапазон ємності, вимірюваних мостовим методом, лежить приблизно в межах від 10 пФ до 10 ... 30 мкФ.

На рис. 1.6, *а* наведена схема багато діапазонного магазинного моста. Його врівноважують за допомогою конденсатора змінної ємності С1 і змінного резистора R1. Застосовуючи до даної схемою умова рівноваги, отримуємо

$$R2 \cdot (R_X{}^2 + 1 / (2 \cdot \pi \cdot F \cdot C_X)^2)^{0,5} = R_3 \cdot (R_1{}^2 + 1 / (2 \cdot \pi \cdot F \cdot C_1)^2)^{0,5} \qquad (13)$$

З огляду на, що $\varphi_2 = \varphi_3 = 0$, друга умова рівноваги можна записати у вигляді рівності $\varphi_X = \varphi_1$ або $tg\ \varphi_X = tg\ \varphi_1$:

$$1 / (2 \cdot \pi \cdot F \cdot C_X \cdot R_X) = 1 / (2 \cdot \pi \cdot F \cdot C_1 \cdot R_1). \qquad (14)$$

Розв’язавши наведені вище рівняння, знаходимо:

$$C_X = C1 \cdot (R2 / R3) \qquad R_X = R1 \cdot (R3 / R2). \qquad (15)$$

При фіксованому відношенні опорів плечей R2 / R3 конденсатор С1 і резистор R1 можна забезпечити шкалами з відліком відповідно в значеннях ємності C_X і опорів втрат R_X.

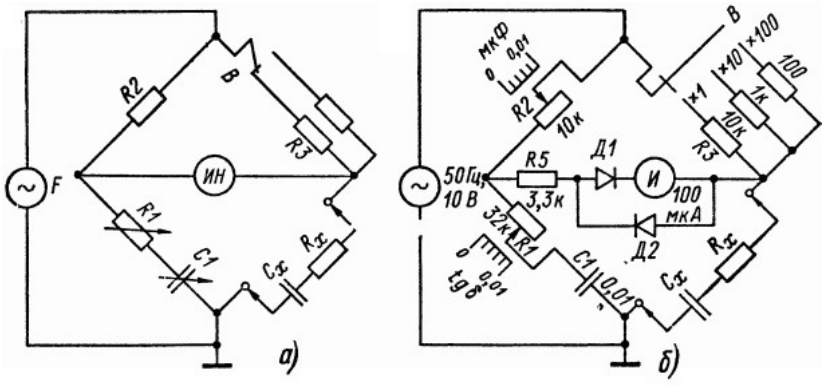


Рисунок 1.6 — Схеми магазинних мостів для вимірювання параметрів конденсаторів

Досить часто в вимірювальних мостах використовується конденсатор С1 постійної ємності, а як елементи регулювання — змінні резистори R1 і R2 (рис. 1.6, б). З формул (15) випливає, що обидві регулювання такого моста виявляються взаємопов'язаними, тому його урівноваження, за показаннями індикатора, має здійснюватися способом послідовного наближення до мінімуму шляхом поперемінного зміни опорів R1 і R2. Значення ємності C_X знаходяться за шкалою резистора R2 з урахуванням множника, який визначається установкою перемикача B. Оскільки безпосередня оцінка опорів втрат R_X виявляється неможливою, то відлік по шкалі резистора R1 виконується в значеннях тангенса кута втрат:

$$\operatorname{tg}\delta = 2 \cdot \pi F \cdot C_X \cdot R_X = 2 \cdot \pi \cdot F \cdot C_1 \cdot R_1 , \qquad (16)$$

який при фіксованій частоті F однозначно визначається значенням опору R1.

Для виключення впливу паразитного зв'язку та похибок самого моста мостовий метод вимірювання ємності часто поєднують з методом порівняння.

1.8 Резонансний метод

Резонансна схема вимірювання ємності (рис. 1.7) включає в себе генератор високої частоти, з контуром якого LC слабо зв'язується індуктивно (або через ємність) вимірювальний контур, що складається з опорної котушки індуктивності L_O та випробуваного конденсатора C_X. Зміною ємності конденсатора С генератор налаштовують в резонанс із власною частотою f_O вимірювального контуру з екстремальних показаннями індикатора резонансу, наприклад електронного вольтметра V. При відомій частоті настройки генератора f_O ємність, що вимірюється визначається за формулою

$$C_X = 1 / ((2 \cdot \pi \cdot f_O)^2 \cdot L_O) \approx 0{,}0253 / (f_O^2 \cdot L_O) \qquad (17)$$

При фіксованому значенні L_O конденсатор С можна забезпечити шкалою з відліком в значеннях ємностей C_X.

Межі вимірювань ємності визначаються значенням індуктивності L_O і діапазоном частот генератора. Ємності більш 0,01-0,05 мкФ резонансним

методом не вимірюються, так як на низьких частотах резонансні криві коливальних контурів стають тупими, що ускладнює фіксацію резонансу.

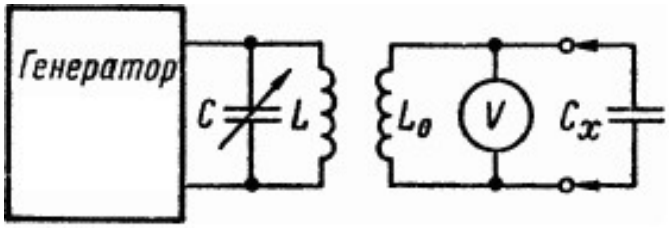


Рисунок 1.7 — Схема вимірювання ємності резонансних методом

1.9 Метод вимірювання ESR

У багатьох електронних пристроях, де застосовуються електролітичні конденсатори, головним критерієм їх справності є мале значення ESR (Equivalent Series Resistance). ESR — один із параметрів конденсатора, що характеризує його активні втрати в колі змінного струму. В еквіваленті його можна уявити, як включений послідовно з конденсатором резистор (рис. 1.1, а), опір якого визначається діелектричними втратами, опіром обкладок, опором внутрішніх контактних з'єднань та виводів.

$$ESR = Ral + Re + Rox + Rb, \quad (18)$$

де Ral — опір алюмінієвих обкладок,

Re — опір електролітичної обкладки,

Rox — опір діелектрика,

Rb — опір виводів.

У цьому випадку кут зсуву фаз між струмом і напругою менше 90°. ESR є важливим параметром, оскільки він передбачає наявність пульсації вихідного сигналу імпульсних перетворювачів комп’ютерної техніка, а також вказує на термін служби конденсатора [2, 7, 11, 22, 23]. Потужність, що розсіюється в ESR, викликає підвищення температури конденсатора і зменшення його ємності і терміну служби.

Типовий конденсатор може бути змодельований як ідеальний конденсатор послідовно з резистором — еквівалентний послідовний опір (рис. 1.1, а).

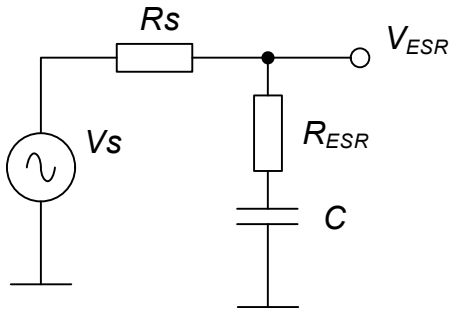


Рисунок 1.8 — Схема вимірювання ESR конденсатора

Схему можна розглядати як простий резистивний дільник, якщо частота джерела змінного струму досить висока, оскільки реактивний опір конденсатора обернено пропорційний частоті практично для будь-якої ємності. Таким чином, можемо використовувати значення вимірюваної напруги на конденсаторі для розрахунку ESR:

$$V_{ESR} \approx V_S \frac{R_{ESR}}{R_{ESR} + R_S} \quad \rightarrow \quad R_{ESR} \approx \frac{R_S}{\frac{V_S}{V_{ESR}} - 1} \quad \approx \alpha R_S, \qquad (19)$$

де α — коефіцієнт згасання (відношення вихідної напруги до вхідної),

R_S — вхідний опір джерела сигналу.

Так як електролітичні конденсатори є поляризованими, то можемо використовувати напругу змінного струму з фіксованим значенням постійного струму або просто використовувати змінну напругу досить низького рівня. Більшість ESR метрів [9] використовують саме цей другий підхід, оскільки він простий в реалізації і не потрібно враховувати полярність під час вимірювання. Вибирається напруга 100 мВ тому, що вона нижче прямої напруги на p/n-переходу, тому можна виконати вимірювання ESR безпосередньо в схемі.

З одного боку більш високі частоти краще для вимірювання ESR через зниження реактивного опору, але не завжди це бажано. Як правило для великих електролітичних конденсаторів (тисячі мікрофарад) частота вимірювань становить від 1 до 5 кГц, а для невеликих конденсаторів (одиниці — сотні мікрофарад) для діагностики використовується частота від 10 до 50 кГц. Замість використання синусоїдальної форми можемо використовуватись прямокутний

сигнал.

Якщо конденсатор ємністю C заряджати від джерела постійного струму I, напруга на його виводах буде лінійно зростати від значення напруги на опорі конденстора (ESR) U_R за законом:

$$C\ dU/dt = I = const. \quad (20)$$

Ємність конденсатора буде визначатись виразом:

$$C = I\ dt/dU. \quad (21)$$

Якщо підрахувати час заряду від значення напруги U1 до U2, взяв значення U2 вдвічі більшим за U1 (рис. 1.9), то розрахунок ємності буде таким:

$$C = I\frac{t_2 - t_1}{U_2 - U_1} = I\frac{t_2 - t_1}{U_1}\ ;\ ESR = \frac{U_1(t_2 - t_1)}{I(t_2 - t_1)} \quad (22)$$

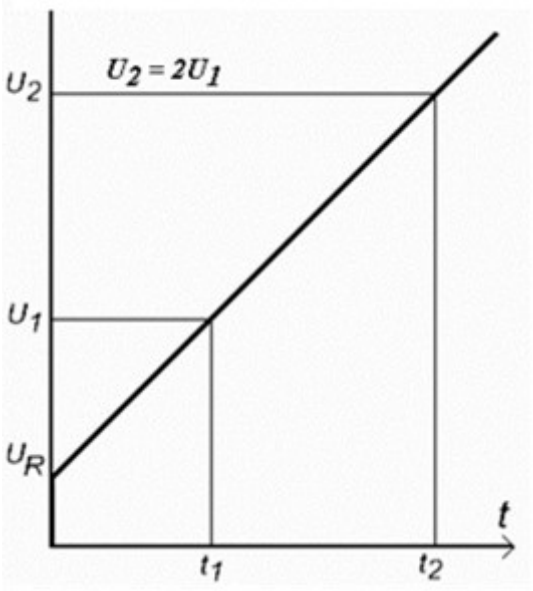


Рисунок 1.9 — Визначення C та ESR за часом розряду/заряду

Потрібно враховувати, що ESR, який вимірюється на постійному струмі, є відносним показником якості електролітичного конденсатора. Значимою складовою ESR є діелектричні втрати, які істотно змінюються зі зміною частоти змінного струму.

2 ЗАСІБ ДІАГНОСТИКИ ЯКОСТІ ЕЛЕКТРОЛІТИЧНИХ КОНДЕНСАТОРІВ КОМП'ЮТЕРНОЇ ТЕХНІКИ

У комп'ютерній техніці, де застосовують електролітичні конденсатори, головним критерієм їх справності є мале значення ESR. Величина його у якісних електролітичних конденсаторів мала та, в залежності від типу та величини ємності конденсатора, може знаходиться в межах від сотих часток до 10…20 Ом. Підвищений внутрішній опір конденсатора може приводити до значного погіршення параметрів комп'ютерної техніки та повній її відмови. У даному розділі розглядаються апаратні засоби, що дозволяють розробити пристрій діагностики якості електролітичних конденсаторів комп'ютерної техніки

2.1 Аналіз методу вимірювання ESR

Еквівалентний послідовний опір ESR можна визначити декількома способами, що наведені у розділі 1. Найбільш часто застосовують спосіб вимірювання ESR та ємності конденсатора, що передбачає вимірювання тривалості зарядки конденсатора до відомого значення напруги з вирахуванням значення ємності за формулою:

$$C = \frac{I \cdot t}{U}, \tag{23}$$

де t — час зарядки конденсатора струмом I до напруги U.

На результат вимірювання впливає опір втрат R_{ESR}, тому рекомендується використовувати метод двох інтервалів часу t_1 та t_2, коли напруга на конденсаторі досягне значення між U_1 та U_2 при сталому значенню I.

$$U_1 = I \cdot (R_{ESR} + {}^{t_1}\!/\!_{C}); \tag{24}$$

$$U_2 = I \cdot (R_{ESR} + {}^{t_2}\!/\!_{C}). \tag{25}$$

Виміряв їх можна визначити величину ємності конденсатора та його ESR:

$$C = I \frac{t_2 - t_1}{U_2 - U_1}; \tag{26}$$

$$R_{ESR} = \frac{U_1 t_2 - U_2 t_1}{I(t_2 - t_1)} . \quad (27)$$

Відомо, що модуль повного комплексного опору конденсатора дорівнює:

$$z = \sqrt{R_{ESR}{}^2 + \frac{1}{(2\pi FC)^2}} \quad (28)$$

На частоті в межах 50-100кГц модуль повного комплексного опору буде дорівнювати R_{ESR}. Тобто виміряв значення повного комплексного опору конденсатора на частоті F, то фактично вимірюється значення $R_{ESR.}$

2.2 Структурна схема пристрою діагностики якості конденсаторів

Так як електролітичні конденсатори є поляризованими, то для їх діагностики використовують змінну напруги невеликої амплітуду, що дозволяє здійснювати випробування без випаювання конденсатора. Форма напруги для діагностики конденсатора може бу-ти як синусоїдальною, так і прямокутної форми. Реактивний опір за рахунок індуктивності в колі зростає пропорційне частоті вхідного сигналу й ця реактивність може значно спотворити результат вимірювання. Тому для електролітичних конденсаторів великої ємності, частоти для яких їх можна використовувати — від 1 до 5 кГц, а для невеликих електролітичних конденсаторів — від 10 до 50 кГц. Значення ESR можна визначити вимірявши час розряду конденсатору між фіксованими значеннями напруги. Математичні обчислення ємності та ESR можна реалізувати мікроконтролером з виводом інформації на цифровий індикатор [7, 16]. Структурна схема вимірювача ESR наведена на рис. 2.1.

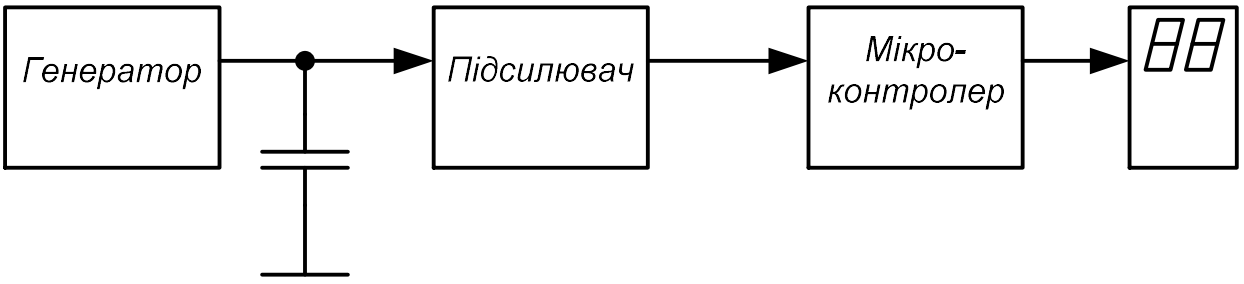


Рисунок 2.1 — Структурна схема вимірювача ESR

Генератор формує прямокутну напругу, яка прикладається до конденсатора, що перевіряється. Якщо $R_{ESR.}$ конденсатора мало, то амплітуда на виході підсилювача дорівнює нулю. При підключенні конденсатора з великим $R_{ESR.}$ амплітуда сигналу збільшиться. Після підсилення та вимірювання постійної напруги мікроконтролером та перерахунку значення в $R_{ESR.}$, воно відображається на індикаторі.

2.3 Вибір мікропроцесорної платформи для пристрою діагностики якості конденсаторів

Щоразу на початку розробки вибирається та елементна база, яка найкращим чином підходить для конкретної задачі. Складній системі може знадобитися кілька 32-розрядних мікроконтролерів, для відносно простого проекту досить функціональних можливостей 8-розрядних мікроконтролерів. Крім того, може знайтися місце для 8-розрядних мікроконтролерів і в складному виробі для управління локальним виконавчим механізмом. У будь-якому випадку при виборі необхідно оцінити енергоефективність та економічність проектованого виробу.

Можливість використання операційної системи у високопродуктивних 32-розрядних мікроконтролерів вважається важливою перевагою. Дійсно, операційна система дозволяє реалізувати кілька завдань на одному 32-розрядних мікроконтролерів. Однак, як правило, з точки зору енергоефективності такий підхід не найкращий. Наприклад, при обміні з зовнішніми пристроями мікроконтролер змушений проводити в режимі очікування багато порожніх циклів. Одним із способів уникнути простою є використання переривань, але при цьому потрібно зберігати в стеку поточний стан, а після обробки переривання повертатися до виконання перерваного завдання, для чого потрібні багато циклів тактової частоти. Та ж ситуація і з перериваннями від зовнішніх пристроїв.

32-розрядних мікроконтролер зазвичай використовує кеш-пам'ять невеликого обсягу, яка розміщується в процесорному ядрі та дозволяє збільшити

продуктивність. Однак при переході в режим зниженого енергоспоживання дані в кеш-пам'яті зберігаються та після пробудження, при переході в активний стан, потрібно їх поновити, на що також витрачається додаткова енергія.

Інша організація у 8-розрядних мікроконтролерів. У більшості випадків у них відсутня кеш-пам'ять, а програма зберігається в енергонезалежній пам'яті. Найчастіше, у флеш-пам'яті. При виконанні програми процесорне ядро звертається до флеш-пам'яті. Така організація зменшує продуктивність, але не вимагає енерговитратних передач даних між флеш-пам'яттю та оперативною пам'яттю. Як випливає з наведених вище прикладів, зменшення продуктивності у 8-розрядних мікроконтролерів при цьому практично не має значення для інтернету речей і деяких інших програм.

Отже, 8-розрядні мікроконтролери нерідко знаходять застосування в системах, де не останню роль відіграє низьке енергоспоживання. Навіть у випадках, коли вимоги до споживання не так критичні, будь-який досвідчений розробник постарається його зменшити, щоб уникнути проблем з джерелами живлення, відведенням тепла та електромагнітною сумісністю. Обробка сигналу може відбуватися і без участі процесорного ядра. Нерідко для цього використовується незалежна від ядра периферія (CIP). Наприклад, якщо потрібно визначити, перевищив зовнішній сигнал допустимі межі, не обов'язково оцифровувати його в АЦП для подальшого порівняння з опорним сигналом — достатньо задіяти вбудований аналоговий компаратор з відповідним опорною напругою.

Вбудовані в мікроконтролер незалежні від ядра периферійного пристрою не тільки зменшують енергоспоживання мікроконтролера, але і дозволяють скоротити число зовнішніх компонентів на платі і, отже, специфікацію. Крім того, ця периферія підвищує продуктивність мікроконтролерів, звільняючи його ресурси для вирішення завдань програми.

Для завдань, орієнтованих на введення/виведення даних, 8-бітні мікроконтролери, як правило, ефективніше використовують пам'ять.

Використання багатопроцесорного рішення може ускладнити проектування, через синхронізацію роботу декількох процесорних ядр.

У більшості випадків розробник зробить вибір, ґрунтуючись на вимогах конкретного проекту, а також на своїх особистих пристрастях та досвіді.

Популярність того чи іншого сімейства мікроконтролера визначається багатьма факторами. Одним з непрямих ознак популярності електронних компонентів є динаміка відвідуваності спеціалізованих сайтів в Інтернеті.

Оскільки мікроконтролер є таким же електронним компонентом, як і будь-яка інша мікросхема, то можна проводити аналіз по фірмам-виробникам, за роками випуску, за сімейством, архітектурою. Аналіз інтернет ресурсів показує, що 8-бітні AVR тримають лідерство (32%) [3, 12, 15].

Для роботи у схемі пристрою діагностики якості електролітичних конденсаторів вибираємо мікроконтролер сімейства AVR ATtiny2313. Розглянемо його архітектуру.

2.4 Архітектура мікроконтролера ATtiny2313

Мікроконтролер ATtiny2313 виконаний за RISC архітектурою. Це високопродуктивний 8-бітний AVR мікроконтролер з малим рівнем енергоспоживання з параметрами [14, 15, 18]:

— 32 регістри загального призначення;

— 1 Кбайт Flash-пам'яті з можливістю ISP програмування;

— 128 байт енергонезалежної пам'яті EEPROM з підтримкою ISP програмування;

— 8 та 16 розрядний таймер/ лічильник з пре дільником частоти; три вихідних канали з високошвидкісним 8-мі розрядним ШІМ; аналоговий компаратор;

— 18 лінії введення/виведення інформації.

Архітектура мікроконтролера Attiny2313 наведена на рисунку 2.2. Спеціалізовані функції мікроконтролера:

— функція внутрісистемного програмування по SPI- порту;

— удосконалена функція ініціалізації при включенні живлення;

— коло виявлення аварійного відключення живлення;

— зовнішні й внутрішні джерела переривання;

— 3 режими пониженого енергоспоживання;

— напруга живлення 2,7—5,5В.

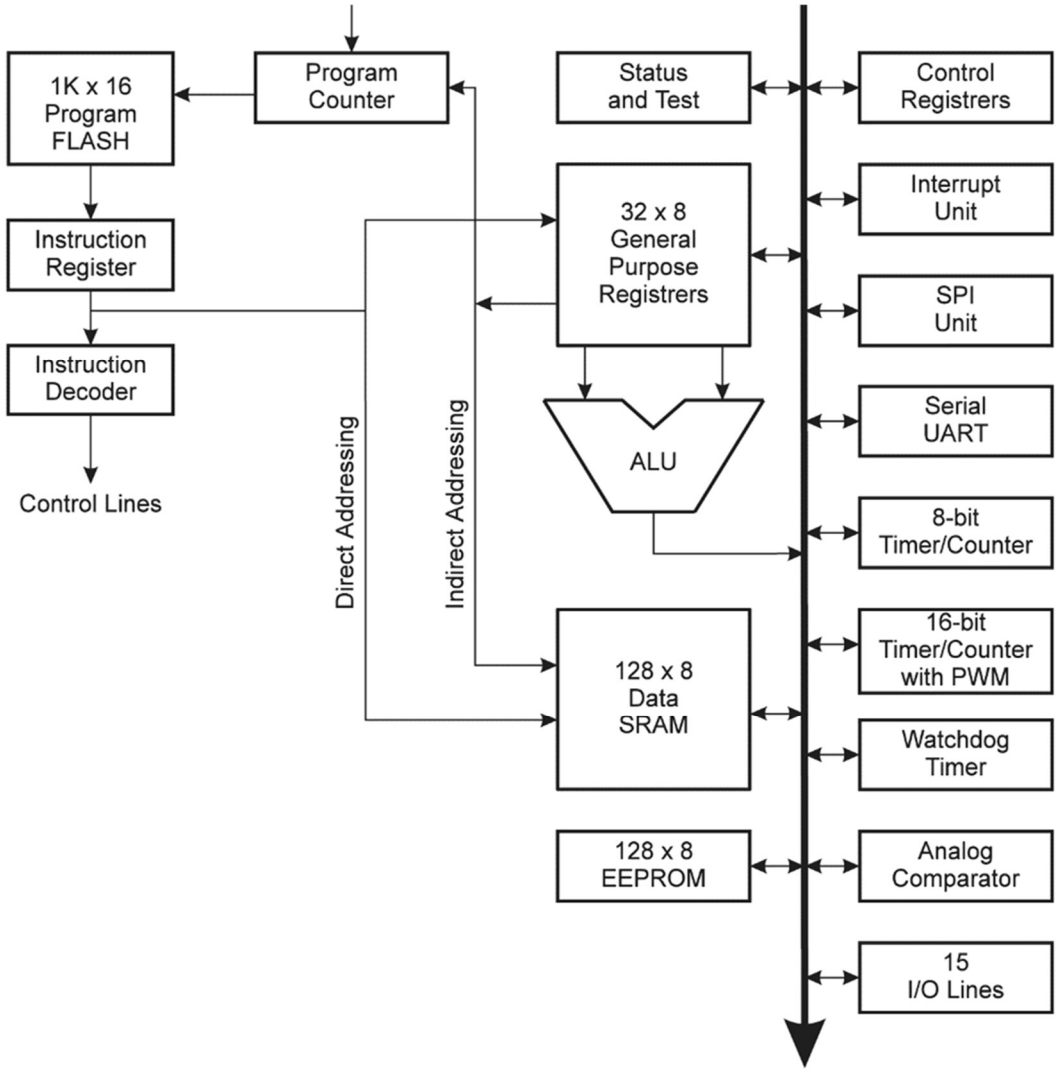


Рисунок 2.2 — Архітектура мікроконтролера ATtiny2313

Мікроконтролери AVR мають компаратор, який вимірює аналогову вхідну напругу на двох входах та видають цифровий вихідний сигнал (0 або 1) в залежності від того, на якому вході (позитивному (PB0) чи негативному (PB1)) є напруга. Лінії аналогового компаратора є сумісними та можуть налаштовуватися як цифрові виходи. На рис. 2.3 показана блок-схема компаратора.

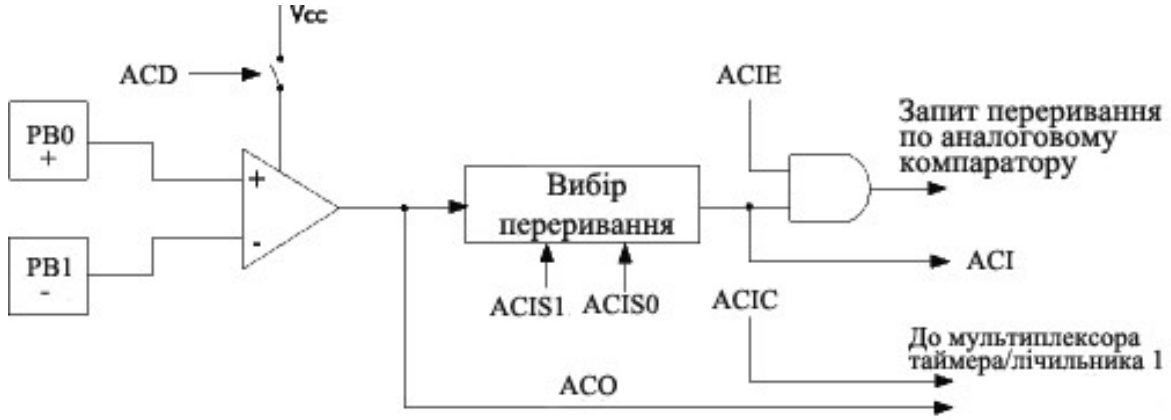


Рисунок 2.3 — Структурна схема аналогового компаратора

2.5 Схема пристрою діагностики якості конденсаторів комп'ютерної техніки

Алфавітно-цифрові LCD-модулі повноцінно формують текстові написи та просту графіку. В них використовуються матриці точок 8x8 та внутрішній контролер.

На практиці популярними стали символьні LCD-модулі з контролером HD44780 (фірма Hitachi) [12, 15].

Екран символьного LCD-модуля складається з декількох рядків (1, 2, 4) з 8, 10, 12, 16, 20, 24, 40 знакомісць. Найбільш поширеним у практиці є LCD-модуль 16x2 (16 символів у двох рядках).

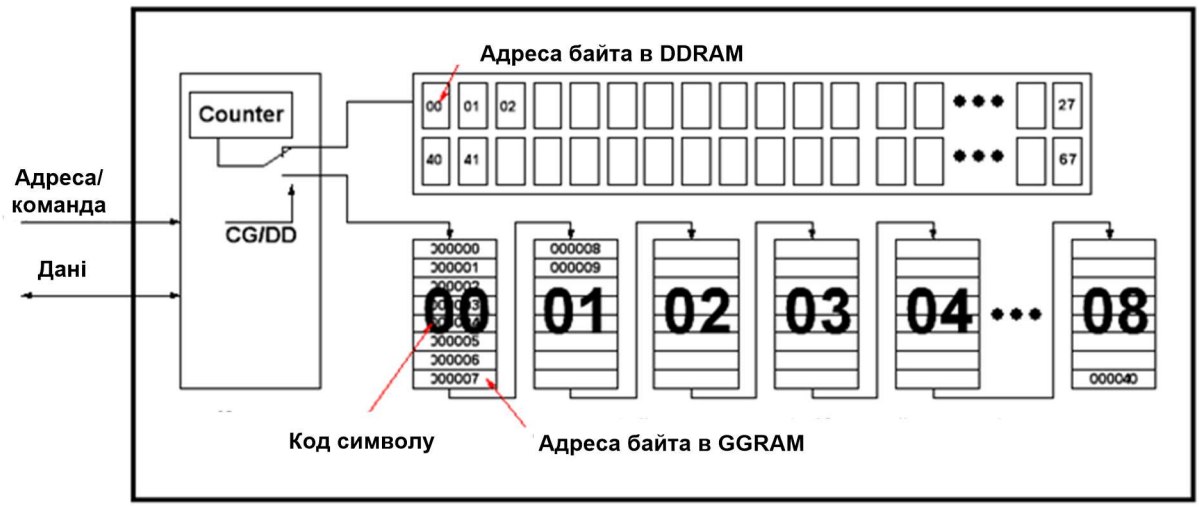


Рисунок 2.4 — Структурна схема LCD-модуля

Перевагою символьних LCD-модулів є простота керування. Усі сигнали відповідають CMOS-рівням, тому індикатор для мікроконтролера, представляється як зовнішня логічна CMOS-мікросхема. Розробнику необхідно правильно підключити сигнальні та інформаційні виводи та передавати по ним сигнали керування. Вбудований в LCD-модуль Controller / Driver поєднує функції пристрою керування та силового драйвера для матриці точкових елементів.Внутрішній RC-генераторon з частотою 200 ... 300 кГц забезпечує роботу LCD-модуля. В комірках пам'яті CGROM записаний знакогенератор. Вільні комірки CGRAM користувач може використовувати для виведення на екран LCD власних символів 8x8.

Призначення виводів алфавітно-цифрових LCD-модулів з контролером HD44780 уніфікована для будь-якої кількості рядків та стовпців.

Призначення сигналів: «DB0» ... «DB7» — восьмирозрядна шина даних; «RS» — сигнал вибору команди / даних (LOW / HIGH); «R / W» — режим роботи індикатора (читання / запис); «E» — дозвіл роботи; «Vo» — напруга для регулювання контрастності; Vcc — живлення +5 В (+3.3 В); GND — загальний вивід; «A», «K» — анод та катод підсвічування екрану.

Розрізняють чотири (рис. 2.5) та восьми (рис. 2.6) бітний режим роботи «алфавітно-цифрових» LCD-модулів. Для програміста LCD-модуль є набір регістрів, в які можна записувати та зчитувати інформацію.

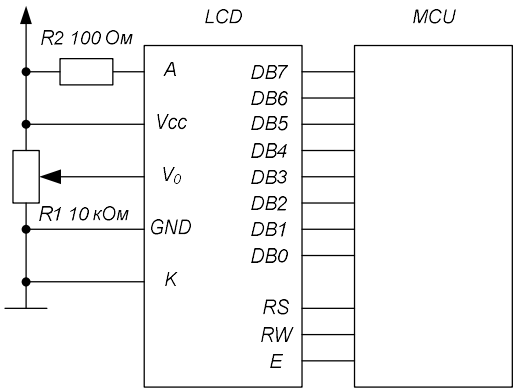


Рисунок 2.5 — Підключення LCD індикатора на базі HD44780 до AVR мікроконтролерів

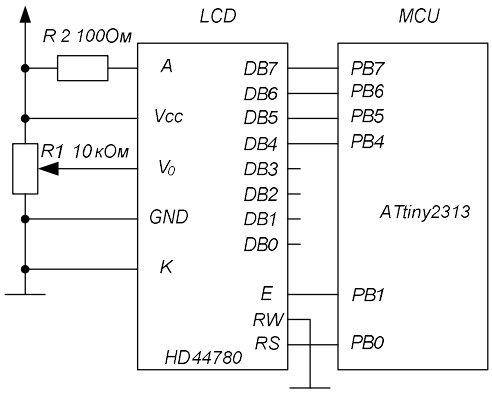


Рисунок 2.6 — Підключення LCD індикатора до мікроконтролера ATtiny2313

Після подачі живлення лінії портів введення / виведення мікроконтролера конфігуруються так: PD0 і PD4 — виходи управління генератора відповідно великого (Iв = 7,69 мА) і малого (Iм = 0,513 мА) струму зарядки вимірюваного конденсатора; PD2 і PD3 — входи запитів переривання; PD5 — вихід управління транзистором VT3, що розряджає вимірюваний конденсатор; PD6 — вхід сигналу від кнопки SB3; PB0 — вихід сигналу управління живленням; PB1 — інвертується вхід вбудованого в мікроконтролер аналогового компаратора, PB2 — PB7 — виходи сигналів управління LCD HG1(рис. 2.7). Неінвертуючий вхід аналогового компаратора програмно підключений до вбудованого в мікроконтролер джерела зразкового напруги 1,0 ... 1,2 В.

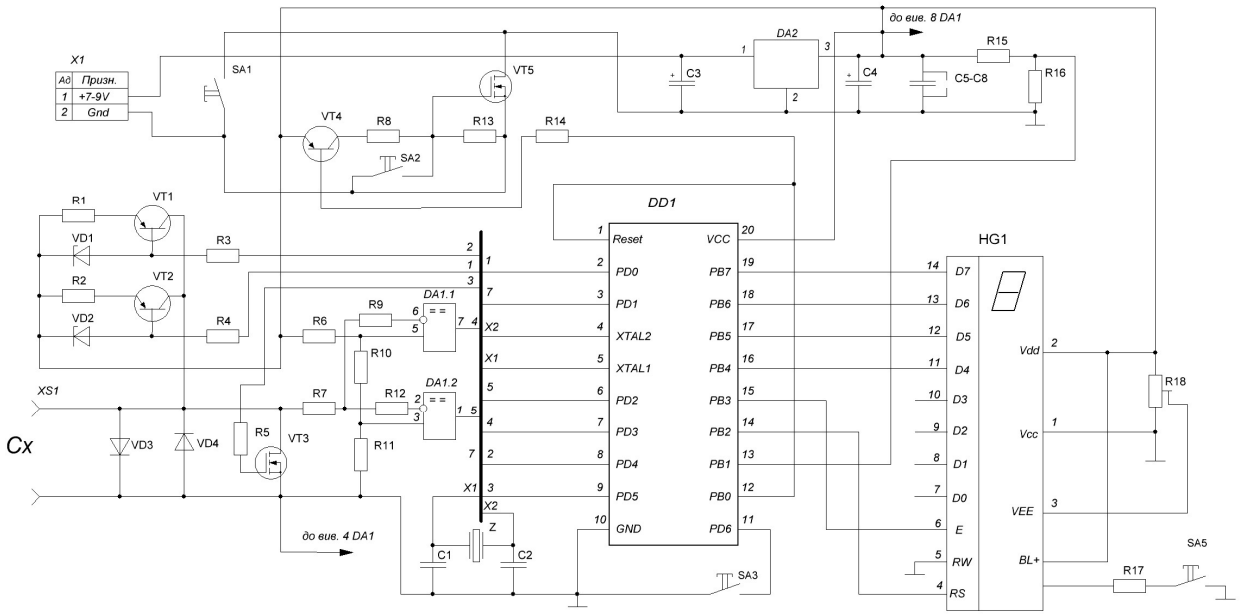


Рисунок 2.7 — Схема пристрою для діагностики якості конденсаторів

Вузол живлення приладу складається з акумуляторної батареї, гнізда X1 для підключення зовнішнього джерела живлення, транзисторів VT4 і VT5, інтегрального стабілізатора DA2, кнопок SA1 (включення приладу) і SA2 (його виключення), а також пов'язаних з цими елементами резисторів і конденсаторів. Напруга з виходу резистивного дільника R16R17, що подається на вхід PB1 мікроконтролера, призначена для програмного контролю напруги живлення.

Діоди VD3, VD4 служать для захисту приладу від пошкодження у разі підключення до нього для вимірювання зарядженого конденсатора. Компаратори DA1.1 і DA1.2 порівнюють напругу на вимірюваному конденсаторі з заданими дільником з резисторів R6 , R9 , R10 пороговими значеннями U1 U2. Резистором R18 регулює контрастність зображення на індикаторі LCD, а резистор R17 обмежує струм у колі його підсвічування. Вимикачем SA2 підсвічування включають і вимикають. Кнопка SA3 переводить прилад в режим налагодження.

Для зниження похибки малі значення ємності (0,1 ... 150 мкФ) вимірюють при малому струмі зарядки (Ім). Генератор даного струму зібраний на елементах VT2, VD2, R2, R4. Він включений при низькому логічному рівні напруги на виході PD0 мікроконтролера. Вимірювання ESR при такому струмі виявляється недостатньо точним за рахунок впливу витоку струму через діод VD3 і вхідні ланки компараторів DA1.1 , DA1.2. З цієї причини ESR конденсаторів будь-якої ємності вимірюється при збільшеному струмі зарядки (Ів), генератор якого складається з елементів VT1, VD1, R1, R3 і включається низьким рівнем на виході PD4 мікроконтролера. Включають прилад натисканням на кнопку SA1, при цьому з виходу стабілізатора DA2 напруга живлення надходить на мікроконтролер DD1. Він починає працювати і після закінчення попередніх операцій встановлює на виводі 12 низький логічний рівень. Транзистор VT4 відкривається, що призводить і до відкривання польового транзистора VT5. Зашунтував кнопку SB1, він утримує прилад включеним і після її відпускання. Для виключення приладу натискають на кнопку SA2.

Для мікропроцесорного пристрою діагностики якості конденсаторів

вибирається варіант з високочастотним кварцовим резонатором для формування тактової частоти мікроконтролера.

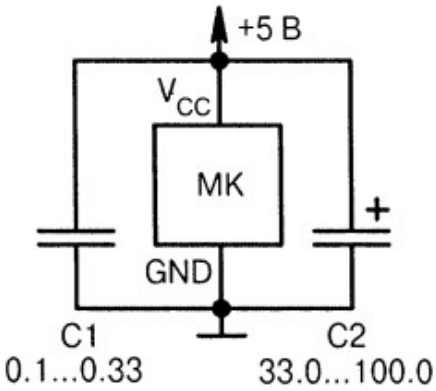


Рисунок 2.8 — Фільтрація завад струму споживання мікроконтролера

Струм споживання мікроконтролера містить низькочастотні та високочастотні складові. Це призводять до появи завад на контактах живлення. Для зменшення завад використовують зв'язку з керамічного та електролітичного конденсаторів (рис. 2.8)

Керамічний конденсатор С1 зменшує високочастотні завади (рис. 2.8) і він повинний бути встановлений як-можна ближче до виводів живлення МК. Номінал ємності 0,1 мкФ орієнтовний. Для зменшення імпульсних завад електролітичний конденсатор С2 має бути танталовим. Ємність —1000 мкФ вибирають для 1А струму навантаження. Як правило, цифрова частина мікроконтролера споживає струм 10...30 мА, то ємність конденсатора С2 вибирається 10...30 мкФ.. Принципова схема пристрою діагностики якості конденсаторів наведена у додатку В.

3 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТА КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ ЗАСОБУ ДІАГНОСТИКИ ЯКОСТІ ЕЛЕКТРОЛІТИЧНИХ КОНДЕНСАТОРІВ

У даному розділі розглядаються питання програмного забезпечення для засобу діагностики якості електролітичних конденсаторів мікроконтролері AVR. Вибирається мова програмування та наводиться алгоритми роботи програмного забезпечення мікроконтролера. Проведено комп'ютерне моделювання роботи схеми пристрою з розробленим програмним забезпеченням.

3.1 Алгоритм роботи засобу діагностики якості електролітичних конденсаторів

Розробка програмного забезпечення мікроконтролера починається зі створення алгоритму роботи. Він допомагає глибше зрозуміти та окреслити основні етапи функціонування МК у складі пристрою. Алгоритм роботи мікропроцесорного пристрою для вимірювання ємності та ESR наведений на рисунку 3.1. При старті мікроконтролера виконується ініціалізація мікроконтролера: настройка стекової пам'яті, порти вводу/виводу, внутрішній компаратор, таймер лічильник 1, система переривань. Далі завантажуються константи з EEPROM для ініціалізації контролера дисплея та встановлюються значення яскравості та підсвітки дисплея. Якщо живлення мікроконтролера менше 5В то на індикаторі виводиться повідомлення «Bat Low» і вимірювання призупиняються. Якщо живлення в нормі, то аналізується чи не натиснута кнопка SA3 (PD6=0). Якщо натиснута, то переходимо в режим калі бровки. Для цього до виводів пристрою підключається опір 1Ом. Проводиться вимірювання ESR і різниця між виміряним значенням та підключеним зразковим опором записується до EEPROM. В режимі вимірювання формуємо імпульс початку вимірювання. Для цього попередньо розряджаємо ємність, що підключена до вимірювання імпульсом, що формується на PD3.

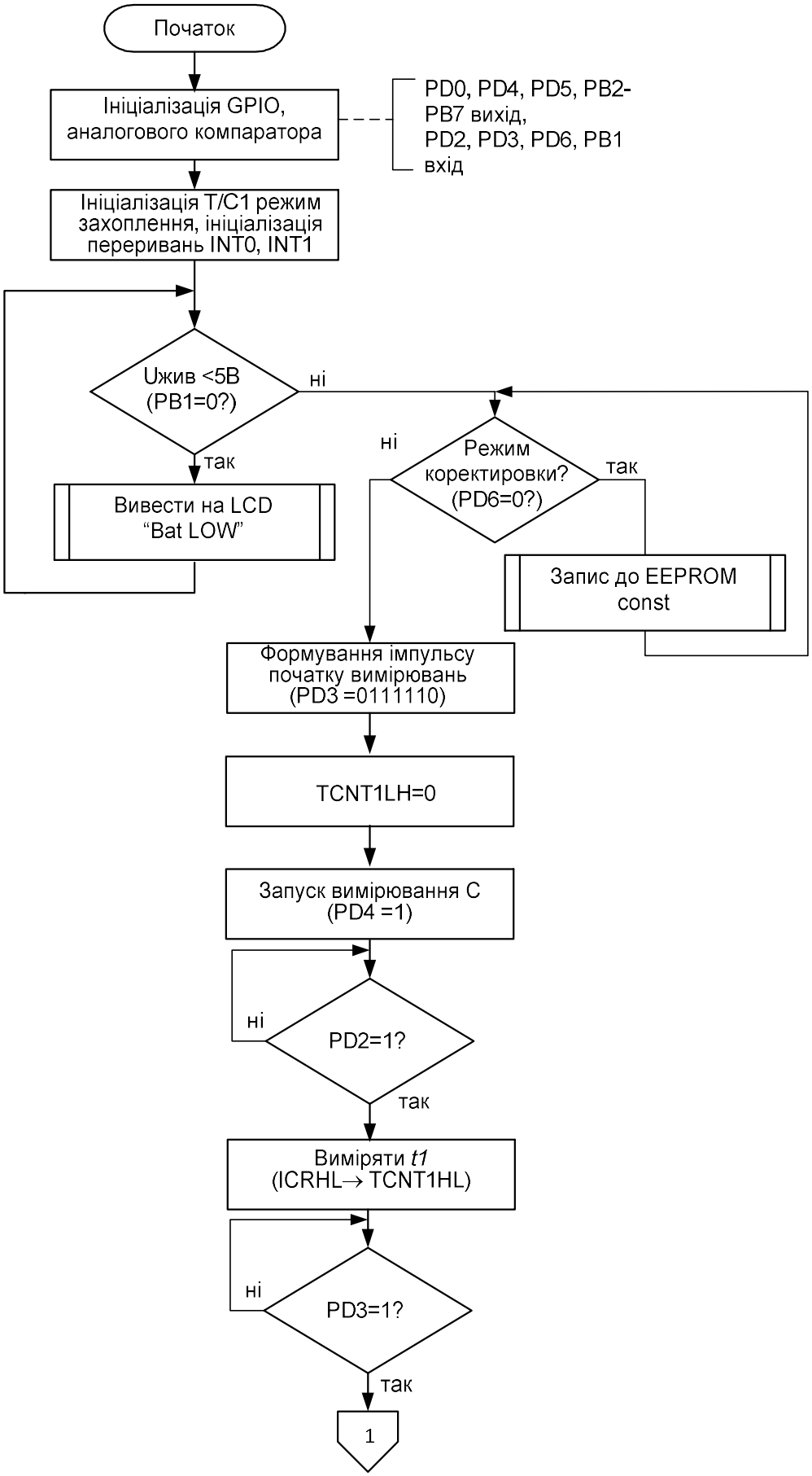


Рисунок 3.1 — Алгоритм роботи вимірювача ємності та ESR

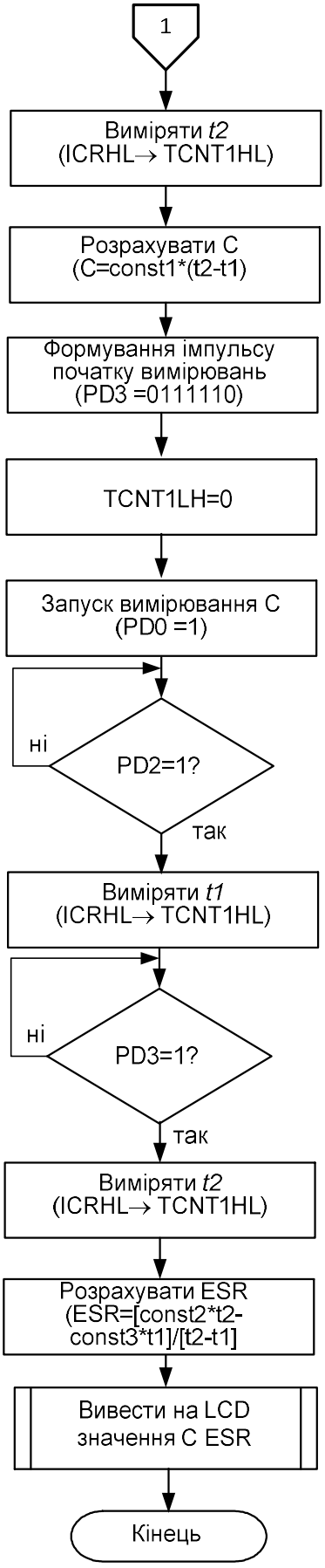


Рисунок 3.1 — Алгоритм роботи вимірювача ємності та ESR (продовження)

Обнуляються значення регістрів TCNT1HL, ICRHL таймера/лічильника Для вимірювання величини ємності С, формується струм Ім (PD4=1), яким вона заряджається. Коли напруга на ємності досягне значення U1 повідомить переривання INT0. За ним час t1 у вигляді коду, що сформований в ICRHL переписується до TCNT1HL, а з нього до регістрів загального призначення. Коли напруга на конденсаторі досягне значення U2, спрацює переривання INT1. За ним час t2 у вигляді коду, що сформований в ICRHL переписується до TCNT1HL, а з нього до регістрів загального призначення. Далі розраховується величина ємності за формулою (26). Для вимірювання ESR формується струм Іб (PD0=1). Процедура вимірювання аналогічна вимірюванні ємності. Величина ESR після визначення t1 та t2 розраховується за формулою (27).

3.2 Організація пам’яті та розподіл адресного простору

Програмна модель AVR-мікроконтролерів зображена на рис. 3.2, на якій відображено доступні ресурси AVR, що доступні програмистом. Арифметико-логічному пристрою доступні 32 регістри загального призначення (R31-R0). Непряма адресація організована через регістрові пари X, Y, Z (R26-R31). 32 регістри загального призначення ($00-$1F), 64 регістри GPIO ($20-$5F) та SRAM 128 x 8 ($60-$DF) утворюють пам'ять даних. Система команд передбачає безпосередню, непряму, регістрову, відносну адресацію.

SRAM мікроконтролера має 128 байт в адресному простіорі від $60 до $DF. Флеш-память займає обсяг 1К х 16 розташована за адресою$000-$3FF. EEPROM має окремий адресний простір, до якого доступ є тільки через спеціальні регістри EEAR ($1E), EEDR ($1D), EECR ($1C). Адреса EEPROM має бути занаесена до регістру EEAR. EEDR - це регістр, який використовується для читання або запису даних. EECR - це регістр, що налаштовує режим роботи EEPROM (стирання, запис, читання).

FlashROM має розрядність 16 біт та призначена для збереження кодів програми. FlashROM дозволяє зберігати коди команд або константи у форматі «байт», «слово».

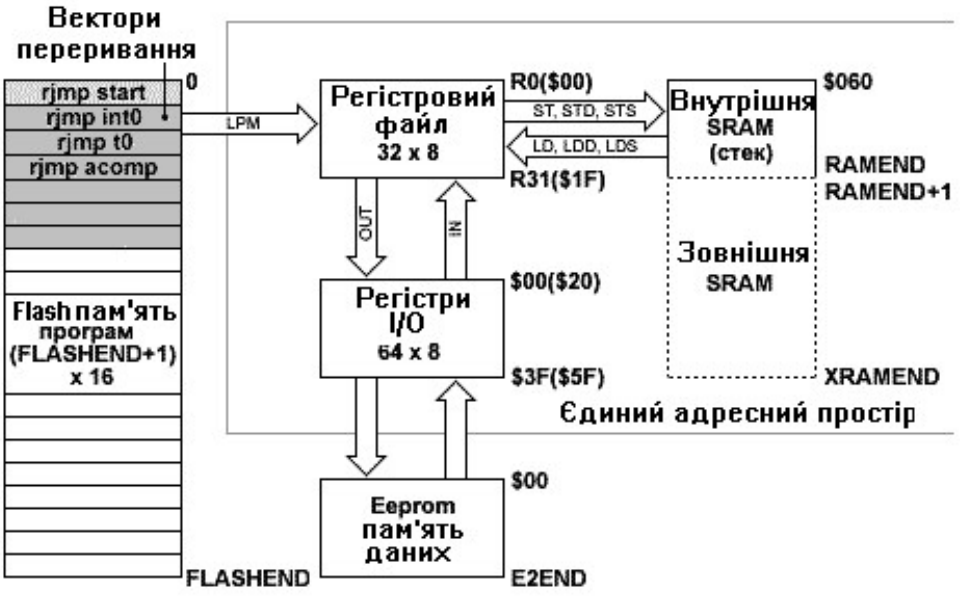


Рисунок 3.2 — Програмна модель AVR-мікроконтролерів

Розподіл регістрів загального призначення наведений нижче

R0 Число сотень долей
R1 Число десятих долей
R2 Число одиниць
R3 Число десятків
R4 Число сотень тисяч
R5 Число тисяч
R6 Число десятків тисяч
R7 1-й байт змінної *a* (молодший байт)
R8 2-й байт змінної *a*
R9 3-й байт змінної *a*
R10 1-й байт змінної *b* (молодший байт)
R11 2-й байт змінної *b*
R12 3-й байт змінної *b*
R13 1-й байт змінної *c* (молодший байт)
R14 2-й байт змінної *c*
R15 3-й байт змінної *c*
R16 4-й байт змінної *c*
R17 Головний робочий регістр
R18 Другий робочий регістр
R19 Регістр для звернення до LCD
R20 1-й байт змінної z (молодший байт)
R21 2-й байт змінної z
R22 3-й байт змінної z
R23 4-й байт змінної z
R24 Регістр поправки коефіцієнта ділення таймера
R25 Регістр величини струму заряду

R28 Регістр лічильник автовиключення
R29 Регістр межі вимірювання в Фарадах

У FLASH ROM будуть розміщені такі константи:

const1b = 8548 — константа переведення результату в ємність для великого струму (32768);

const1m = 9124 — константа переведення результату в ємність для малого струму (524288);

const2 = 2051 — константа 100*U1/Ib;

const3 = 3967 — константа 100*U2/Ib;

ESRmax=2001 — максимальне значення ESR (20,00 Ом);

porog1 = 32700 — поріг перемикання на молодший діапазон (по U2);

porog2 = 15001 — поріг перемикання на малий струм (150 мкФ);

porog3 = 2001 — поріг перемикання межи на малому струмі (20 мкФ);

d20m = 65535 — значення тривалості паузи (~20мс при 20МГц);

d3m = 9999 — значення тривалості паузи (~3мс при 20МГц);

d40u = 235 — значення тривалості паузи (~40мкс при 20МГц);

d1u = 6 — значення тривалості паузи (~1мкс при 20МГц).

EEPROM буде використовуватись так:

Таблиця 3.1 — Використання комірок EEPROM

Адреса комірки EEPROM	Дані
0×00	1-й байт поправки для n1
0×01	2-й байт поправки для n1
0×02	1-й байт поправки для n2
0×03	2-й байт поправки для n2
0×04	1-й байт поправки для (n2-n1)
0×05	2-й байт поправки для (n2-n1)
0×06	ознака розрахунку поправки

3.3 Програмне забезпечення засобу діагностики якості електролітичних конденсаторів

Переваги програмованої системи мікроконтролера tinyAVR неможливо реалізувати без створення ефективного програмного коду.

Мова С — це мова програмування високого рівня, тому написаний на ньому код повинен бути перетворений в машинну мову, яку ваш контролер розуміє й може виконувати. Таке перетворення виконує компілятор. Контролери tinyAVR «розуміють» тільки двійковий формат та повинні отримувати послідовність байтів. Байти, що підлягають передачі в мікроконтролер, зберігаються у вигляді файлу, де вони записані в шістнадцятковій системі числення. Тому повинен бути присутнім такий інструмент, який перетворює код з мови С у шістнадцятковий файл.

Існує безліч різних компіляторів з мови С для мікроконтролерів AVR, але найбільш поширеним є AVR-GCC. WinAVR має хороше інтегроване середовище розробки (для операційної системи Windows).

Асемблер — це набір правил, за якими послідовність команд процесора, записаних в мнемонічному вигляді, може об'єднуватися в програму. Програма спочатку виходить в текстовому форматі. Цей формат повинен являти собою «чистий текст» в однобайтового кодуванні.Цю програму потім компілюють за допомогою власне асемблера (assembler-складальник) —так називається програма, яка переводить текст з мнемонічними позначеннями в послідовність команд і даних, записаних вже в двійковій формі, та придатну для завантаження в пам'ять контролера.

В реальності вибір визначається компактністю коду та ліцензійною чистотою програмного забезпечення. Якщо виріб простий та програміст використовує безкоштовний фірмовий асемблер. Лістинг програми вимірювання наведений у додатку Б.

3.4 Комп’ютерне моделювання роботи пристрою діагностики якості електролітичних конденсаторів

Автономне налагодження пристрою на мікроконтролері полягає у налагодженні як самої апаратури, так і програмного забезпечення.

Налагодження апаратури припускає тестування певних елементів мікропроцесорної системи (МПС) (контролерів, пам’яті, виконавчих пристроїв, елементів індикації, елементів живлення, тактових генераторів). Для цього проектують тестові вхідні дії та аналізують реакції на них. Наступною дією є перевірка їх взаємодії. Для цього аналізують сигнали на системній магістралі. Відлагодження програм мікропроцесорних пристроїв виконується також на персональних комп’ютерах, які використовувались для розробки програмного забезпечення. Відлагодження може бути розпочате на комп’ютері за відсутності апаратної частини мікропроцесорного пристрою. Для цього використовують інтерпретатори та емулятори, що імітують функції апаратних засобів. Перевірка коректності роботи програмного забезпечення виконується у процесі тестування: покроковий режим та трасування програми. Засоби відлагодження програми дозволяють керувати виконанням програм, збирати інформацію під час виконання програми, забезпечувати обмін інтерактивний режим між розробником і комп’ютером на рівні програмного забезпечення, моделювати роботу не встановлених апаратних засобів.

Як правило мікропроцесорна система працює в реальному часі, тобто її функціональність залежить від етапів виконання окремих програм та швидкості роботи апаратури. Тому система вважається працездатною, якщо програми коректно функціонують на реальній апаратурі системи в реальних умовах. Для розробки та налагодження мікропроцесорної системи потрібні вимірювальні прилади для вимірювання напруги, відтворення форми сигналу, формування сигналів певної форми; подавати послідовність сигналів синхронно на декілька входів у відповідності з заданим алгоритмом роботи; фіксувати значення сигналів багатьох ліній протягом одного часового інтервалу, який визначається

подіями, що задаються комбінацією або послідовністю сигналів на лініях; обробляти і подавати зібрану інформацію або у вигляді часової діаграми, або у вигляді таблиці логічних станів, або мовою високого рівня. Для комп'ютерного моделювання широко використовуються осцилографи, вольтметри, вимірювачі частоти, генератори кодів, що дозволяють проводити це на схемотехнічному рівні. Професіонали можуть використовувати логічні аналізатори, налагоджувальні та діагностичні комплекси.

Сучасна лабораторія повинна мати новітнє вимірювальне обладнання та фахівців, здатних підтримувати його в робочому стані. Якщо освітньому закладу утримувати таку лабораторію складно, то розв'язання подібних задач індивідуальним користувачем неможливо.

Завдяки персональному комп'ютеру можна створити віртуальну лабораторію, яка імітує діяльність дослідника в реальній лабораторії Процес моделювання має бути максимально наближений до реальності. Тоді розробник здійснює природну послідовність таких дій, як складання схеми, підключення та установлення режимів роботи вимірювальних приладів, отримання результатів роботи в звичній формі. Таку можливість надає програма Proteus VSM [5, 15].

Дана програма може моделювати роботу аналогових та цифрових схем з дискретними елементами, мікроконтролерів серії AVR, x51, PIC1X, 68000, STM32. Програма має велику бібліотеку напівпровідникових пристроїв, пасивних компонентів, ламп, індикаторів, кнопок, клавіатур, динаміків; генераторів сигналів, двигунів.

Можливість перевіряти роботу МК у реальному часі та у взаємодії з моделями реальних джерел сигналу, еквівалентів навантажень відрізняє Proteus VSM від простих симуляторів.

Пакет Proteus складається з двох програм: ISIS — моделювання електронних схем та ARES — програма створення друкованих плат. У ISIS передбачене покрокове налагодження мікропроцесорної системи, можливість анімації роботи елементів схеми.Розглянемо налагодження вимірювача ESR в

програмі Proteus VSM 8.6. Метою дослідження є перевірка працездатності розробленої схеми та перевірка програмного забезпечення. Схема вимірювача ESR ISIS пакету Proteus VSM приведена на рисунку 3.3.

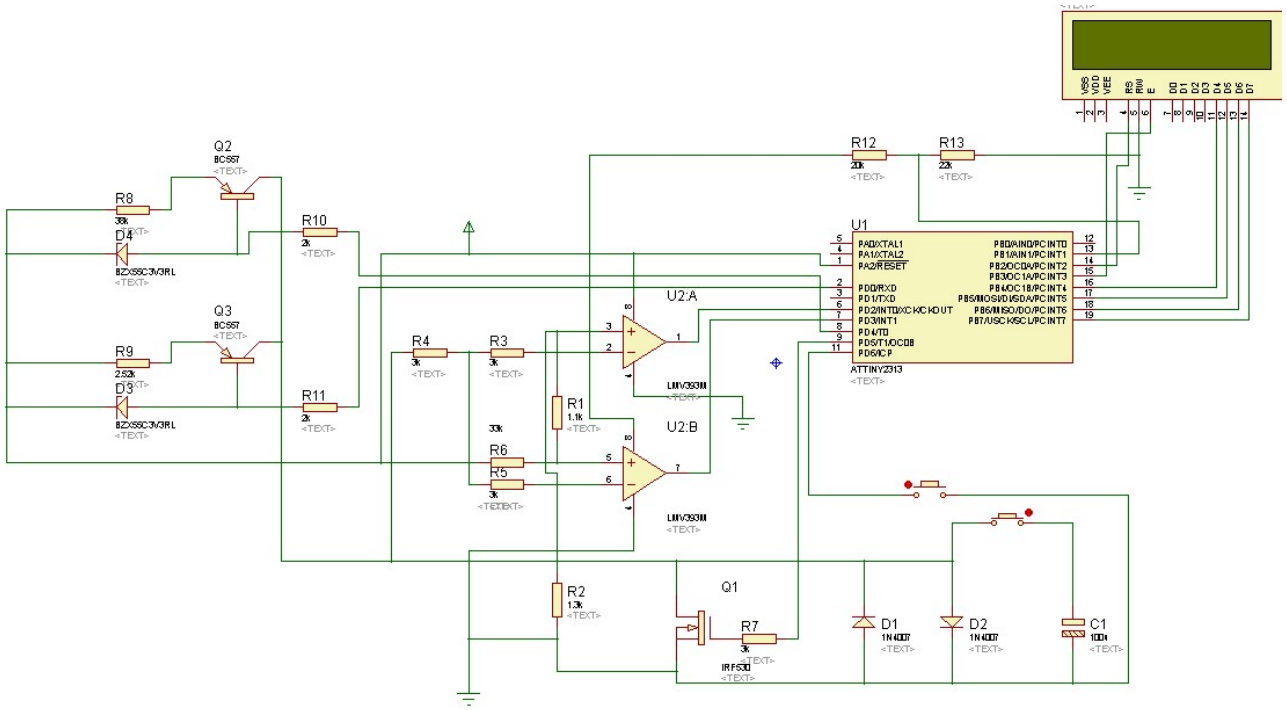


Рисунок 3.3 — Схема вимірювача ESR у середовищі Proteus VSM

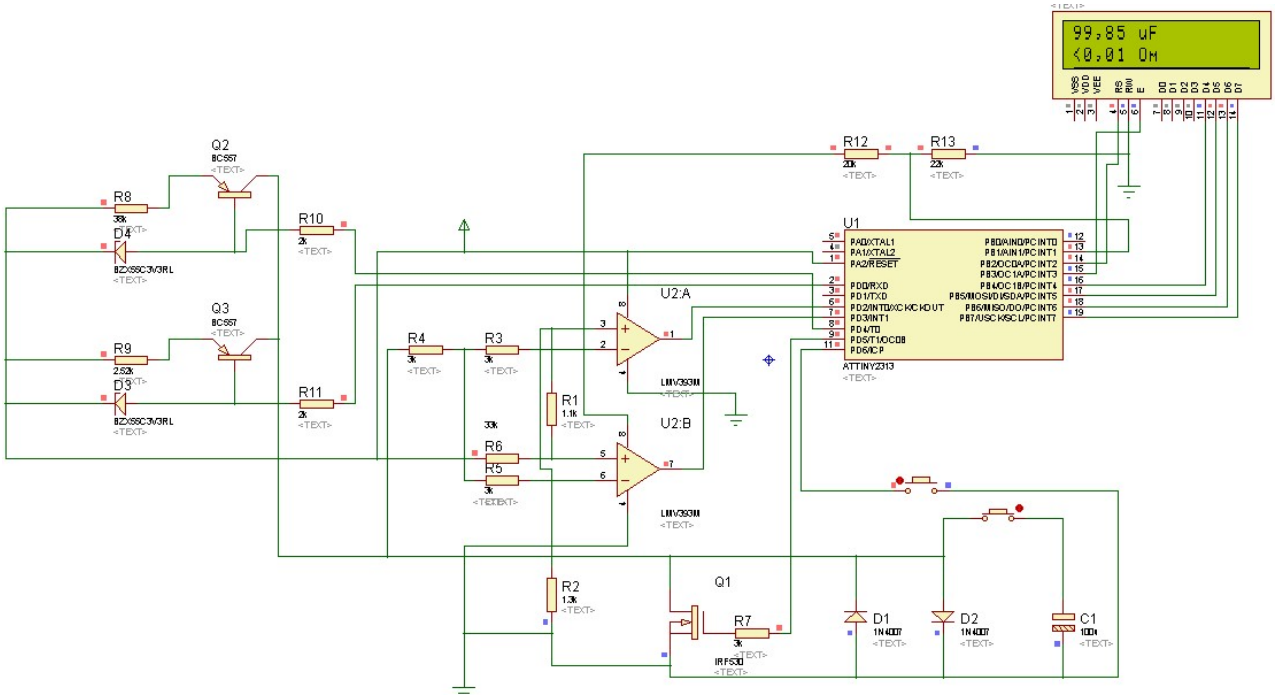


Рисунок 3.4 — Моделювання роботи вимірювача ESR у середовищі Proteus VSM

Після складання схеми вибирається меню *Source → Add/Remove Source Code Files* й підключається програма *C_ESR.ASM*. Вибирається режим відладки програми *Debug→Start/Restart Debugging.* Виводиться відладочна інформація CPU Stack (стек), CPU Registers (регістри загального призначення), CPU Data Memory (пам'ять даних — ОЗП), CPU Source Code (лістинг програми), CPU Program Memory (машинні коди програми). Якщо програма не має помилок, то утворюється файл *C_ESR_Meter_1_1.HEX,* який дозволяє перевірити роботу схеми в реальному режимі роботи.

Для того, щоб продивитись роботу ESR вимірювача необхідно натиснути кнопку ▶. На дисплеї відображається значення величини ємності конденсатора (підключена ємність 100 мкФ) (рисунок 3.4) та величина його ESR.

Провівши моделювання роботи вимірювача ESR можна вважати, що схема розроблена вірно та програмне забезпечення відповідає розробленій схемі. Остаточна перевірка роботи вимірювача на мікроконтролері ATtiny2313 повинна проводитись на дослідному зразку.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою роботи є можливість розробки пристрою діагностики якості електролітичних конденсаторів, який дозволяє прискорити процес діагностики працездатності комп'ютерної техніки та зменшить час виявлення несправних елементів під час ремонту імпульсних блоків живлення персональних комп'ютерів, моніторів, засобів оргтехніки, телекомунікаційного та серверного обладнання та проведення технологічного аудиту й оцінювання комерційного потенціалу розробки (результатів МКР), створеної у результаті науково-технічної діяльності.

Для проведення технологічного аудиту залучимо трьох незалежних експертів. Цими експертами будуть: к.т.н., доцент Роптанов Володимир Ілліч – доцент кафедри обчислювальної техніки Вінницького національного технічного університету; к.т.н., доцент Колесник Ірина Сергіївна – доцент кафедри обчислювальної техніки ВНТУ; Бурдейний Олексій Миколайович – керівник сервісного відділу ТОВ «ЕНРОН». Здійснюємо оцінювання комерційного потенціалу розробки за 12-ма критеріями за 5-ти бальною шкалою.

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 4.1.

Таблиця 4.1 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище та ініціали експерта		
	Роптанов В. І.	Колесник І. С.	Бурдейний О.М.
	Бали, виставлені експертами:		
1	4	4	4
2	3	2	2
3	3	3	3
4	2	3	2
5	2	2	2
6	2	2	2
7	2	0	1
8	4	4	3
9	2	4	3
10	4	3	4
11	4	4	4

Продовження таблиці 4.1

Критерії	Прізвище та ініціали експерта		
	Роптанов В. І.	Колесник І. С.	Бурдейний О.М.
	Бали, виставлені експертами:		
12	4	4	4
Сума	36	35	34
Середньоариф-метична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = 35$		

Отже, з отриманих даних таблиці 4.1 видно, що нова розробка має рівень комерційного потенціалу вище середнього.

Сьогодні на ринку існує декілька пристроїв [1, 9] для діагностики працездатності якості електролітичних конденсаторів, які відрізняються більшою вартістю збільшеним часом проведення тестування або менш інформативні.

4.2 Прогнозування витрат на виконання науково-дослідної роботи та конструкторсько–технологічної роботи

Для розробки нового програмного продукту необхідні такі витрати.

Основна заробітна плата для розробників визначається за формулою (4.1):

$$З = \frac{М}{Тр} \cdot t \ \ [грн], \qquad (4.1)$$

де М – місячний посадовий оклад конкретного розробника;

$Т_Р$ – число робочих днів, $Т_р = 22$;

t – число днів роботи розробника.

Розрахунки заробітних плат для керівника і програміста наведені в таблиці 4.2.

Таблиця 4.2 – Розрахунок основної заробітної плати розробників

Найменування посади	Місячний посадовий оклад,грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
1 Керівник	16000	727,27	14	10181,82
2 Інженер	12000	545,45	22	12000
Всього				22181,82

Розрахуємо додаткову заробітну плату:

$$З_{дод}= 0,1\cdot 22181,82=2218,18 \text{ (грн.)}$$

Нарахування на заробітну плату операторів НЗП розраховується як 37,5…40% від суми їхньої основної та додаткової заробітної плати:

$$Н_{зп}= (З_о + З_р)\cdot \frac{\beta}{100}, \quad (4.2)$$

$$Н_{зп}= (22181,82 +2218,18)\cdot \frac{37,5}{100} = 9150 \text{ (грн.)}.$$

Розрахунок амортизаційних витрат для програмного забезпечення виконується за такою формулою:

$$А = \frac{Ц\cdot Н_а\cdot Т}{100\cdot 12} [\text{грн.}], \quad (4.3)$$

де Ц – балансова вартість обладнання, грн;

$Н_а$ – річна норма амортизаційних відрахувань % (для програмного забезпечення 25%);

Т – Термін використання (Т=3 міс.).

Таблиця 4.3 – Розрахунок амортизаційних відрахувань

Найменування програмного забезпечення	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань,грн
Персональний комп’ютер	16000	25	3	1000
Всього:				1000

Розрахуємо витрати на комплектуючі. Витрати на комплектуючі розрахуємо за формулою:

$$K = \sum_1^n Н_i \cdot Ц_i \cdot К_i, \quad (4.4)$$

де n – кількість комплектуючих;

Ні – кількість комплектуючих і-го виду;

Ці – покупна ціна комплектуючих і-го виду, грн;

Кі – коефіцієнт транспортних витрат (приймемо Кі = 1,1).

Таблиця 4.4 - Витрати на комплектуючі, що були використані для розробки програмного забезпечення

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	шт.	180	1	180
Пачка паперу	уп.	130	1	130
Ручка	шт.	10	1	10
Всього з урахуванням транспортних витрат				352

Витрати на силову електроенергію розраховуються за формулою:

$$В_е=В\cdot П\cdot Ф\cdot К_п ; \quad (4.5)$$

де В – вартість 1кВт-години електроенергії (В=2,44 грн/кВт);

П – установлена потужність комп'ютера (П=0,6кВт);

Ф – фактична кількість годин роботи комп'ютера (Ф=200 год.);

$К_п$ – коефіцієнт використання потужності ($К_п< 1$, $К_п = 0,7$).

$$В_е= 2,44 \cdot 0,6\cdot 200\cdot 0,7 = 205 \text{ (грн.)}$$

Розрахуємо інші витрати $В_{ін}$.

Інші витрати $І_в$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто:

$$В_{ін} = (1..3) \cdot (З_о + З_р). \quad (4.6)$$

Отже, розрахуємо інші витрати:

$$В_{ін} = 1 * (22181,82 + 2218,18) = 24400 \text{ (грн).}$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи:

$$В = З_о + З_д + Н_{зп} + А + К + В_е + І_в$$

$$В = 22181,82 +2218,18+9150+1000+352+205+24400= 59507 \text{ (грн.)}$$

Розрахуємо загальну вартість наукової роботи $B_{заг}$ за формулою:

$$В_{заг} = \frac{В}{\alpha} \quad (4.7)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{59507}{1} = 59507$$

Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta} \quad (4.8)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Отже, розрахуємо загальні витрати:

$$ЗВ = \frac{59507}{0,9} = 66118,89 \text{ (грн.)}$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У даному підрозділі кількісно спрогнозуємо, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи.

Зростання чистого продукту для даного методу можна оцінити у теперішній вартості грошей. Зростання чистого прибутку забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності .

Оцінка зростання чистого прибутку підприємства від впровадження результатів наукової розробки. У цьому випадку збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою

$$\Delta П_i = \sum_{i=1}^{n} (\Delta П_я \cdot N + П_я \cdot \Delta N)_i \quad (4.9)$$

де $\Delta П_{Я}$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

Δ N – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_{Я}$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

В результаті впровадження результатів наукової розробки витрати матеріалів на новий метод зменшяться на 50 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 50 грн), а кількість користувачів, які будуть користуватись збільшиться: протягом першого року – на 300 користувачів, протягом другого року – на 200 користувачів, протягом третього року – 150 користувачів. Реалізація методу до впровадження результатів наукової розробки складала 1000 користувачів, а прибуток, що отримував розробник до впровадження результатів наукової розробки – 100 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Отже, збільшення чистого продукту $\Delta П_1$ протягом першого року складатиме

$$\Delta П_1 = 30 * 1000 + (60 + 30) * 300 = 95000 \text{ (грн.)}.$$

Протягом другого року

$$\Delta П_2 = 50 * 1000 + (100 + 50) * (300 + 200) = 125000 \text{ (грн.)}.$$

Протягом третього року

$$\Delta П_3 = 50 * 1000 + (100 + 50) * (300 + 200 + 150) = 147500 \text{ (грн.)}.$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності. Теперішню вартість інвестицій PV, що вкладаються в наукову розробку приймемо рівну загальним витратам PV = ЗВ = 66118,89 грн.

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули

$$E_{абс} = (ПП - PV), \tag{4.10}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн;

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде має вигляд, рисунок 4.1.

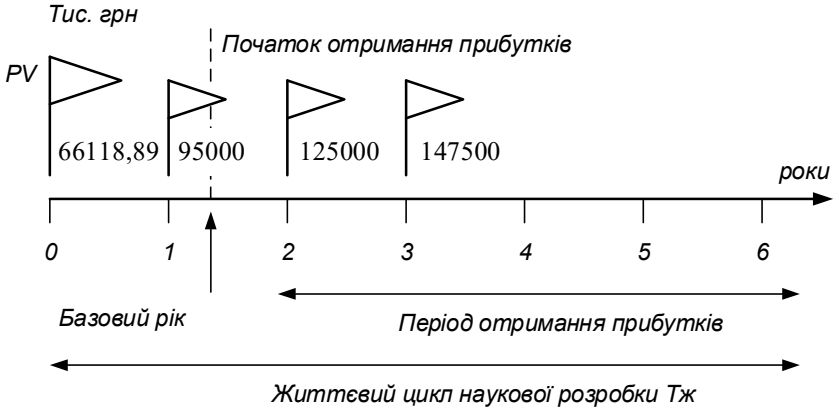


Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_1^T \frac{\Delta П_i}{(1+\tau)^t}\ , \tag{4.11}$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом=яких виявляються результати виконаної та впровадженої НДДКР, грн;

Т –період часу, протягом якою виявляються результати впровадженої

НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках).

$$\text{ПП} = \frac{66118{,}89}{(1+0{,}1)^0} + \frac{95000}{(1+0{,}1)^1} + \frac{125000}{(1+0{,}1)^2} + \frac{147500}{(1+0{,}1)^3} = 366\,607{,}24(\text{грн.}),$$

$$\text{Е}_{абс} = \left(366\,607{,}24 - 66118{,}89\right) = 300488{,}35 \text{ (грн.)}.$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього користуються формулою

$$E_B = \sqrt[Тж]{1 + \frac{Е\text{абс}}{PV}} - 1, \qquad (4.12)$$

де $T_ж$ – життєвий цикл наукової розробки, роки

$$E_B = \sqrt[3]{1 + \frac{300488{,}35}{66118{,}89}} - 1 = 0{,}77(77\%)$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою

$$= \tau d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2020 році в Україні d = 0,2;

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = 0,1

$$\tau_{\text{мін}} = 0{,}2 + 0{,}1 = 0{,}3 .$$

Оскільки $\text{Е}_в = 77\ \% > \tau_{\text{мін}} = 0{,}3 = 30\%$, то у інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

Розрахуємо термін окупності вкладених у реалізацію наукового

проекту інвестицій за формулою

$$\mathrm{T_{ОК}} = \frac{1}{\mathrm{E_В}},$$

$$\mathrm{T_{ОК}} = \frac{1}{0{,}77} = 1{,}3(\text{роки})$$

Так як $T_{ОК} \leq$ 3...5-ти років, то фінансування даної наукової розробки буде доцільним.

ВИСНОВКИ

У результаті виконання магістерської дипломної роботи були розглянуті методи та засіб діагностики якості електролітичних конденсаторів, що необхідно в процесі ремонту комп'ютерної техніки, мережного устаткування та засобів оргтехніки.

В рамках роботи було розглянуто основні причини несправностей електролітичних конденсаторів, проаналізовані методи вимірювання ємності конденсаторів та обрано метод оцінки ESR, який є швидким та інформативним. Величина ESR електролітичного конденсатора на частоті 50-100кГц дорівнює значенню повного комплексного опору конденсатора. Якість електролітичного конденсатора характеризується малим часом розряду конденсатора між двома пороговими значеннями напруги.

Мікроконтролер ATtiny2313 програмно формує прямокутну напругу, яка прикладається до конденсатора, що перевіряється. Конденсатор заряджається до напруги 1В і вимірюється таймером мікроконтролера час розряду конденсатора до напруги 0,5В. Порогові значення напруги для управління таймером встановлюються компаратором мікроконтролера. Фіксоване значення струму заряду дозволяє визначити величину ємності конденсатора та його ESR. Математичні обчислення С та ESR виконується програмно мікроконтролером з виведенням інформації на LCD індикатор.

Засіб діагностики якості електролітичних конденсаторів відрізняється застосуванням методу оцінки ESR та сучасної елементної бази, що дозволяє прискорити процес діагностики працездатності електролітичних конденсаторів та достовірність отриманих результатів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Афонский А. А., Дьяконов В. П. Измерительные приборы и массовые электронные измерения. М: СОЛОН-ПРЕСС, 2007. 544 с.
2. Измерение ЭПС (ESR) конденсаторов. URL: https://elwo.ru/publ/spravochniki/izmerenie_ehps_esr_kondensatorov/2-1-0-993
3. История 8 бит. Chip News Украина. Инженерная микроэлектроника. №8 (158), 2016. С. 36-38.
4. Козловський В. О. Методичні вказівки до виконання студентами-магістрантами наукового напрямку економічної частини магістерських кваліфікаційних робіт. Вінниця: ВНТУ, 2012. 12 с.
5. Краткий учебный курс PROTEUS. Русское руководство для начинающих. URL: http://proteus123.narod.ru/
6. Лебедев М. Б. CodeVision AVR. Пособие для начинающих. М.: Додэка XXI, 2008. 594 с.
7. Липавский И. Измеритель ESR с линейной шкалой. Ремонт & Сервис. 2006. №4. URL: https://cutt.ly/sydHF0T
8. Меерсон А. М. Радиоизмерительная техника. Ленинград: Энергия, 1978. 408 с.
9. Обзор современных измерителей импеданса (измерители RLC). URL: https://prist.ru/info/articles/lcr-meters.htm
10. Пиз А. Роберт. Практическая электроника аналоговых устройств. Поиск неисправностей и отработка проектируемых схем. М: ДМК Пресс. 2001. 320 с.
11. Простой метод измерения ESR конденсаторов. URL: https://radioprog.ru/post/757
12. Рюмик С. М. 1000 и одна микронтроллерная схема. Вып. IV. М. : ДМК-Пресс, 2017. 336 с.
13. Томел Д., Уидмер Н. Поиск неисправностей в электронике. М.: НТ Пресс, 2007. 416 с.

14. Трамперт В. Измерение, управление и регулировка с помощью AVR микроконтроллеров. К.: «МК-Пресс», 2006. 208 с.
15. Цирульник С. М., Лисенко Г. Л. Проектування мікропроцесорних систем. Вінниця: ВНТУ, 2012. 191с.
16. Цирульник С. М., Ткачук В. М., Гаврасієнко А. О. Прилад для вимірювання параметрів LC. Збірник тез доповідей І МНТК «Вимірювання, контроль та діагностика в технічних системах (ВКДТС-2011)». Вінниця, 2011. С. 95.
17. Шмаков С. Б. Энциклопедия радиолюбителя. Современная элементная база. СПб.: Наука и Техника, 2012. 384 с.
18. Шонфелдер Г., Шнайдер К. Измерительные устройства на базе микропроцессора ATmega. СПб.: БХВ-Петербург, 2012. 288 с.
19. Шпак Ю. А. Программирование на языке С для AVR и PIC микроконтроллеров. К: ДМК-Пресс, 2006. 400 с.
20. Amaral A.M.R., Cardoso A.J.M.: An experimental technique for estimating the ESR and reactance intrinsic values of aluminium electrolytic capacitors. Proc. Instrumentation and Measurement Technology Conf., IMTC 2006, April 2006, pp. 1820—1825.
21. Chen Y.-M., Chou M.-W., Wu H.-C.: Electrolytic capacitor failure prediction of LC filter for switching-mode power converters. Proc. 40th Annual Meeting IEEE Industry Applications Society, October 2005, vol. 2, pp. 1464—1469.
22. ESR метр. URL: https://elwo.ru/publ/esr_metr/1-1-0-635
23. ESR. Способы измерения. URL: https://tel-spb.ru/esr.html
24. Sankaran V.A., Rees F.L., Avant C.S.: Electrolytic capacitor life testing and prediction. Proc. 32nd Annual Meeting IEEE Industry Applications Society, October 1997, vol. 2, pp. 1058—1065

Додаток А

(обов’язковий)

ВНТУ

ПОГОДЖЕНО

“__” _____ 2020 р.

ЗАТВЕРДЖУЮ

Зав. кафедри ОТ,

д.т.н., професор

Т. Б. Мартинюк

“__” _____ 2020 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

МЕТОДИ ТА ЗАСІБ ДІАГНОСТИКИ ЯКОСТІ ЕЛЕКТРОЛІТИЧНИХ КОНДЕНСАТОРІВ КОМП’ЮТЕРНОЇ ТЕХНІКИ

08-23.МКР.001.00.000 ПЗ

Керівник роботи

к.т.н., доц. кафедри ОТ ВНТУ

_____ Роптанов В. І.

Виконавець: ст. гр. 1КІ-18м

_____ Кірше А. О.

Вінниця-2020

1 Підстава для виконання роботи

Робота проводиться на підставі наказу ректора по Вінницькому національному технічному університету № від 2020 р. та індивідуального завдання на магістерську кваліфікаційну роботу.

2 Мета і призначення

Метою даної магістерської кваліфікаційної роботи є дослідження методів та засобу діагностики якості електролітичних конденсаторів комп'ютерної техніки, що дозволяють прискорити процес діагностики працездатності електролітичних конденсаторів та зменшить час виявлення несправних елементів під час ремонту комп'ютерної техніки.

Задачами дослідження є:

— аналіз несправностей електролітичних конденсаторів;
— аналіз методів тестування електролітичних конденсаторів;
— обґрунтування методу та засобу діагностики якості електролітичних конденсаторів комп'ютерної техніки;
— удосконалення існуючих пристроїв діагностики працездатності конденсаторів для практичної реалізації пристрою діагностики якості електролітичних конденсаторів комп'ютерної техніки.

Об'єктом дослідження є процес діагностики несправностей елементів комп'ютерної техніки.

Предметом дослідження є методи та засіб діагностики якості електролітичних конденсаторів.

Основними завданнями роботи є:

— огляд літературних джерел;
— теоретичне дослідження методів тестування електролітичних конденсаторів та вибір методу для діагностики їх якості;
— теоретичне дослідження приладів для вимірювання ємності конденсаторів та вимірювачів параметрів RLC;

- розробка мікропроцесорного пристрою діагностики якості електролітичних конденсаторів;
- розробка програмного забезпечення для мікропроцесорного пристрою діагностики якості електролітичних конденсаторів;
- аналіз економічної ефективності проведеної розробки.

3 Вихідні дані для проведення МКР

Список рекомендованих джерел розробки:

1. Афонский А. А., Дьяконов В. П. Измерительные приборы и массовые электронные измерения. М: СОЛОН-ПРЕСС, 2007. 544 с.
2. Пиз А. Роберт. Практическая электроника аналоговых устройств. Поиск неисправностей и отработка проектируемых схем. М: ДМК Пресс. 2001. 320 с.
3. Цирульник С. М., Лисенко Г. Л. Проектування мікропроцесорних систем. Вінниця: ВНТУ, 2012. 191 с.
4. Шонфелдер Г., Шнайдер К. Измерительные устройства на базе микропроцессора ATmega. СПб.: БХВ-Петербург, 2012. 288 с.
5. ESR. Способы измерения. URL: https://tel-spb.ru/esr.html
6. ESR метр. URL: https://elwo.ru/publ/esr_metr/1-1-0-635
7. Рюмик С. М. 1000 и одна микронтроллерная схема. Вып. IV. М. : ДМК-Пресс, 2017. 336 с.

4 Виконавець

Вінницький національний технічний університет, кафедра обчислювальної техніки, студент групи 1КІ-18м Кірше Андрій.

5 Вимоги до виконання МКР

Комплексно розглянути та актуалізовані методи вимірювання параметрів електролітичних конденсаторів та удосконалити пристрій діагностики якості електролітичних конденсаторів, який відрізняється застосуванням сучасної елементної бази, що дозволить прискорити процес діагностики та ремонту комп’ютерної техніки, мережевого обладнання, засобів оргтехніки та

достовірність отриманих результатів.

6 Етапи МКР і терміни їх виконання

№	Назва та зміст етапу	Термін виконання		Очікувані результати	Звітна документація
		початок	закінчення		
1.	Розробка технічного завдання (ТЗ)	1.02.2020	1.03.2020	Розроблене ТЗ	Додаток А
2.	Огляд літературних джерел. Огляд і аналіз існуючих методів та засобів вимірювання параметрів конденсаторів. Вибір методу для діагностики якості електролітичних конденсаторів	1.03.2020	1.04.2020	Проведений аналіз	Вступ. Розділ 1
3.	Огляд та аналіз елементної бази пристрою. Розробка схеми пристрою	1.04.2020	4.05.2020	Схема пристрою	Розділ 2
4	Розробка програмного забезпечення Комп’ютерне моделювання.	4.05.2020	30.05.2020	Програмне забезпечення для аналізу інформації про рівень рідини	Розділ 3
5.	Аналіз економічної ефективності розробки	30.05.2020	6.06.2020	Економічна частина	Розділ 4
7.	Оформлення пояснювальної записки (ПЗ) та графічної частини	6.06.2020	15.06.2020	Оформлена документація	ПЗ та графічна частина
8.	Попередній захист та рецензування МКР	16.06.2020	20.06.2020	Позитивні відзиви	Відзив. Рецензія
9.	Захист МКР ДЕК	21.06.2020	21.06.2020	Позитивний захист	Протокол ДЕК

7 Очікувані результати та порядок реалізації МКР

У результаті виконання роботи будуть розроблені:

— рекомендації, що дозволяють використовувати метод вимірювання ESR для оперативної діагностики якості електролітичних конденсаторів;
— застосування методу вимірювання ESR для реалізації прототипу мікропроцесорного діагностики якості електролітичних конденсаторів.

8 Матеріали, які подають після закінчення роботи та під час етапів

За результатами виконання МКР до ДЕК подаються пояснювальна записка, графічна частина МКР, відзив і рецензія.

9 Порядок приймання МКР та її етапів

Поетапно результати виконання МКР розглядаються керівником роботи та обговорюються на засіданні кафедри.

Захист магістерської кваліфікаційної роботи відбувається на відкритому засіданні ДЕК.

10 Вимоги щодо технічного захисту інформації з обмеженим доступом

У зв’язку з тим, що інформація не є конфіденційною, заходи з її технічного захисту не передбачаються.

Додаток Б

Лістинг програмного забезпечення для мікроконтролера

```
.include   "tn2313def.inc"  ;Присоединение файла описания

.list                              ;Включение листинга

.def  sotye             = R0  ;Число сотых долей (1)
.def  desyatye    = R1  ;Число десятых долей (2)
.def  edinicy     = R2  ;Число единиц (3)
.def  desyatki    = R3  ;Число десятков (4)
.def  sotni             = R4  ;Число сотен тысяч (5)
.def  tyshi             = R5  ;Число тысяч (6)
.def  des_tysh    = R6  ;Число десятков тысяч (7)
.def  a1 = R7                      ;1-й байт переменной a (младший байт)
.def  a2 = R8                      ;2-й байт переменной a
.def  a3 = R9                      ;3-й байт переменной a
.def  b1 = R10               ;1-й байт переменной b (младший байт)
.def  b2 = R11               ;2-й байт переменной b
.def  b3 = R12               ;3-й байт переменной b
.def  c1 = R13               ;1-й байт переменной c (младший байт)
.def  c2 = R14               ;2-й байт переменной c
.def  c3 = R15               ;3-й байт переменной c
;-----------------------------------------------------
.def  c4 = R16               ;4-й байт переменной c
.def  temp  = R17       ;Определение главного рабочего регистра
.def  temp2       = R18       ;Определение второго рабочего регистра
.def  lcd   = R19             ;Определение регистра для обращения к LCD
.def  z1    = R20             ;1-й байт переменной z (младший байт)
.def  z2    = R21             ;2-й байт переменной z
.def  z3    = R22             ;3-й байт переменной z
.def  z4    = R23             ;4-й байт переменной z
.def  k_del = R24       ;Регистр поправки коэффициента деления таймера
.def  tok_zar     = R25       ;Регистр величины тока заряда
.def  autoff      = R28       ;Регистр счётчика автовыключения (он же YL)
.def  autoff2     = R29       ;Регистр предела измерения в Фарадах (он же YH)

;-----------------------------------------------------
.equ  const1b = 8548   ;Константа перевода результата в ёмкость для большего
тока (32768)
.equ  const1m = 9124   ;Константа перевода результата в ёмкость для малого
тока (524288)
.equ  const2 = 2051    ;Константа 100*U1/Ib
.equ  const3 = 3967    ;Константа 100*U2/Ib
.equ  const4 = 129     ;Константа для задания скорости USART (9600 бит/с)

.equ  ESRmax=2001      ;Порог максимального значения ESR (20,00 Ом)
.equ  porog1 = 32700   ;Порог переключения на младший предел (по U2)
.equ  porog2 = 15001   ;Порог переключения на малый ток (150 мкФ)
.equ  porog3 = 2001    ;Порог переключения предела на малом токе (20 мкФ)
.equ  d20m = 65535     ;Значение длительности паузы (~20мс при 20МГц)
(тактов-9)/6
.equ  d3m = 9999       ;Значение длительности паузы (~3мс при 20МГц)
(тактов-9)/6
.equ  d40u = 235       ;Значение длительности паузы (~40мкс при 20МГц)
(тактов-4)/3
```

```
.equ  d1u   =     6                ;Значение длительности паузы (~1мкс при
20МГц) (тактов-4)/3
.equ  RS = 2                       ;
.equ  E = 3                  ;

;------------------------Резервирование ячеек памяти EEPROM

.eseg
.org  0x08             ;установка текущего адреса сегмента

n1_1b:
.byte 1                      ;Ячейка EEPROM для 1-го байта поправки для n1

n1_2b:
.byte 2                      ;Ячейка EEPROM для 2-го байта поправки для n1

n2_1b:
.byte 3                      ;Ячейка EEPROM для 1-го байта поправки для n2

n2_2b:
.byte 4                      ;Ячейка EEPROM для 2-го байта поправки для n2

n2_n1_1b:
.byte 5                      ;Ячейка EEPROM для 1-го байта поправки для (n2-
n1)

n2_n1_2b:
.byte 6                      ;Ячейка EEPROM для 2-го байта поправки для (n2-
n1)

flag:
.byte 7                      ;Ячейка EEPROM для записи флага расчёта
поправки

;------------------------Начало программного кода

.cseg
.org  0                      ;установка текущего адреса на ноль

;------------------------Переопределение векторов прерываний

start:      rjmp  init  ;Переход на начало программы
            rjmp  com_U1      ;Внешнее прерывание 0
            rjmp  com_U2      ;Внешнее прерывание 1
            reti              ;Таймер/счётчик 1, захват
            reti              ;Таймер/счётчик 1, совпадение, канал A
            rjmp  ovtim1      ;Таймер/счётчик 1, прерывание по переполнению
            reti              ;Таймер/счётчик 0, прерывание по переполнению
            reti              ;Прерывание USART приём завершён
            reti              ;Прерывание USART регистр данных пуст
            reti              ;Прерывание USART передача завершена
            reti              ;Прерывание по компаратору
            reti              ;Прерывание по изменению на любом контакте
            reti              ;Таймер/счётчик 1, совпадение, канал B
            reti              ;Таймер/счётчик 0, совпадение, канал B
            reti              ;Таймер/счётчик 0, совпадение, канал A
            reti              ;USI готовность к старту
            reti              ;USI переполнение
            reti              ;EEPROM готовность
            reti              ;Переполнение охранного таймера
```

```
;-----------------------Модуль инициализации
init:
;-----------------------Инициализация стека

ldi   temp, RAMEND           ;Выбор адреса внешнего стека
out   SPL, temp              ;Запись его в регистр стека

;-----------------------Инициализация USART

cli                                    ;Глобально запрещаем прерывания
ldi         temp, high(const4)     ;Старший полубайт константы в temp
out         UBRRH, temp            ;Запись в регистр скорости обмена USART
старшего полубайта
ldi         temp, low(const4);Младший полубайт константы в temp
out         UBRRL, temp            ;Запись в регистр скорости обмена USART
младшего полубайта
ldi         temp, 0b00000000 ;Установка одинарной скорости обмена
out         UCSRA, temp            ;
ldi         temp, 0b00001000 ;Разрешение передачи, запрет прерываний
out         UCSRB, temp            ;
ldi         temp, 0b00000110 ;Настройка режима работы USART (8-бит, асинхр.,
без контроля чётности)
out         UCSRC, temp            ;

;-----------------------Инициализация портов ВВ

ldi         temp,0b00000100  ;PA2 на вывод, PA0, PA1 на ввод
out         DDRA,temp        ;
ldi         temp,0b00000100  ;Устанавливаем PA2, отключаем резисторы на PA0,
PA1
out         PORTA,temp       ;

ldi         temp,0b11111101  ;PB0, PB2...PB7 на вывод, PB1 на ввод
out         DDRB,temp        ;
ldi         temp,0b00000000  ;Обнуляем PB1...PB7, устанавливаем PB1 и
включаем резистор на PB0
out         PORTB,temp       ;

ldi         temp,0b00110011  ;PD0, PD1, PD4...PD5 на вывод, PD2, PD3, PD6 на
ввод
out         DDRD,temp        ;
ldi         temp,0b01111111  ;Устанавливаем PD0, PD1, PD4...PD5 и включаем
резисторы PD2, PD3, PD6
out         PORTD,temp       ;

;-----------------------Инициализация таймера T1

ldi         temp,0b00000000  ;Останов таймера, нормальный режим
out         TCCR1B,temp      ;

;-----------------------Инициализация компаратора

ldi         temp,0b01110000  ;Настройка компаратора
out         ACSR,temp        ;

;-----------------------Определение масок прерываний

ldi         temp,0b00000000  ;Выбор режима вызова внешних прерываний PD2
(U1) и PD3 (U2)
out         MCUCR,temp       ;
```

```
ldi         temp,0b00000000  ;Запрет прерываний по таймеру 1
out         TIMSK,temp

ldi         temp,0b00000000  ;Запрет внешних прерываний
out         GIMSK,temp       ;

;----------------------Инициализация LCD (KS0066)

lcd_init:
rcall del20m                 ;Вызов подпрограммы задержки (~20мс)
rcall del20m                 ;Вызов подпрограммы задержки (~20мс)
rcall del20m                 ;Вызов подпрограммы задержки (~20мс)

                                   ;Команда 0010
sbi         PORTB,E                ;Строб вверх
ldi         temp,0b00101000  ;Команда (строб вверх + команда)
out         PORTB,temp       ;Запись команды в порт
rcall del1u            ;Вызов подпрограммы задержки (~1мкс)
cbi         PORTB,E                ;Строб вниз
rcall del40u                 ;Вызов подпрограммы задержки (~40мкс)

ldi         lcd,0b00101000   ;Команды 0010 и 1000
rcall lcd_com                ;Вызов подпрограммы отправки команды на LCD

ldi         lcd,0b00001000   ;Команды 0000 и 1000
rcall lcd_com                ;Вызов подпрограммы отправки команды на LCD

ldi         lcd,0b00000001   ;Команды 0000 и 0001
rcall lcd_com                ;Вызов подпрограммы отправки команды на LCD

rcall del3m            ;Вызов подпрограммы задержки (~3мс)

ldi         lcd,0b00000110   ;Команды 0000 и 0110
rcall lcd_com                ;Вызов подпрограммы отправки команды на LCD

;----------------------Начало основной программы

cli                          ;Глобально запрещаем прерывания
clr         k_del            ;Очистка k_del
clr         tok_zar                ;Очистка tok_zar
clr         autoff                 ;Очистка autoff
clr         autoff2                ;Очистка autoff2
ldi         lcd,0b00001100   ;Включение дисплея и выбор курсора
rcall lcd_com                ;

;----------------------Отображение приветствия

ldi         ZL,low(Prvt*2)   ;Запись в ZL младшего байта адреса ячейки из
таблицы Prvt (альтернативная адресация)
ldi         ZH,high(Prvt*2)  ;Запись в ZH старшего байта адреса ячейки из
таблицы Prvt (альтернативная адресация)
rcall lcd_all                ;Вызов подпрограммы отображения двух строк на
LCD по данным из таблиц

main:
clr         c3                     ;Очистка c3 (очистка 3-го байта счётчика
таймера)
sbi         PORTD,0                ;Выключение большого тока заряда
конденсатора
sbi         PORTD,4                ;Выключение малого тока заряда
конденсатора
```

```
sbi         PORTD,5                 ;Включение разряда конденсатора
clr         temp                ;Обнуляем счётный регистр таймера 1
out         TCNT1H,temp         ;
out         TCNT1L,temp         ;

;-----------------------Автовыключение и контроль за напряжением питания

autoff_check:
cpi         autoff,255          ;Сравнение autoff с 255 (~4'35")
breq  autoff_check1     ;Если autoff=255, то на autoff_check1, иначе далее

cpi         k_del,4                 ;Сравнение k_del с 4
brlo  autoff_check00    ;Если k_del<4, то на autoff_check00, иначе далее

cpi         autoff2,10          ;Сравнение autoff2 с 10 (~2'17")
brlo  bat_check1        ;Если autoff2<10, то на bat_check1, иначе далее
rjmp  autoff_check1     ;На autoff_check1

autoff_check00:
cpi         k_del,3                 ;Сравнение k_del с 3
brlo  autoff_check0     ;Если k_del<3, то на autoff_check0, иначе далее

cpi         autoff2,31          ;Сравнение autoff2 с 31 (~2'22")
brlo  bat_check1        ;Если autoff2<31, то на bat_check1, иначе далее
rjmp  autoff_check1     ;На autoff_check1

autoff_check0:
cpi         autoff2,100         ;Сравнение autoff2 с 100 (~2'17")
brlo  bat_check1        ;Если autoff2<100, то на bat_check1, иначе далее

autoff_check1:
;-----------------------Выключение прибора

ldi         temp,0b01110000  ;Разрешение перехода в режим Power-Down
out         MCUCR,temp          ;
sbi         PORTB,0                 ;Выключение питания
sleep                       ;Переход в режим Power-Down

bat_check1:
clr         z1                      ;Очистка z1
clr         z2                      ;Очистка z2
ldi         temp,55                 ;Запись 55 в temp

bat_check2:
rcall del20m                ;Вызов подпрограммы задержки (~20мс)
dec         temp                ;Уменьшение temp на 1
sbis  ACSR,ACO          ;Если на выходе компаратора 1, то пропуск след.
команды, иначе далее
rjmp  bat_check3        ;На bat_check3
inc         z1                      ;Увеличение z1 на 1
rjmp  bat_check4        ;На bat_check4

bat_check3:
inc         z2                      ;Увеличение z2 на 1

bat_check4:
cpi         temp,0                  ;Сравнение temp с 0
brne  bat_check2        ;Если temp<>0, то на bat_check2, иначе далее

cp          z1,z2               ;Сравнение z1 с z2
brlo  pusk_0                ;Если z1<z2, то на pusk_0, иначе далее
```

```
ldi         ZL,low(Bat*2)     ;Запись в ZL младшего байта адреса ячейки из
таблицы Bat (альтернативная адресация)
ldi         ZH,high(Bat*2)    ;Запись в ZH старшего байта адреса ячейки из
таблицы Bat (альтернативная адресация)
rcall lcd_all                 ;Вызов подпрограммы отображения двух строк на
LCD по данным из таблиц
inc         autoff2                 ;Увеличение autoff2 на 1

;-----------------------Передача сообщения о разряде батареи

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xD6         ;Запись 0xD6 в temp
out         UDR,temp          ;Передача из temp в USART

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

;-----------------------Формирование паузы для отображения сообщения о
замене батареи (~0,3с при 20МГц)

rcall del300m                 ;Вызов подпрограммы задержки (~300мс)

cpi         k_del,2                 ;Сравнение k_del с 2
brsh  pusk_0                  ;Есл k_del>=2, то на pusk_0, иначе далее

ldi         ZL,low(Gdit*2)    ;Запись в ZL младшего байта адреса ячейки из
таблицы Gdit (альтернативная адресация)
ldi         ZH,high(Gdit*2)   ;Запись в ZH старшего байта адреса ячейки из
таблицы Gdit (альтернативная адресация)
ldi         temp,1                  ;Отображение 1-й строки
rcall lcd_mess          ;Вызов подпрограммы отображения сообщений

ldi         ZL,low(Gdit*2)    ;Запись в ZL младшего байта адреса ячейки из
таблицы Gdit (альтернативная адресация)
ldi         ZH,high(Gdit*2)   ;Запись в ZH старшего байта адреса ячейки из
таблицы Gdit (альтернативная адресация)
ldi         temp,2                  ;Отображение 2-й строки
rcall lcd_mess          ;Вызов подпрограммы отображения сообщений

pusk_0:
;-----------------------Передача сообщения о пределе измерения

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xC0         ;Запись 0xC0 в temp
or          temp,k_del        ;Логическое сложение k_del с temp, результат в
temp
out         UDR,temp          ;Передача из temp в USART

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

pusk:
ldi         temp,0b01000000   ;Разрешение внешнего прерывания только по PD2
(U1)
out         GIMSK,temp        ;
```

```
ldi         temp,0b10000000  ;Разрешение прерывания только по переполнению
таймера 1
out         TIMSK,temp       ;

cbi         PORTD,5                ;Выключение разряда конденсатора

rcall del1u             ;Вызов подпрограммы задержки (~1мкс), подождём
немного закрытия MOSFET

sei                                ;Глобально разрешаем прерывания

cpi         tok_zar,1        ;Сравнение tok_zar с 1
breq  m_tok             ;Если tok_zar=1, то на m_tok, иначе далее

;-----------------------Измерение на большом токе заряда

b_tok:
ldi         temp,0b00000001  ;Пуск таймера (clk/1)
add         temp,k_del       ;с коррекцией коэффициента деления таймера
cbi         PORTD,0                ;Включение заряда конденсатора
out         TCCR1B,temp      ;
rjmp  cikl              ;На cikl

;-----------------------Измерение на малом токе заряда

m_tok:
ldi         temp,0b00000001  ;Пуск таймера (clk/1)
add         temp,k_del       ;с коррекцией коэффициента деления таймера
cbi         PORTD,4                ;Включение заряда конденсатора
out         TCCR1B,temp      ;

cikl:
rjmp  cikl              ;Зацикливаем

;-----------------------Уменьшение предела измерения (увеличение частоты
таймера)

clk_up:
cpi         k_del,0                ;Сравнение k_del с 0
breq  clk_no_ch         ;Если k_del=0, то на clk_no_ch, иначе далее
dec         k_del                  ;Уменьшение поправки на 1 (уменьшение
коэффициента деления)
rjmp  clk_pred          ;На clk_pred

;-----------------------Увеличение предела измерения (уменьшение частоты
таймера)

clk_down:
clr         tok_zar                ;Очистка k_del
cpi         k_del,4                ;Сравнение k_del с 4.
brlo  clk_down_0        ;Если k_del<4, то на clk_down_0, иначе далее
rjmp  step3             ;Иначе на step3 (отображение сообщения о КЗ)

clk_down_0:
inc         k_del                  ;Увеличение поправки на 1 (увеличение
коэффициента деления)

clk_pred:
ldi         ZL,low(Pred*2)   ;Запись в ZL младшего байта адреса ячейки из
таблицы Pred (альтернативная адресация)
```

```
ldi         ZH,high(Pred*2)  ;Запись в ZH старшего байта адреса ячейки из
таблицы Pred (альтернативная адресация)
rcall lcd_all                ;Вызов подпрограммы отображения двух строк на
LCD по данным из таблиц
rcall pred_num          ;Вызов подпрограммы отображения номера предела
clr         autoff                  ;Очистка autoff

;-----------------------Передача сообщения "ждите..."

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xD4         ;Запись 0xD4 в temp
out         UDR,temp          ;Передача из temp в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

clk_no_ch:
rjmp  main              ;На main

;-----------------------Проверка нижнего порога (porog1)
;Если значение счётчика больше или равно porog1, то на внесение поправок (на
step4),
;Если значение счётчика меньше porog1, то на проверку частоты таймера (далее
на step2)

step1:
ldi         z1,low(porog1)   ;Заносим младший байт porog1 в z1
ldi         z2,high(porog1)  ;Заносим старший байт porog1 в z2
ldi         z3,0             ;Заносим 0 в z3
cp          b1,z1            ;Сравнение b2+b1 с z2+z1 с учётом переноса
cpc         b2,z2            ;
cpc         b3,z3            ;
brsh  step4             ;Если b3+b2+b1 больше или равно z3+z2+z1, то на
step4, иначе далее (на step2)

;-----------------------Проверка частоты таймера
;Если k_del=0 (clk/1), то на внесение поправок (на step4), если k_del<>0,
;то на  уменьшение предела измерения (на clk_up)

step2:
cpi         k_del,0                 ;Сравнение k_del с 0
breq  step4             ;Если k_del=0, то на внесение поправок (на step4)
rjmp  clk_up                  ;Иначе на уменьшение предела измерения (на
clk_up)

;-----------------------Отображение сообщения о КЗ при превышении ожидания

step3:
ldi         ZL,low(KZ_Cmx*2) ;Запись в ZL младшего байта адреса ячейки из
таблицы KZ_Cmx (альтернативная адресация)
ldi         ZH,high(KZ_Cmx*2);Запись в ZH старшего байта адреса ячейки из
таблицы KZ_Cmx (альтернативная адресация)
rcall lcd_all                       ;Вызов подпрограммы отображения двух строк
подряд на LCD по данным из таблиц
ldi         temp,85                     ;Допустимые значения 1, 3, 5, 15,
17, 51, 85, 255 (при 85 около 50 сек)
add         autoff,temp

;-----------------------Передача сообщений "Cx > max" и "ESR < min"
```

```
rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xD0         ;Запись 0xD0 в temp
out         UDR,temp          ;Передача из temp в USART

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

rjmp  main             ;На main

;----------------------Внесение поправок на задержки

step4:
cpi         tok_zar,1         ;Сравнение tok_zar с 1
breq  step6_0                 ;Если tok_zar=1, то на step6_0, иначе далее

;----------------------Чтение из EEPROM

ldi         temp2,flag        ;Записывает в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM

cpi         temp2,100         ;Сравнение temp2 с 100
breq  step5            ;Если temp2=100, то на step5, иначе далее
rjmp  step6            ;На step6

step5:
ldi         temp2,n1_1b       ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z1,temp2          ;Копируем прочитанный байт из temp2 в z1

ldi         temp2,n1_2b       ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z2,temp2          ;Копируем прочитанный байт из temp2 в z2

rcall predel                  ;Вызов подпрограммы predel

sub         a1,z1             ;Вычитание z1 из a1
sbc         a2,z2             ;Вычитание z2 из a2 с учётом переноса
sbc         a3,temp2          ;Вычитание temp2 из a3 с учётом переноса

ldi         temp2,n2_1b       ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z1,temp2          ;Копируем прочитанный байт из temp2 в z1

ldi         temp2,n2_2b       ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z2,temp2          ;Копируем прочитанный байт из temp2 в z2

rcall predel                  ;Вызов подпрограммы predel

sub         b1,z1             ;Вычитание z1 из b1
sbc         b2,z2             ;Вычитание z2 из b2 с учётом переноса
sbc         b3,temp2          ;Вычитание temp2 из b3 с учётом переноса
```

```
ldi         temp2,n2_n1_1b   ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z1,temp2         ;Копируем прочитанный байт из temp2 в z1

ldi         temp2,n2_n1_2b   ;Записываем в temp2 адрес читаемой ячейки
EEPROM
rcall EEPROM_read      ;Вызов подпрограммы чтения ячейки EEPROM
mov         z2,temp2         ;Копируем прочитанный байт из temp2 в z2

rcall predel               ;Вызов подпрограммы predel

sub         c1,z1            ;Вычитание z1 из c1
sbc         c2,z2            ;Вычитание z2 из c2 с учётом переноса
sbc         c3,temp2         ;Вычитание temp2 из c3 с учётом переноса

;-----------------------Расчёт и отображение

step6:

push  c1                     ;Сохранение c1 в стеке (сохранение n2-n1)
push  c2                     ;Сохранение c2 в стеке
push  c3                     ;Сохранение c3 в стеке

push  a1                     ;Сохранение a1 в стеке (сохранение n1)
push  a2                     ;Сохранение a2 в стеке
push  a3                     ;Сохранение a3 в стеке
rjmp  step6_1                ;На step6_1

step6_0:

;-----------------------Умножение (c3+c2+c1)*(XH+XL)=(c4+c3+c2+c1) или (n2-
n1)*const1b

ldi         XL,low(const1m)        ;Запись младшего байта const1m в XL
ldi         XH,high(const1m) ;Запись старшего байта const1m в XH
rjmp  step6_2                      ;На step6_2

step6_1:
ldi         XL,low(const1b)        ;Запись младшего байта const1b в XL
ldi         XH,high(const1b) ;Запись старшего байта const1b в XH

step6_2:
rcall       mul16x16         ;Вызов подпрограммы перемножения 16-ти битных
чисел

;-----------------------Деление на 32...32768 с помощью переносов

cpi         k_del,4                ;Если поправка равна 4 (clk/1024), то
переход на divC_32
breq  divC_32              ;иначе далее

cpi         k_del,3                ;Если поправка равна 3 (clk/256), то
переход на divC_128
breq  divC_128         ;иначе далее

cpi         k_del,2                ;Если поправка равна 2 (clk/64), то
переход на divC_512
breq  divC_512         ;иначе далее
```

```
cpi         k_del,1                 ;Если поправка равна 1 (clk/8), то переход
на divC_4096
breq  divC_4096         ;иначе далее

divC_32768:
rcall del32_8                ;/8

divC_4096:
rcall del32_8                ;/8

divC_512:
rcall del32_4                ;/4

divC_128:
rcall del32_4                ;/4

divC_32:
rcall del32_32          ;/32

cpi         tok_zar,0         ;Сравнение tok_zar с 0
breq  step6_3                 ;Если tok_zar=0, то на step6_3, иначе далее
rcall del32_2                 ;/2
rcall del32_8                 ;/8
clr         k_del             ;Очистка k_del
clr         tok_zar                 ;Очистка tok_zar

;-----------------------Перевод двоичного результата ёмкости в двоично-
десятичный (ёмкость, малый ток)

rcall bin2dec                 ;Вызов подпрограммы перевода двоичного числа в
двоично-десятичное (ёмкость, малый ток))

;-----------------------Отображение первой строки LCD

rcall lcd_1str          ;Вызов подпрограммы отображения первой строки LCD

;-----------------------Передача с 1-го по 7-й байт через USART

rcall usart_1_7         ;Вызов подпрограммы передачи с 1-го по 7-й байт через
USART

rjmp  main              ;На main

step6_3:
ldi         z1,low(porog2)    ;Запись младшего байта porog2 в z1
ldi         z2,high(porog2)   ;Запись старшего байта porog2 в z2
clr         z3                      ;Очистка z3
clr         z4                      ;Очистка z4
cp          c1,z1             ;Сравнение c1 с z1
cpc         c2,z2             ;Сравнение c2 с z2 с учётом переноса
cpc         c3,z3             ;Сравнение c3 с z3 с учётом переноса
cpc         c4,z4             ;Сравнение c4 с z4 с учётом переноса
brsh  step6_3_1         ;Если (c4+c3+c2+c1)>=porog2, то на step6_3_1, иначе
далее
ldi         tok_zar,1         ;Запись 1 в tok_zar

ldi         z1,low(porog3)    ;Запись младшего байта porog3 в z1
ldi         z2,high(porog3)   ;Запись старшего байта porog3 в z2
cp          c1,z1             ;Сравнение c1 с z1
cpc         c2,z2             ;Сравнение c2 с z2 с учётом переноса
cpc         c3,z3             ;Сравнение c3 с z3 с учётом переноса
```

```
cpc         c4,z4             ;Сравнение c4 с z4 с учётом переноса
brlo  step6_3_0         ;Если (c4+c3+c2+c1)<porog3, то на step6_3_0, иначе
далее
inc         k_del             ;Увеличение поправки на 1 (увеличение
коэффициента деления)

step6_3_0:
rjmp  step7             ;На step7

step6_3_1:
ldi         z1,0x80               ;Запись 1-го байта порога переключения на
доли Фарад (10000000)
ldi         z2,0x96               ;Запись 2-го байта порога переключения на
доли Фарад
ldi         z3,0x98               ;Запись 3-го байта порога переключения на
доли Фарад
cp          c1,z1             ;Сравнение c1 с z1
cpc         c2,z2             ;Сравнение c2 с z2 с учётом переноса
cpc         c3,z3             ;Сравнение c3 с z3 с учётом переноса
cpc         c4,z4             ;Сравнение c4 с z4 с учётом переноса
brlo  step6_4               ;Если (c4+c3+c2+c1)<(z4+z3+z2+z1), то на
step6_4, иначе далее

;-----------------------Деление (c4+c3+c2+c1)/10000=(c3+c2+c1)    (b3+b2+b1)
step6_3_2:
ldi         temp,low(10000)  ;Запись младшего байта 10000 в temp
mov         a1,temp                 ;Копирование из temp в a1
ldi         temp,high(10000);Запись старшего байта 10000 в temp
mov         a2,temp                 ;Копирование из temp в a2
clr         a3                      ;Очистка a3

rcall       div32_16          ;Вызов подпрограммы деления 32-х на 16-ти
битное число

;-----------------------Перевод двоичного результата ёмкости в двоично-
десятичный

step6_4:
rcall bin2dec               ;Вызов подпрограммы перевода двоичного числа в
двоично-десятичное (ёмкость)

;-----------------------Отображение первой строки LCD

rcall lcd_1str          ;Вызов подпрограммы отображения первой строки LCD
clr         autoff                  ;Очистка autoff

;-----------------------Передача с 1-го по 7-й байт через USART

step6_5:
rcall usart_1_7         ;Вызов подпрограммы передачи с 1-го по 7-й байт через
USART

step7:
pop         c3                      ;Извлечение из стека в c3 (извлечение n1)
pop         c2                      ;Извлечение из стека в c2
pop         c1                      ;Извлечение из стека в c1

;-----------------------Умножение (c3+c2+c1)*(XH+XL)=(c4+c3+c2+c1) или
(U2/I)*n1

ldi         XL,low(const3)    ;Запись младшего байта const3 в XL
```

```
ldi         XH,high(const3)  ;Запись старшего байта const3 в XH
rcall       mul16x16         ;Вызов подпрограммы перемножения 16-ти битных
чисел

mov         z1,c1            ;Копирование результата умножения из c1 в z1
mov         z2,c2            ;Копирование результата умножения из c2 в z2
mov         z3,c3            ;Копирование результата умножения из c3 в z3
mov         z4,c4            ;Копирование результата умножения из c4 в z4

mov         c1,b1            ;Копирование 1-го байта n2 из b1 в c1
mov         c2,b2            ;Копирование 2-го байта n2 из b2 в c2
mov         c3,b3            ;Копирование 3-го байта n2 из b3 в c3

;-----------------------Умножение (c3+c2+c1)*(XH+XL)=(c4+c3+c2+c1) или
(U1/I)*n2

ldi         XL,low(const2)   ;Запись младшего байта const2 в XL
ldi         XH,high(const2)  ;Запись старшего байта const2 в XH
rcall       mul16x16         ;Вызов подпрограммы перемножения 16-ти битных
чисел

pop         a3                    ;Извлечение из стека в a3 (извлечение n2-
n1)
pop         a2                    ;Извлечение из стека в a2
pop         a1                    ;Извлечение из стека в a1

sub         c1,z1            ;Вычитание z1 из c1 ((U1/I)*n2)-((U2/I)*n1)
sbc         c2,z2            ;Вычитание z2 из c2 с учётом переноса
sbc         c3,z3            ;Вычитание z3 из c3 с учётом переноса
sbc         c4,z4            ;Вычитание z4 из c4 с учётом переноса

brpl  step8                  ;Если результат положительный, то на step8,
иначе далее
ldi         ZL,low(ESRmn*2)        ;Запись в ZL младшего байта адреса ячейки
из таблицы ESRmn (альтернативная адресация)
ldi         ZH,high(ESRmn*2) ;Запись в ZH старшего байта адреса ячейки из
таблицы ESRmn (альтернативная адресация)

;-----------------------Передача сообщения "ESR < 0,01 Om"

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xD3        ;Запись 0xD3 в temp
out         UDR,temp         ;Передача из temp в USART

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи

rjmp  step8_0                ;На step8_0

;-----------------------Деление (c4+c3+c2+c1)/(a3+a2+a1)=(c3+c2+c1)
(b3+b2+b1) или (((U1/I)*n2)-((U2/I)*n1))/(n2-n1)
step8:
rcall       div32_16         ;Вызов подпрограммы деления 32-х на 16-ти
битное число

;-----------------------Сравнение результата вычитания с максимальным
значением

clr         c3                         ;Очистка c3
```

```
ldi         temp,low(ESRmax) ;Запись младшего байта ESRmax в temp
ldi         temp2,high(ESRmax)     ;Запись старшего байта ESRmax в temp2
cp          c1,temp                     ;Сравнение c1 с temp
cpc         c2,temp2                ;Сравнение c2 с temp2 с учётом переноса
brlo  step9                   ;Если (c2+c1)< ESRmax, то на step9, иначе далее
ldi         ZL,low(ESRmx*2)         ;Запись в ZL младшего байта адреса ячейки
из таблицы ESRmx (альтернативная адресация)
ldi         ZH,high(ESRmx*2) ;Запись в ZH старшего байта адреса ячейки из
таблицы ESRmx (альтернативная адресация)

;-----------------------Передача сообщения "ESR >20,0 Om"

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xD2         ;Запись 0xD2 в temp
out         UDR,temp          ;Передача из temp в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

step8_0:
ldi         temp,2                  ;Отображение 2-й строки
rcall lcd_mess          ;Вызов подпрограммы отображения сообщений

rjmp  main              ;На main

;-----------------------Перевод двоичного результата ESR в двоично-
десятичный

step9:
rcall bin2dec                 ;Вызов подпрограммы перевода двоичного числа в
двоично-десятичное (ESR)

;-----------------------Отображение второй строки LCD

step10:
rcall lcd_2str          ;Вызов подпрограммы отображения второй строки LCD

;-----------------------Передача 8-го...11-й байт через USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0x80         ;Запись 0x80 в temp
or          desyatki,temp     ;Логическое сложение desyatki с temp, результат
в desyatki
out         UDR,desyatki      ;Передача из desyatki в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0x90         ;Запись 0x90 в temp
or          edinicy,temp      ;Логическое сложение edinicy с temp, результат
в edinicy
out         UDR,edinicy       ;Передача из edinicy в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xA0         ;Запись 0xA0 в temp
```

```
or          desyatye,temp     ;Логическое сложение desyatye с temp, результат
в desyatye
out         UDR,desyatye      ;Передача из desyatye в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi         temp,0xB0         ;Запись 0xB0 в temp
or          sotye,temp        ;Логическое сложение sotye с temp, результат в
sotye
out         UDR,sotye         ;Передача из sotye в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

rjmp  main              ;На main

;-----------------------Подпрограмма отображение первой строки LCD

lcd_1str:

;-----------------------Отображение ёмкости

ldi         lcd,0x80          ;Выбор знакоместа (начало 1й строки)
rcall lcd_com                 ;Вызов подпрограммы отправки команды на LCD

;-----------------------Отображение "12345" или " 2345"

clr         temp              ;Очистка temp
cp          des_tysh,temp     ;Если значение десятков тысяч не равно нулю, то
переход на lcd_1str_4
brne  lcd_1str_4        ;иначе далее

cp          tyshi,temp        ;Если значение тысяч не равно нулю, то переход
на lcd_1str_3
brne  lcd_1str_3        ;иначе далее

cp          sotni,temp        ;Если значение сотен не равно нулю, то переход
на lcd_1str_2
brne  lcd_1str_2        ;иначе далее

cp          desyatki,temp     ;Если значение десятков не равно нулю, то
переход на lcd_1str_0
brne  lcd_1str_0        ;иначе далее

;-----------------------Отображение " 5,67" или "45,67"

ldi         lcd,0x20          ;Отображение " " (ёмкость)
rcall lcd_dat                 ;
rjmp  lcd_1str_1        ;На lcd_1str_1

lcd_1str_0:
cpi         k_del,4                 ;Сравнение k_del с 4
brne  lcd_1str_0_1      ;Если k_del<>4, то на lcd_1str_0_1

ldi         lcd,0x30          ;Отображение "0" (ёмкость)
rcall lcd_dat                 ;

ldi         lcd,0x2C          ;Отображение "," (ёмкость)
rcall lcd_dat                 ;
```

```
lcd_1str_0_1:
ldi         lcd,0x30          ;Определение кода для числа десятков (ёмкость)
add         lcd,desyatki      ;
rcall lcd_dat                 ;Отображение числа десятков (ёмкость)

lcd_1str_1:
ldi         lcd,0x30          ;Определение кода для числа единиц (ёмкость)
add         lcd,edinicy;
rcall lcd_dat                 ;Отображение числа единиц (ёмкость)

cpi         k_del,4                 ;Сравнение k_del с 4
brne  lcd_1str_1_1     ;Если k_del<>4, то на lcd_1str_1_1

ldi         lcd,0x30          ;Определение кода для числа десятых долей
(ёмкость)
add         lcd,desyatye      ;
rcall lcd_dat                 ;Отображение числа десятых долей (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа сотых долей
(ёмкость)
add         lcd,sotye         ;
rcall lcd_dat                 ;Отображение числа сотых долей (ёмкость)

ldi         lcd,0x20          ;Отображение " " (ёмкость)
rcall lcd_dat                 ;

rjmp  F_lcd             ;На F_lcd

lcd_1str_1_1:
ldi         lcd,0x2C          ;Отображение "," (ёмкость)
rcall lcd_dat                 ;

ldi         lcd,0x30          ;Определение кода для числа десятых долей
(ёмкость)
add         lcd,desyatye      ;
rcall lcd_dat                 ;Отображение числа десятых долей (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа сотых долей
(ёмкость)
add         lcd,sotye         ;
rcall lcd_dat                 ;Отображение числа сотых долей (ёмкость)
rjmp  uF_lcd                  ;На uF_lcd

;-----------------------Отображение "345,6"

lcd_1str_2:
ldi         lcd,0x30          ;Определение кода для числа сотен (ёмкость)
add         lcd,sotni         ;
rcall lcd_dat                 ;Отображение числа сотен (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа десятков (ёмкость)
add         lcd,desyatki            ;
rcall lcd_dat                 ;Отображение числа десятков (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа единиц (ёмкость)
add         lcd,edinicy;
rcall lcd_dat                 ;Отображение числа единиц (ёмкость)

ldi         lcd,0x2C          ;Отображение "," (ёмкость)
rcall lcd_dat                 ;
```

```
ldi         lcd,0x30          ;Определение кода для числа десятых долей
(ёмкость)
add         lcd,desyatye      ;
rcall lcd_dat                 ;Отображение числа десятых долей (ёмкость)
rjmp  uF_lcd                  ;На uF_lcd

;-----------------------Отображение " 2345" или "12345"

lcd_1str_3:
ldi         lcd,0x20          ;Отображение " " (ёмкость)
rcall lcd_dat                 ;
rjmp  lcd_1str_5        ;На lcd_1str_5

lcd_1str_4:
ldi         lcd,0x30          ;Определение кода для числа десятков тысяч
(ёмкость)
add         lcd,des_tysh      ;
rcall lcd_dat                 ;Отображение числа десятков тысяч (ёмкость)

lcd_1str_5:
ldi         lcd,0x30          ;Определение кода для числа тысяч (ёмкость)
add         lcd,tyshi         ;
rcall lcd_dat                 ;Отображение числа тысяч (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа сотен (ёмкость)
add         lcd,sotni         ;
rcall lcd_dat                 ;Отображение числа сотен (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа десятков (ёмкость)
add         lcd,desyatki      ;
rcall lcd_dat                 ;Отображение числа десятков (ёмкость)

ldi         lcd,0x30          ;Определение кода для числа единиц (ёмкость)
add         lcd,edinicy;
rcall lcd_dat                 ;Отображение числа единиц (ёмкость)

uF_lcd:
ldi         lcd,0x20          ;Отображение " " (ёмкость)
rcall lcd_dat                 ;

ldi         lcd,0x75          ;Отображение "u" (ёмкость)
rcall lcd_dat                 ;

F_lcd:
ldi         lcd,0x46          ;Отображение "F" (ёмкость)
rcall lcd_dat                 ;
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отображение второй строки LCD

lcd_2str:

;-----------------------Отображение ESR

ldi         lcd,0xC0          ;Выбор знакоместа (начало 2й строки)
rcall lcd_com                 ;

clr         temp              ;Очистка temp
cp          desyatki,temp     ;Если значение десятков равно нулю, то переход
на lcd_2str_0
breq  lcd_2str_0        ;иначе далее
```

```
ldi         lcd,0x30          ;Определение кода для числа десятков (ESR)
add         lcd,desyatki                ;
rcall lcd_dat                 ;Отображение числа десятков (ESR)
rjmp  lcd_2str_1        ;Переход на lcd_2str_1

lcd_2str_0:
ldi         lcd,0x20          ;Отображение " " (ESR)
rcall lcd_dat                 ;

lcd_2str_1:
ldi         lcd,0x30          ;Определение кода для числа единиц (ESR)
add         lcd,edinicy             ;
rcall lcd_dat                 ;Отображение числа единиц (ESR)

ldi         lcd,0x2C          ;Отображение "," (ESR)
rcall lcd_dat                 ;

ldi         lcd,0x30          ;Определение кода для числа десятых долей (ESR)
add         lcd,desyatye                ;
rcall lcd_dat                 ;Отображение числа десятых долей (ESR)

ldi         lcd,0x30          ;Определение кода для числа сотых долей (ESR)
add         lcd,sotye               ;
rcall lcd_dat                 ;Отображение числа сотых долей (ESR)

ldi         lcd,0x20          ;Отображение " " (ESR)
rcall lcd_dat                 ;

ldi         lcd,0x4F          ;Отображение "O" (ESR)
rcall lcd_dat                 ;

ldi         lcd,0x6D          ;Отображение "m" (ESR)
rcall lcd_dat                 ;
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отображение сообщений на LCD

lcd_mess:
ldi         z4,8              ;Запись 8 в z4
cpi         temp,1                  ;Сравнение temp с 1
brne  lcd_mess_2str     ;Если str не равен 1, то на lcd_mess_2str, иначе
далее

lcd_mess_1str:
ldi         lcd,0x80          ;Выбор знакоместа (начало 1й строки)
rcall lcd_com                 ;
rjmp  lcd_mess_0        ;На lcd_mess_0

lcd_mess_2str:
ldi         lcd,0xC0          ;Выбор знакоместа (начало 2й строки)
rcall lcd_com                 ;

lcd_mess_0:
lpm         lcd,Z+                  ;Запись lcd байта состояния строк из
ячейки с адресом из Z
rcall lcd_dat                 ;
dec         z4                      ;Уменьшение z4 на 1
cpi         z4,0              ;Сравнение z4 с 0
brne  lcd_mess_0        ;Если z4<>0, то на lcd_mess_0, иначе далее
ret                                 ;Выход из подпрограммы
```

```
;-----------------------Подпрограмма отображения двух строк подряд на LCD по
данным из таблиц

lcd_all:
ldi         temp,1                  ;Отображение 1-й строки
rcall lcd_mess          ;Вызов подпрограммы отображения сообщений
ldi         temp,2                  ;Отображение 2-й строки
rcall lcd_mess          ;Вызов подпрограммы отображения сообщений
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отображения номера предела

pred_num:
ldi         lcd,0x87            ;Выбор знакоместа (конец 1й строки)
rcall lcd_com               ;

ldi         lcd,0x30            ;Определение кода для предела
add         lcd,k_del           ;
inc         lcd                       ;
rcall lcd_dat               ;
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма преобразование 24 битного числа в
двоично-десятичное
;Число для преобразования из XL, XH, результат в des_tysh, tyshi, sotni,
desyatki,
;edinicy, desyatye, sotye

bin2dec:
clr         des_tysh            ;Сброс десятков тысяч
clr         tyshi               ;Сброс тысяч
clr         sotni               ;Сброс сотен
clr         desyatki            ;Сброс десятков
clr         edinicy                   ;Сброс единиц
clr         desyatye            ;Сброс десятых долей
clr         sotye               ;Сброс сотых долей

des_tysh0:
ldi         z1,0x40                   ;Занесение 1000000 в z1...z3
ldi         z2,0x42                   ;
ldi         z3,0x0F                   ;

des_tysh1:
rcall c_minus_z         ;Вычитание 1000000
brmi  des_tysh2         ;Если результат вычисления отрицательный, то переход
на des_tysh2
inc         des_tysh            ;Иначе увеличение на 1 числа десятков тысяч
rjmp  des_tysh1         ;На des_tysh1

des_tysh2:
rcall c_plus_z          ;Прибавление 1000000 к отрицательному результату

tyshi0:
ldi         z1,0xA0                   ;Занесение 100000 в z1...z3
ldi         z2,0x86                   ;
ldi         z3,0x01                   ;

tyshi1:
rcall c_minus_z         ;Вычитание 100000
```

```
brmi  tyshi2                ;Если результат вычисления отрицательный, то
переход на tyshi2
inc         tyshi           ;Иначе увеличение на 1 числа тысяч
rjmp  tyshi1                ;На tyshi1

tyshi2:
rcall c_plus_z          ;Прибавление 100000 к отрицательному результату

sotni0:
ldi         z1,0x10             ;Занесение 10000 в z1...z3
ldi         z2,0x27             ;
ldi         z3,0x00             ;

sotni1:
rcall c_minus_z         ;Вычитание 10000
brmi  sotni2                ;Если результат вычисления отрицательный, то
переход на sotni2
inc         sotni           ;Иначе увеличение на 1 числа сотен
rjmp  sotni1                ;На sotni1

sotni2:
rcall c_plus_z          ;Прибавление 10000 к отрицательному результату

desyatki0:
ldi         z1,0xE8             ;Занесение 1000 в z1...z3
ldi         z2,0x03             ;

desyatki1:
rcall c_minus_z         ;Вычитание 1000
brmi  desyatki2         ;Если результат вычисления отрицательный, то переход
на desyatki2
inc         desyatki        ;Иначе увеличение на 1 числа десятков
rjmp  desyatki1         ;На desyatki1

desyatki2:
rcall c_plus_z          ;Прибавление 1000 к отрицательному результату

edinicy0:
ldi         z1,0x64             ;Занесение 100 в z1...z3
ldi         z2,0x00             ;

edinicy1:
rcall c_minus_z         ;Вычитание 100
brmi  edinicy2          ;Если результат вычисления отрицательный, то переход
на edinicy2
inc         edinicy             ;Иначе увеличение на 1 числа единиц
rjmp  edinicy1          ;На edinicy1

edinicy2:
rcall c_plus_z          ;Прибавление 100 к отрицательному результату

desyatye0:
ldi         z1,0x0A             ;Занесение 10 в z1...z3

desyatye1:
rcall c_minus_z         ;Вычитание 10
brmi  desyatye2         ;Если результат вычисления отрицательный, то переход
на desyatye2
inc         desyatye        ;Иначе увеличение на 1 числа десятых долей
rjmp  desyatye1         ;На desyatye1
```

```
desyatye2:
rcall c_plus_z          ;Прибавление 10 к отрицательному результату

sotye0:
ldi         z1,0x01                 ;Занесение 1 в z1...z3

sotye1:
rcall c_minus_z         ;Вычитание 1
brmi  bin2dec1          ;Если результат вычисления отрицательный, то переход
на bin2dec1
inc         sotye               ;Иначе увеличение на 1 числа сотых долей
rjmp  sotye1                    ;На sotye1

bin2dec1:
clr         c1                      ;Очистка c1
clr         c2                      ;Очистка c2
clr         c3                      ;Очистка c3
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма перемножения 16-ти битных чисел с
оптимизацией
;времени выполнения засчёт смены умножаемого и множителя местами в
зависимости от их величин
;1-е число: (c4+c3+c2+c1), 2-е число: (XH+XL), Результат: (c4+c3+c2+c1)

;-----------------------Очистка используемых регистров

mul16x16:
clr         temp                ;Очистка temp
clr         c4                      ;Очистка c4

;-----------------------Оптимизация времени вычисления

mul16x16_0:
cp          c1,XL               ;Сравнение 1-ых байтов чисел
cpc         c2,XH               ;Сравнение 2-ых байтов чисел с учётом переноса
cpc         c3,temp                 ;Сравнение 3-их байтов чисел с учётом
переноса
brsh  mul16x16_1        ;Если c3+c2+c1 больше или равно XH+XL, то на
mul16x16_1,
push  c1                        ;иначе далее (смена чисел местами через стек)
push  c2                        ;
push  XL                        ;
push  XH                        ;
pop         c2                      ;
pop         c1                      ;
pop         XH                      ;
pop         XL                      ;

;-----------------------Подготовка к умножению

mul16x16_1:
mov         a1,c1               ;Копирование 1-го байта умножаемого в a1
mov         a2,c2               ;Копирование 2-го байта умножаемого в a1
mov         a3,c3               ;Копирование 3-го байта умножаемого в a1

;-----------------------Умножение

mul16x16_2:
sbiw  X,1                       ;Уменьшение множителя на 1
cp          XL,temp                 ;Сравнение результата с 0
```

```
cpc         XH,temp                 ;Сравнение результата с 0, с учётом
переноса
breq  mul16x16_3        ;Если множитель равен нулю, то переход на mul16x16_3,
иначе далее
add         c1,a1               ;Сложение c1 и a1, результат в c1
adc         c2,a2               ;Сложение c2 и a2 с учётом переноса, результат
в c2
adc         c3,a3               ;Сложение c3 и a3 с учётом переноса, результат
в c3
adc         c4,temp                 ;Сложение c4 и temp с учётом переноса,
результат в c4
rjmp  mul16x16_2        ;На mul16x16_2

mul16x16_3:
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма деления 32-х битного на 16-ти битное
число
;Делимое: (c4+c3+c2+c1), Делитель: (XH+XL), Результат: (c3+c2+c1)
(b3+b2+b1)

div32_16:
clr         temp                ;Очистка temp
clr         temp2               ;Очистка temp2
clr         XL                      ;Очистка XL
clr         XH                      ;Очистка XH

div32_16_0:
sub         c1,a1               ;Вычитание a1 из c1
sbc         c2,a2               ;Вычитание a2 из c2 с учётом переноса
sbc         c3,a3               ;Вычитание a3 из c3 с учётом переноса
sbc         c4,temp                 ;Вычитание temp из c4 с учётом переноса
brcs  div32_16_1        ;Если результат вычисления отрицательный, то переход
на div32_16_1
adiw  X,1                       ;Прибавление 1 к регистровой паре X
adc         temp2,temp          ;Сложение temp2 и temp с учётом переноса,
результат в temp2
rjmp  div32_16_0        ;Переход на div32_16_0

div32_16_1:
mov         c1,XL               ;Копирование 1-го байта результата из XL в c1
mov         c2,XH               ;Копирование 2-го байта результата из XH в c2
mov         c3,temp2            ;Копирование 3-го байта результата из temp2 в
c3
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма деления 32-х битного числа на 2 через
перенос вправо

del32_2:
lsr         c4                      ;/2 (итого /2)
ror         c3                      ;
ror         c2                      ;
ror         c1                      ;
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма деления 32-х битного числа на 4 через
перенос вправо

del32_4:
rcall del32_2               ;/2 (итого /2)
```

```
rcall del32_2                ;/2 (итого /4)
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма деления 32-х битного числа на 8 через
перенос вправо

del32_8:
rcall del32_2                ;/2 (итого /2)
rcall del32_4                ;/4 (итого /8)
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма деления 32-х битного числа на 32 через
перенос вправо

del32_32:
rcall del32_4                ;/4 (итого /4)
rcall del32_8                ;/8 (итого /32)
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма вычитания (c3+c2+c1)-
(z3+z2+z1)=(c3+c2+c1)

c_minus_z:
sub         c1,z1            ;
sbc         c2,z2            ;
sbc         c3,z3            ;
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма сложения
(c3+c2+c1)+(z3+z2+z1)=(c3+c2+c1)

c_plus_z:
add         c1,z1            ;
adc         c2,z2            ;
adc         c3,z3            ;
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма задержки (~1мкс при 20МГц)
;Число тактов для выполнения (3*N)+4

del1u:
ldi         temp2,d1u        ;Запись d1u в temp2

del1u_1:
dec         temp2            ;Уменьшение temp2 на 1
brne  del1u_1                ;Если результат не равен 0, то переход на
del1u_1
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма задержки (~40мкс при 20МГц)
;Число тактов для выполнения (3*N)+4

del40u:
ldi         temp2,d40u       ;Запись d40u в temp2

del40u_1:
dec         temp2            ;Уменьшение temp2 на 1
brne  del40u_1         ;Если результат не равен 0, то переход на del40u_1
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма задержки (~3мс при 20МГц)
```

```
;Число тактов для выполнения (6*N)+9

del3m:
ldi         XL,low(d3m)       ;Задание длительности
ldi         XH,high(d3m)      ;Задание длительности
clr         temp2             ;Очистка temp

del3m_1:
sbiw  X,1                     ;Вычитание из регистровой пары X числа 1
cp          XL,temp2          ;Сравнение результата с 0
cpc         XH,temp2          ;Сравнение результата с 0, с учётом переноса
brne  del3m_1                 ;Если результат не равен 0, то переход на
del3m_1
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма задержки (~20мс при 20МГц)
;Число тактов для выполнения (6*N)+9

del20m:
ldi         XL,low(d20m)      ;Задание длительности
ldi         XH,high(d20m)     ;Задание длительности
clr         temp2             ;Очистка temp

del20m_1:
sbiw  X,1                     ;Вычитание из регистровой пары X числа 1
cp          XL,temp2          ;Сравнение результата с 0
cpc         XH,temp2          ;Сравнение результата с 0, с учётом переноса
brne  del20m_1          ;Если результат не равен 0, то переход на del20m_1
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма задержки (~300мс при 20МГц)

del300m:
ldi         temp,15               ;Задание длительности

del300m_1:
rcall del20m                  ;Вызов подпрограммы задержки (~20мс)
dec         temp              ;Уменьшение temp на 1
cpi         temp,0                  ;Сравнение temp с 0
brne  del300m_1         ;Если temp<>0, то на del300m_1, иначе далее
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отправки команды на LCD

lcd_com:
cbi         PORTB,RS          ;Команда
in          temp,PORTB
andi  temp,0b00000011  ;Маска 00000011 для обнуления лишнего
ori         temp,0b00001000   ;Логическое сложение temp с 00001000 (строб
вверх + команда)
rcall lcd_out                 ;Вызов подпрограммы lcd_out
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отправки данных на LCD

lcd_dat:
sbi         PORTB,RS          ;Данные
in          temp,PORTB
andi  temp,0b00000011  ;Маска 00000011 для обнуления лишнего
ori         temp,0b00001100   ;Логическое сложение temp с 00001100 (строб
вверх + команда)
```

```
rcall lcd_out                ;Вызов подпрограммы lcd_out
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма отправки на LCD

lcd_out:
rcall del1u             ;Вызов подпрограммы задержки (~1мкс)
sbi         PORTB,E                ;Строб вверх
push  lcd                  ;Сохранение lcd в стеке
andi  lcd,0b11110000    ;Маска 11110000 для обнуления младшего полубайта
or          lcd,temp          ;Логическое сложение lcd с temp (строб вверх +
команда)
out         PORTB,lcd         ;Запись команды в порт
rcall del1u             ;Вызов подпрограммы задержки (~1мкс)
cbi         PORTB,E                ;Строб вниз
rcall del40u               ;Вызов подпрограммы задержки (~40мкс)

sbi         PORTB,E                ;Строб вверх
pop         lcd                    ;Извлечение lcd из стека
swap  lcd                  ;Смена полубайт местами
andi  lcd,0b11110000    ;Маска 11110000 для обнуления младшего полубайта
or          lcd,temp          ;Логическое сложение lcd с temp (строб вверх +
команда)
out         PORTB,lcd         ;Запись команды в порт
rcall del1u             ;Вызов подпрограммы задержки (~1мкс)
cbi         PORTB,E                ;Строб вниз
rcall del40u               ;Вызов подпрограммы задержки (~40мкс)
rcall del40u               ;Вызов подпрограммы задержки (~40мкс)
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма ожидания готовности буфера передачи

USART_ready:
sbis  UCSRA,UDRE        ;Если UDRE установлен, то пропуск след. команды
rjmp  USART_ready       ;На USART_ready
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма записи в ячейку EEPROM
;Адрес записываемой ячейки в temp2, записываемый байт в temp

EEPROM_write:
out         EEAR,temp2         ;Переносим адрес в EEAR
out         EEDR,temp          ;Переносим записывемый байт в EEDR

EEPROM_write_1:
sbic  EECR,EEPE         ;Если EERE сброшен, то пропуск след. команды
rjmp  EEPROM_write_1    ;На EEPROM_write_1
sbi         EECR,EEMPE         ;Разрешаем запись в EEPROM (устанавливаем
EEMPE)
sbi         EECR,EEPE          ;Записываем байт в ячейку EEPROM (устанавливаем
EEPE)
rcall del3m             ;Вызов подпрограммы задержки (~3мс)
rcall del3m             ;Вызов подпрограммы задержки (~3мс)
ret                                ;Выход из подпрограммы

;-----------------------Подпрограмма чтения ячейки EEPROM
;Адрес читаемой ячейки в temp2, прочитанная ячейка в temp2

EEPROM_read:
out         EEAR,temp2         ;Переносим адрес в EEAR
```

```
EEPROM_read_1:
sbic  EECR,EEPE         ;Если EEPE сброшен, то пропуск след. команды
rjmp  EEPROM_read_1     ;На EEPROM_read_1
sbi         EECR,EERE         ;Читаем ячейку EEPROM (устанавливаем EERE)
in          temp2,EEDR        ;Переносим прочитанный байт в temp2
ret                           ;Выход из подпрограммы

;-----------------------Подпрограмма изменения поправки в зависимости от
предела

predel:
ldi         temp,1                ;Запись 1 в temp
clr         temp2             ;Очистка temp2

cpi         k_del,0               ;Если поправка предела равна 0 (clk/1), то
переход на predel5
breq  predel5                 ;иначе далее

cpi         k_del,1               ;Если поправка предела равна 1 (clk/8), то
переход на predel4
breq  predel4                 ;иначе далее

cpi         k_del,2               ;Если поправка предела равна 2 (clk/64),
то переход на predel3
breq  predel3                 ;иначе далее

cpi         k_del,3               ;Если поправка предела равна 3 (clk/256),
то переход на predel2
breq  predel2                 ;иначе далее

;-----------------------Деление на 8...1024 с помощью переносов

predel1:
lsr         z2                    ;/2 (итого /2)
ror         z1                    ;
lsr         z2                    ;/2 (итого /4)
ror         z1                    ;

predel2:
lsr         z2                    ;/2 (итого /2)
ror         z1                    ;
lsr         z2                    ;/2 (итого /4)
ror         z1                    ;

predel3:
lsr         z2                    ;/2 (итого /2)
ror         z1                    ;
lsr         z2                    ;/2 (итого /4)
ror         z1                    ;
lsr         z2                    ;/2 (итого /8)
ror         z1                    ;

predel4:
lsr         z2                    ;/2 (итого /2)
ror         z1                    ;
lsr         z2                    ;/2 (итого /4)
ror         z1                    ;
lsr         z2                    ;/2 (итого /8)
ror         z1                    ;

predel5:
```

```
cp          z1,temp                 ;Сравнение z1 с temp
cpc         z2,temp2          ;Сравнение z2 с temp2 с учётом переноса
brsh  predel6                 ;Если z2+z1>=1, то на predel6, иначе далее
clr         z1                      ;Очистка z1
clr         z2                      ;Очистка z2

predel6:
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма отображения числовых строк (для
отладки)

otlad:
sbic  PIND,6                  ;Если PD6=0, то пропуск след. команды
rjmp  otlad_2                 ;На otlad_2
rcall del3m             ;Вызов подпрограммы задержки (~3мс)
sbic  PIND,6                  ;Если PD6=0, то пропуск след. команды
rjmp  otlad_2                 ;На otlad_2

rcall       lcd_both_str      ;Вызов подпрограммы отображения двух числовых
строк

otlad_1:
rcall knop_off          ;Вызов подпрограммы ожидания отпускания кнопки

pop         temp                ;Очистка стека (поскольку выход не по ret)
pop         temp                ;Очистка стека
rjmp  main              ;На main

otlad_2:
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма ожидания отпускания кнопки

knop_off:
sbis  PIND,6                  ;Если PD6=1, то пропуск след. команды
rjmp  knop_off          ;На knop_off
rcall del3m             ;Вызов подпрограммы задержки (~3мс)
sbis  PIND,6                  ;Если PD6=1, то пропуск след. команды
rjmp  knop_off          ;На knop_off
ret                                 ;Выход из подпрограммы

;-----------------------Подпрограмма передачи с 1-го по 7-й байт через USART

usart_1_7:
rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи
ldi   temp,0x10    ;Запись 0x10 в temp
or des_tysh,temp ;Логическое сложение des_tysh с temp, результат в des_tysh
out   UDR,des_tysh      ;Передача из des_tysh в USART

rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи
ldi   temp,0x20         ;Запись 0x20 в temp
or    tyshi,temp        ;Логическое сложение tyshi с temp, результат в tyshi
out   UDR,tyshi         ;Передача из tyshi в USART

rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи
ldi   temp,0x30         ;Запись 0x30 в temp
or    sotni,temp        ;Логическое сложение sotni с temp, результат в sotni
out   UDR,sotni         ;Передача из sotni в USART
```

```
rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи
ldi   temp,0x40    ;Запись 0x40 в temp
or    desyatki,temp ;Логическое сложение desyatki с temp, результат в
desyatki
out UDR,desyatki  ;Передача из desyatki в USART

rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи
ldi   temp,0x50   ;Запись 0x50 в temp
or edinicy,temp ;Логическое сложение edinicy с temp, результат в edinicy
out         UDR,edinicy      ;Передача из edinicy в USART

rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи
ldi   temp,0x60         ;Запись 0x60 в temp
or    desyatye,temp     ;Логическое сложение desyatye с temp, результат в
desyatye
out   UDR,desyatye      ;Передача из desyatye в USART

rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи
ldi   temp,0x70   ;Запись 0x70 в temp
or    sotye,temp  ;Логическое сложение sotye с temp, результат в sotye
out   UDR,sotye   ;Передача из sotye в USART

rcall USART_ready      ;Вызов подпрограммы ожидания готовности буфера
передачи
ret                                ;Выход из подпрограммы

;----------------------Подпрограмма отображения одной числовой строки

lcd_one_str:
ldi   lcd,0x4E          ;Отображение "N"
rcall lcd_dat                ;

ldi   lcd,0x3A          ;Отображение ":"
rcall lcd_dat                ;

ldi   lcd,0x30          ;Определение кода для числа тысяч
add   lcd,tyshi         ;
rcall lcd_dat                ;Отображение числа тысяч

ldi   lcd,0x30          ;Определение кода для числа сотен
add   lcd,sotni         ;
rcall lcd_dat                ;Отображение числа сотен

ldi   lcd,0x30          ;Определение кода для числа десятков
add   lcd,desyatki      ;
rcall lcd_dat                ;Отображение числа десятков

ldi   lcd,0x30          ;Определение кода для числа единиц
add   lcd,edinicy;
rcall lcd_dat                ;Отображение числа единиц

ldi   lcd,0x30          ;Определение кода для числа десятых долей
add   lcd,desyatye      ;
rcall lcd_dat                ;Отображение числа десятых долей

ldi   lcd,0x30          ;Определение кода для числа сотых долей
add   lcd,sotye         ;
rcall lcd_dat                ;Отображение числа сотых долей
ret                               ;Выход из подпрограммы
```

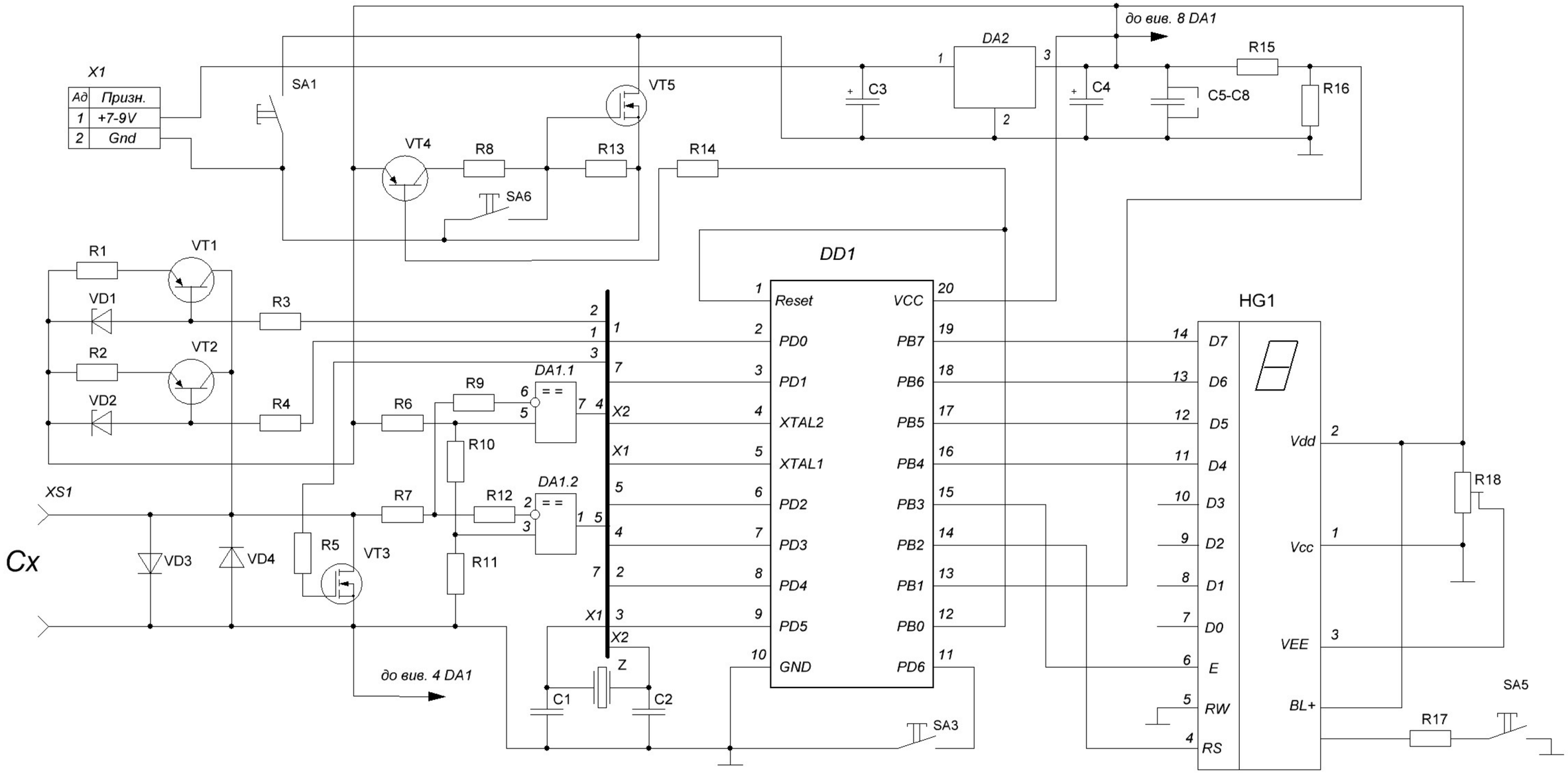
```
;-----------------------Подпрограмма отображения двух числовых строк по
данным

lcd_both_str:
mov   c1,a1             ;Копирование из a1 в c1
mov   c2,a2             ;Копирование из a2 в c2
mov   c3,a3             ;Копирование из a3 в c3

;-----------------------Перевод двоичного результата коэффициента в двоично-
десятичный

rcall bin2dec                ;Вызов подпрограммы перевода двоичного числа в
двоично-десятичное

;-----------------------Отображение смещения от нуля для U1 "1=     "

ldi   lcd,0x80          ;Выбор знакоместа (начало 1й строки)
rcall lcd_com                ;

rcall lcd_one_str       ;Вызов подпрограммы отображения одной числовой строки

mov   c1,b1             ;Копирование из b1 в c1
mov   c2,b2             ;Копирование из b2 в c2
mov   c3,b3             ;Копирование из b3 в c3

;-----------------------Перевод двоичного результата коэффициента в двоично-
десятичный

rcall bin2dec                ;Вызов подпрограммы перевода двоичного числа в
двоично-десятичное

;-----------------------Отображение смещения от нуля для U2 "2=     "

ldi   lcd,0xC0          ;Выбор знакоместа (начало 2й строки)
rcall lcd_com           ;

rcall lcd_one_str       ;Вызов подпрограммы отображения одной числовой строки

ret                                ;Выход из подпрограммы

;-------------Подпрограмма обработки прерывания по переполнению таймера 1
;Поскольку выход не по reti, то глобально запрещённые прерывания остаются
запрещены

ovtim1:
cli                          ;Глобально запрещаем прерывания
ldi   temp,3                 ;Заносим 3 в temp
cp    c3,temp                ;Сравнение c3 с temp
breq  ovtim1_0          ;Если c3=3, то на ovtim1_0, иначе далее
inc   c3                     ;Увеличение c3 на 1
reti                         ;Выход из подпрограммы обработки прерывания

ovtim1_0:
sbi   PORTD,0                ;Выключение большого тока заряда конденсатора
sbi   PORTD,4                ;Выключение малого тока заряда конденсатора
sbi   PORTD,5                ;Включение разряда конденсатора
ldi   temp,0b00000000   ;Останов таймера
out   TCCR1B,temp       ;
pop   temp              ;Очистка стека (поскольку выход по clk_down, а не по
reti)
pop   temp              ;Очистка стека
```

```
rjmp  clk_down          ;Переход на clk_down

;-----------------------Подпрограмма обработки внешнего прерывания по U1
;Поскольку выход не по reti, то глобально запрещённые прерывания остаются
запрещены

com_U1:
in    a1,TCNT1L         ;Сохраняем 1-й байт n1
in    a2,TCNT1H         ;Сохраняем 2-й байт n1
mov   a3,c3             ;Копируем из c3 в a3 3-й байт n1
ldi   temp,0b10000000   ;Разрешение внешнего прерывания только по PD3 (U2)
out   GIMSK,temp        ;
reti                          ;Выход из подпрограммы обработки прерывания

;-----------------------Подпрограмма обработки внешнего прерывания по U2

com_U2:
in    c1,TCNT1L         ;Сохраняем 1-й байт n2
in    c2,TCNT1H         ;Сохраняем 2-й байт n2
cli                     ;Глобально запрещаем прерывания
sbi   PORTD,0           ;Выключение большого тока заряда конденсатора
sbi   PORTD,4           ;Выключение малого тока заряда конденсатора
sbi   PORTD,5           ;Включение разряда конденсатора
ldi   temp,0b00000000   ;Останов таймера
out   TCCR1B,temp       ;
pop   temp              ;Очистка стека (поскольку выход не по reti)
pop   temp              ;Очистка стека

;-----------------------Проверка (c2+c1)-(a2+a1)<0 и (c2+c1)<>0

com_U2_0:
mov   b1,c1             ;Копирование в b1 из c1 (n2)
mov   b2,c2             ;Копирование в b2 из c2
mov   b3,c3             ;Копирование в b3 из c3
sub   c1,a1             ;Определение разности n2-n1
sbc   c2,a2             ;результат в c3+c2+c1
sbc   c3,a3             ;

cpi   tok_zar,0         ;Сравнение tok_zar с 0
breq  com_U2_0_1        ;Если tok_zar=0, то на com_U2_0_1, иначе далее

com_U2_0_0:
ldi   temp,255          ;Запись 255 temp
rjmp  com_U2_0_2        ;На com_U2_0_2

com_U2_0_1:
ldi   temp,30                 ;Запись 30 temp

com_U2_0_2:
clr   temp2             ;Очистка temp2
cp    c1,temp                 ;Сравнение c1 с temp
cpc   c2,temp2          ;Сравнение c2 с temp2 с учётом переноса
cpc   c3,temp2          ;Сравнение c3 с temp2 с учётом переноса
brlo  com_U2_1          ;Если (c2+c1)<30(255), то на com_U2_1, иначе далее

rcall otlad             ;Вызов подпрограммы отображения числовых строк (для
отладки)

rjmp  step1             ;На step1

;-----------------------Калибровка
```

```
com_U2_1:
cpi   k_del,0               ;Сравнение k_del с 0
breq  com_U2_2          ;Если k_del=0, то на com_U2_2, иначе далее
rjmp  com_U2_5          ;На com_U2_5

com_U2_2:
cpi   tok_zar,0         ;Сравнение tok_zar с 0
breq  com_U2_2_0        ;Если tok_zar=0, то на com_U2_2_0, иначе далее
clr   tok_zar               ;Очистка tok_zar
rjmp  com_U2_5          ;На com_U2_5

com_U2_2_0:
sbic  PIND,6                ;Если PD6=0, то пропуск след. команды
rjmp  com_U2_5          ;На com_U2_5
rcall del3m             ;Вызов подпрограммы задержки (~3мс)
sbic  PIND,6                ;Если PD6=0, то пропуск след. команды
rjmp  com_U2_5          ;На com_U2_5

;-----------------------Проверка флага записи в EEPROM

ldi   temp2,flag        ;Записывает в temp2 адрес читаемой ячейки EEPROM
rcall EEPROM_read       ;Вызов подпрограммы чтения ячейки EEPROM

cpi   temp2,100         ;Сравнение temp2 с 100
brne  com_U2_3          ;Если temp2<>100, то на com_U2_3, иначе далее

;-----------------------Сброс флага записи в EEPROM

ldi   temp,0xFF         ;Заносим 0xFF в temp
ldi   temp2,flag        ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

;-----------------------Отображение "Калибр. " и "откл.   "

ldi   ZL,low(Popr_off*2)     ;Запись в ZL младшего байта адреса ячейки из
таблицы Popr_off (альтернативная адресация)
ldi   ZH,high(Popr_off*2)    ;Запись в ZH старшего байта адреса ячейки из
таблицы Popr_off (альтернативная адресация)
rcall lcd_all               ;Вызов подпрограммы отображения двух строк
подряд на LCD по данным из таблиц

rcall knop_off          ;Вызов подпрограммы ожидания отпускания кнопки

rjmp  main              ;На main

;-----------------------Запись в EEPROM

;-----------------------Запись n1 (1-й и 2-й байт)

com_U2_3:
mov   temp,a1               ;Копируем a1 в temp
ldi   temp2,n1_1b       ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

mov   temp,a2               ;Копируем a2 в temp
ldi   temp2,n1_2b       ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

;-----------------------Запись n2 (1-й и 2-й байт)
```

```
mov   temp,b1                ;Копируем b1 в temp
ldi   temp2,n2_1b       ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

mov   temp,b2                ;Копируем b2 в temp
ldi   temp2,n2_2b       ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

;-----------------------Запись n2-n1 (1-й и 2-й байт)

mov   temp,c1                ;Копируем c1 в temp
ldi   temp2,n2_n1_1b    ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

mov   temp,c2                ;Копируем c2 в temp
ldi   temp2,n2_n1_2b    ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

;-----------------------Запись флага записи в EEPROM

ldi   temp,100          ;Заносим 100 в temp
ldi   temp2,flag        ;Заносим в temp2 адрес ячейки EEPROM
rcall EEPROM_write      ;Вызов подпрограммы записи в EEPROM

rcall lcd_both_str      ;Вызов подпрограммы отображения двух числовых строк

;-----------------------Предотвращение многократной записи в EEPROM при
удержании кнопки

rcall knop_off          ;Вызов подпрограммы ожидания отпускания кнопки

ldi   ZL,low(Popr_on*2);Запись в ZL младшего байта адреса ячейки из таблицы
Popr_on (альтернативная адресация)
ldi   ZH,high(Popr_on*2)     ;Запись в ZH старшего байта адреса ячейки из
таблицы Popr_on (альтернативная адресация)
rcall lcd_all                       ;Вызов подпрограммы отображения двух строк
подряд на LCD по данным из таблиц
rcall del300m                       ;Вызов подпрограммы задержки (~300мс)

rjmp  main                  ;На main

;-----------------------Отображение "Cx < min" и "Rx > max"

com_U2_5:
ldi   ZL,low(Cmn_Rmx*2);Запись в ZL младшего байта адреса ячейки из таблицы
Cmin_Rmax (альтернативная адресация)
ldi   ZH,high(Cmn_Rmx*2)     ;Запись в ZH старшего байта адреса ячейки из
таблицы Cmin_Rmax (альтернативная адресация)
rcall lcd_all                       ;Вызов подпрограммы отображения двух строк
подряд на LCD по данным из таблиц
inc   autoff                        ;Увеличение autoff на 1
;-----------------------Передача сообщения "Cx < min" и "ESR > max"
rcall USART_ready       ;Вызов подпрограммы ожидания готовности буфера
передачи

ldi   temp,0xD1    ;Запись 0xD1 в temp
out   UDR,temp     ;Передача из temp в USART
rcall USART_ready  ;Вызов подпрограммы ожидания готовности буфера передачи

rjmp  clk_up                ;На уменьшение предела измерения (на clk_up)
;-----------------------Таблица сообщений
```

```
Prvt:
.db   0x43,0x20,0x26,0x20,0x45,0x53,0x52,0x20       ;"C & ESR "
.db   0x76,0x65,0x72,0x2E,0x20,0x31,0x2E,0x30       ;"ver. 1.0"

Pred:
.db   0x52,0x61,0x6E,0x67,0x65,0x3A,0x20,0x20       ;"Range:  "
.db   0x77,0x61,0x69,0x74,0x2E,0x2E,0x2E,0x20       ;"wait... "
Cmn_Rmx:
.db   0x43,0x78,0x20,0x3C,0x20,0x6D,0x69,0x6E       ;"Cx < min"
.db   0x45,0x53,0x52,0x3E,0x20,0x6D,0x61,0x78       ;"ESR> max"

ESRmn:
.db   0x3C,0x30,0x2C,0x30,0x31,0x20,0x4F,0x6D       ;"<0,01 Om"
ESRmx:
.db   0x3E,0x32,0x30,0x2C,0x30,0x20,0x4F,0x6D       ;">20,0 Om"
KZ_Cmx:
.db   0x43,0x78,0x20,0x3E,0x20,0x6D,0x61,0x78       ;"Cx > max"
.db   0x6F,0x72,0x20,0x73,0x68,0x6F,0x72,0x74       ;"or short"
Popr_on:
.db   0x43,0x6F,0x72,0x72,0x65,0x63,0x2D,0x20       ;"Correc-"
.db   0x74,0x69,0x6F,0x6E,0x20,0x4F,0x4E,0x20       ;"tion ON "
Popr_off:
.db   0x43,0x6F,0x72,0x72,0x65,0x63,0x2D,0x20       ;"Correc-"
.db   0x74,0x69,0x6F,0x6E,0x20,0x4F,0x46,0x46       ;"tion OFF"
Bat:
.db   0x52,0x65,0x70,0x6C,0x61,0x63,0x65,0x20       ;"Replace "
.db 0x62,0x61,0x74,0x74,0x65,0x72,0x79,0x21   ;"battery!"
Gdit:
.db   0x77,0x61,0x69,0x74,0x2E,0x2E,0x2E,0x20       ;"wait... "
```



					08-23.МКР.001.00.000 Е3	
					Методи та засіб діагностики якості електролітичних конденсаторів комп'ютерної техніки	Лит / Масса / Масштаб
Изм	Лист	№ докум.	Подпись	Дата		Лист / Листов
Розробив		Кірше А.О.			Принципова схема пристрою діагностики якості електролітичних конденсаторів	ВНТУ гр. 1КІ-18м
Провер.		Роптанов В. І.				
Т. контр.						
Реценз.						
Н. контр.		Швець С.І.				
Затверд.		Мартинюк Т.Б.				

Поз. познач.	Найменування	Кіл.	Примітка
	Конденсатори		
C1	К10-7В-Н10-0,1мкФ ± 10%	1	
C2, C3	К10 -17а - М33 - 22 пФ + 5%	2	
C4,C5	К10-7В-Н10-0,1мкФ ± 10%	2	
C6	СарХоп-16В-1000 мкФ	1	
C7	К10 -17а - М33 - 47 пФ + 5%	1	
C8	СарХоп-25В-2200 мкФ	1	
C9- C10	К10 -17а - М33 - 47 пФ + 5%	2	
C11	К10 -17а – Н30 - 200 пФ + 5%	1	
C12	К10-7В-Н10-0,1мкФ ± 10%	1	
DA1	Мікросхема LM78L05	1	
DD1	Мікросхема ATtiny2313	1	
HG1	Індикатор ВТ-А554RD	1	
	Резистори		
R1	С2 – 23 – 0,125 – 10 кОм ±5%	1	
R2-R8	С2 – 23 – 0,125 – 200 Ом ±5%	7	
R9	С2 – 23 – 0,125 – 100 Ом ±5%	1	
R10, R11	С2 – 23 – 0,125 – 10 кОм ±5%	2	
R12	С2 – 23 – 0,125 –1 мОм ±5%	1	
R13	С2 – 23 – 0,125 – 1 кОм ±5%	1	
R14	С2 – 23 – 0,125 – 10 кОм ±5%	1	
R15	С2 – 23 – 0,125 – 1 кОм ±5%	1	
R16	С2 – 23 – 0,125 – 10 кОм ±5%	1	
R17	С2 – 23 – 0,125 – 1 кОм ±5%	1	
R18,R19	С2 – 23 – 0,125 – 10 кОм ±5%	2	
R20,R21	С2 – 23 – 0,125 – 1 кОм ±5%	2	
R22	С2 – 23 – 0,125 – 10 кОм ±5%	2	
R23	С2 – 23 – 0,125 – 1 кОм ±5%	1	
R24	С2 – 23 – 0,125 – 10 Ом ±5%	1	

Змн.	Арк.	№ докум.	Підпис	Дата	08-23.МКР.001.00.000 ПЕЗ		
Розроб.		Кірше А. О.			Методи та засіб діагностики якості електролітичних конденсаторів комп'ютерної техніки Перелік елементів	Літ.	Арк. 103 / Аркушів 2
Перевір.		Роптанов В. І.					
Реценз.						Гр 1КІ-18 м	
Н. Контр.		Швець С.І.					
Затверд.		Мартинюк Т.Б.					

Поз. познач.	Найменування	Кіл.	Примітка
VD1-VD6	Діод 1N4148	6	
VT1-VT3	Транзистор BC808-40	3	
VT4	Транзистор BC847C	1	
VT5	Транзистор IRF9510	1	
VT6	Транзистор BC808-40	1	
XS1-XS2	З'єднувач PBS-02	2	
ZQ1	Резонатор кварцовий HC-49 10м Гц	1	