ЗМІСТ

ВСТУП

Часто для швидкої розробки прототипу радіоапаратури виникає необхідність абстрагуватись від розкиду параметрів комерційних зразків транзисторів. Найбільш очевидним шляхом вирішення цієї проблеми буде підбір транзисторів з необхідними параметрами. Це в свою чергу потребує засобів, що дозволяють швидко вимірювати, обробляти та відображати необхідні параметри.

Найбільш повною характеристикою транзистора є його вольт-амперна характеристика (ВАХ). За відомими ВАХ визначають решту параметрів транзистора: коефіцієнти передачі струму та напруги, вхідний та вихідний опір, напругу насичення та інші. Ці характеристики широко використовують для розрахунку транзисторних каскадів, для побудови лінійних та інших моделей транзисторів. Саме визначення ВАХ транзистору і буде основним завданням при розробці даної системи.

Розроблена система представляє собою програмний-апаратний комплекс призначений для вимірювання, обробки та відображення характеристик біполярних транзисторів, що дозволяє проводити вимірювання в автоматизованому режимі.

Сьогодні засоби обчислювальної техніки досягли значного розвитку. Їх використання для обробки та відображення інформації на багато дешевше ніж використання вбудованих апаратних засобів обробки та відображення результатів роботи вимірювальних пристроїв. Розробка тестування і найголовніше модифікація програмних засобів обробки та відображення потребує значно менше сил та засобів ніж при використанні для цього самостійно розроблених апаратних засобів. Тому в даній системі передбачається програмна частина для візуалізації отриманих результатів.

В більшості випадків характеристики транзисторів зображуються у вигляді так званих сімейств, де наводяться декілька кривих при фіксованому значенні одного із параметрів.

Хоча розробники апаратури звикли і майже не уявляють можливість роботи з іншими формами подання характеристик, із розвитком обчислювальної техніки та засобів обробки графіки з’являється можливість більш наочного відображення. Оскільки характеристики транзисторів є функціями двох змінних найбільш простим і наочним буде відображення таких характеристик у тривимірному просторі.

В даний час немає жодного подібного приладу, що виконує візуалізацію характеристик за допомогою тривимірної графіки, придатного для реального використання. Дана робота присвячена розробці саме такого приладу.

1 АНАЛІТИЧНИЙ ОГЛЯД АНАЛОГІВ

Огляд покликаний систематизувати інформацію про існуючі прилади подібного призначення, він має не лише підтвердити доцільність розробки запропонованої системи але і дозволить вибрати успішні схемні рішення, що були реалізовані в інших подібних системах.

У розділі розглянуто прилади різних років випуску та складності, від любительських схем до дорогих комерційних рішень, що потребують промислового живлення.

Наведено загальні характеристики кожної конкретної моделі, виявлено їх переваги та недоліки.

1.1 Схема приставки до осцилографа

На рисунку 1.1 зображена схема характерографа опублікована в [1]. Схема призначена для перевірки малопотужних транзисторів обох структур. При цьому виводи транзисторів структури n-p-n включаються тільки в гнізда XS1–XS3, а транзистори структури p-n-p — в гнізда XS3–XS5.

Фіксовані струми бази для вимірюваних транзисторів отримують завдяки включенню в коло бази вагових (тобто кратних деякому значенню) резисторів R13 (R), R13 (2R), R11 (4R) за допомогою електронних ключів VT5, VT4 та VT3 відповідно. Електронні ключі, в свою чергу, керуються сигналами з виходу датчика DD1, тому в залежності від стану датчика отримують вісім значень струму бази.

Лічильник переключається імпульсами частотою 100 Гц — вони подаються на вхід C2 лічильника з колектора транзистора VT2. Сигнал подається на базу транзистора у вигляді пульсуючої напруги частотою 100 Гц з діода VD5.

На діодах VD1–VD5 зібраний випрямляч для живлення кола бази вимірюваного транзистора та мікросхеми DD1. Напруга на мікросхему подається з параметричного стабілізатора виконаного на резисторі R1 та

стабілітрона VD7 та підключеного до випрямляча. Інший параметричний стабілізатор виконаний на резисторі R2 та стабілітроні VD6, призначений для отримання напруги, що живить базове коло транзистора, що перевіряється, інакше кажучи він визначає струми через резистори R11–R13. Щоб ці струми можна було вимірювати в залежності від коефіцієнта передачі вимірюваного транзистора, в стабілізатор введено регулюючий транзистор VT1, на базу якого подається напруга з параметричного стабілізатора через змінний резистор R3. При зміні положення регулятора цього резистора змінюється напруга на резисторі навантаження R5, а значить змінюється частка струму в базовому колі вимірюваного транзистора при відкритті ключів на транзисторах VT3–VT5. Для обмеження струму в базових колах транзисторних ключів встановлені резистори R8–R10.

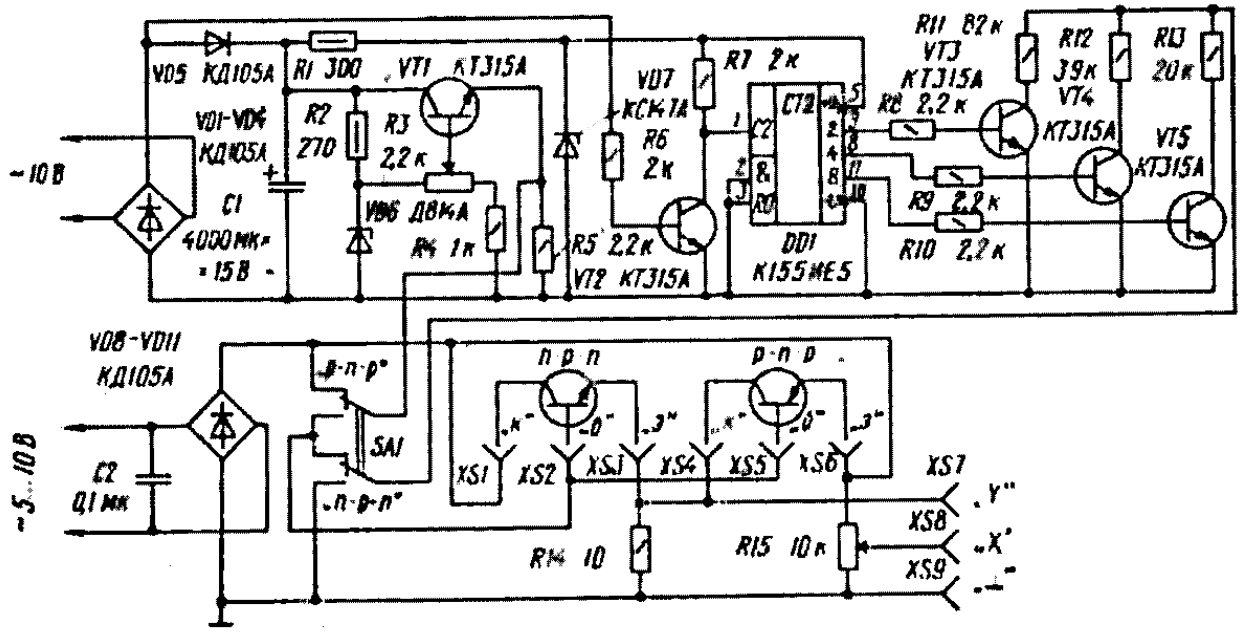

Рисунок 1.1 — Схема характерографа

На діодах VD8–VD11 зібрано інший випрямляч, але без конденсатора фільтра на виході. Тому з нього знімається пульсуюча напруга частотою 100 Гц, що використовується для живлення кола колектор-емітер вимірюваного транзистора. Напруга з резистора R14, пропорційна струму колектора транзистора структури p-n-p або струму емітера транзистора структури n-p-n, подається на вертикальний вхід осцилографа. Оскільки в

схемі включення транзистора за схемою зі спільним емітером струм колектора слабо відрізняється від струму емітера, виявилося можливим включити резистор R14 в коло емітера вимірюваного транзистора структури n-p-n. При такій схемі зміщення променя осцилографа від нульового положення відбувається вправо і вгору, тобто характеристики виходять зручними для читання.

Напрямок струму в колі бази в залежності від структури вимірюваного транзистора замінюються за допомогою перемикача SA1.

Змінну напругу на випрямлячі можна подавати лише з різних обмоток трансформатора. При чому обмотка, з якої знімається на напруга на діоди VD1–VD4, повинна мати якомога менший ємнісний зв'язок з мережевою обмоткою, інакше можуть з’явитись наводки на зображенні з частотою мережі. Найпростіше зменшити цей зв’язок за допомогою використання П-подібного магнітопроводу. Більш високочастотні завади, які можуть проникнути з мережі усуваються за допомогою конденсатора С2.

Вагові резистори R11–R13 обирають в залежності від необхідних струмів бази. Для вимірювання транзисторів малої потужності автором обрано «вагу» 20 кОм. При дослідженні більш потужних транзисторів вона можу бути збільшена. Але в будь-якому варіанті співвідношення опорів резисторів R13, R12 та R11 повинно бути 1:2:4 [1].

Основна превага наведеної схеми в її простоті, вона може бути зібрана навіть радіолюбителем. Однак вона не може функціонувати самостійно, вона потребує використання осцилографа для відображення отриманих результатів, що, в деяких випадках, робить використання схеми неможливим.

1.2 Характерограф Л2-100 ТЕКО

Більш нове рішення запропоноване фірмою «ТЕСТПРИБОР», цифровий запам'ятовує характерограф напівпровідникових приладів Л2-100 ТЕКО (рис. 1.2), що позиціонується розробниками як сучасна заміна Л2-56.

Вимірювач параметрів напівпровідникових приладів Л2-56, який колись випускався вітчизняною промисловістю, здобув заслужене визнання фахівців завдяки своїй функціональності, зручності і простоті використання.

Рисунок 1.2 — Зовнішній вигляд характерографа Л2-100

Цифровий запам'ятовує характерограф напівпровідникових приладів Л2-100 призначений для візуального спостереження статичних вольт-амперних характеристик (ВАХ) напівпровідникових приладів, вимірювання напруги на їх електродах і струмів в їх колах.

Характерограф дозволяє досліджувати ВАХ напівпровідникових діодів, стабілітронів, біполярних і польових транзисторів, тиристорів, сімісторів і інших напівпровідникових приладі, а також оптоелектронних і пасивних компонентів.

Основні параметри характерографа Л2-100:

- Максимальний струм: 50А;
- Максимальна напруга: 5000В;
- Вбудований кольоровий TFT РК-дисплей (640 × 480 точок);
- Цифрова обробка і відображення ВАХ;
- Можливість збереження до 10 ВАХ в пам'яті;
- Можливість збереження і відновлення налаштувань приладу;

– Можливість порівняння досліджуваної ВАХ з зразком;
– Підключення до персонального комп'ютера через високошвидкісний USB-порт;
– Збереження побудованих ВАХ в одному з графічних форматів при підключенні до комп'ютера;
– Збереження даних в табличному форматі CSV або XLS для подальшої обробки;

Спрощена функціональна схема характерографа представлена на рис. 1.3. Основними вузлами приладу є: джерело живлення колекторного кола, генератор ступенів напруги (струму), підсилювач індикаторний по вертикалі, підсилювач індикаторний по горизонталі.

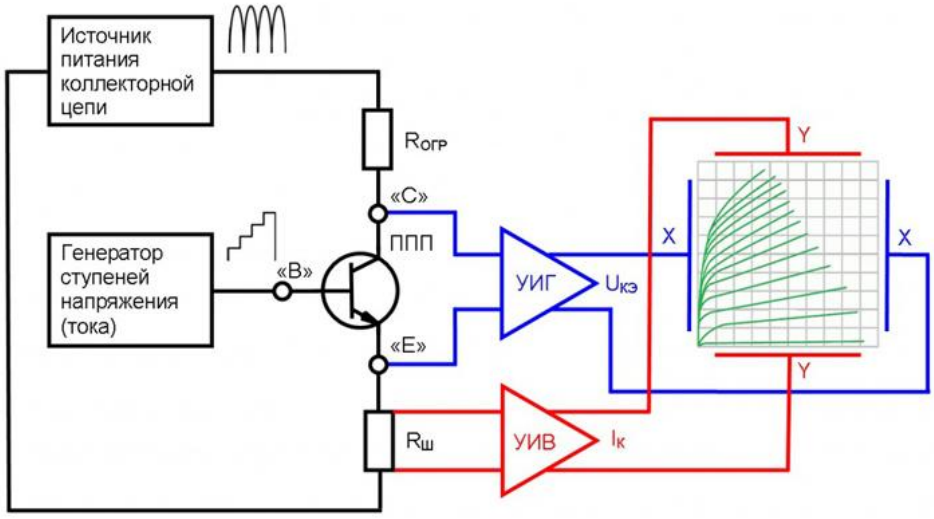

Рисунок 1.3 — Функціональна схема Л2-100

Для отримання на екрані характерографа сімейства ВАХ досліджуваного транзистора, в колекторне коло транзистора через обмежувальний резистор $R_{огр}$ подається напруга від джерела живлення колекторної ланцюга. Від генератора кроку на керуючий електрод транзистора подається струм або напругу, що змінюється східчасто. При цьому в колекторному колі транзистора виникають імпульси струму, що створюють пропорційне падіння напруги на вимірювальному шунті $R_ш$. Напруга між колектором і емітером транзистора і напруга на вимірювальному шунті надходять через

відповідні підсилювачі в канали горизонтальної та вертикальної розгортки характерографа, які, у свою чергу, формують зображення сімейства ВАХ на екрані (рис. 1.3).

Характерограф також оснащений високовольтним джерелом (на схемі не показаний), який дозволяє досліджувати ВАХ транзисторів при напругах до 5 кВ.

Даний характерограф цікавий також тим, що має можливість взаємодії з персональним комп'ютером. Для цього в комплект поставки входить спеціальне програмне забезпечення. На рис. 1.4 зображено зовнішній вигляд вікна програми драйвера для даного характерографа.

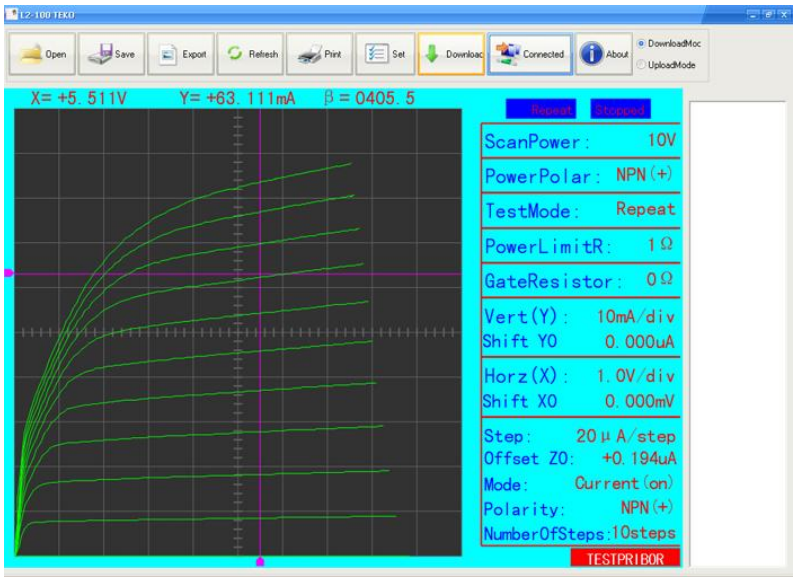

Рисунок 1.4 — Вікно програмного забезпечення

За допомогою персонального комп'ютера можна виконувати наступні функції. Здійснювати синхронне відображення ВАХ на екрані. Зберігати ВАХ разом з налаштуваннями приладу і результатами вимірювань з можливістю їх подальшого завантаження в характерограф. Роздруковувати ВАХ на принтері. Експортувати ВАХ в різноманітних графічних форматах, а також табличних форматах, зрозумілих популярним редакторам, що дозволяє проводити їх подальшу обробку [2].

Основним недоліком приладу для масового використання є його ціна. Слід відмітити, що прилад є самодостатнім і підключення до персонального

комп’ютера є додатковою функцією, а для обробки і відображення використовується вбудований процесор і монітор. Вартість приладу можна зменшити якщо перекласти ці функції на ПК.

1.3 Характерограф Type 576

Зовнішній Type 576 вигляд характерографа представлений на рис. 1.5.

Рисунок 1.5 — Зовнішній вигляд Type 576

Характерограф Type 576 динамічний напівпровідниковий тестер, який дозволяє проводити вимірювання та відображення характеристик різноманітних двох і трьох вивідних пристроїв, включаючи біполярні транзистори, польові транзистори, МОП-транзистори, кремнієві тиристори однопереходні транзистори. Він дозволяє проводити різноманітні

вимірювання з використанням схеми зі спільним емітером або спільною базою.

Прилад має в своєму розпорядженні генератори змінної та постійної напруги колектора в діапазоні від 0 до ± 1500 вольт. Генератор крок виробляє поточний крок струму або напруги, які можуть бути застосовані до будь-якого базового або емітерного виводу пристрою, що вимірюється. В режимі генерації струму можливо отримати значення струму з кроком від 5 мА до 2 А, а в режимі генерації напруги значення з кроком від 5 мВ до 40 В. Також струм або напруга може бути також отримана у вигляді коротких імпульсів. Калібрований крок зсуву дозволяє компенсувати вихід генератора кроку позитивним або негативним. Вертикальний дисплейний підсилювач дозволяє отримувати колекторний струм або струм витоку з максимальним коефіцієнтом відхилення 1 мА на поділку при проведенні вимірювань витоку. Горизонтальний підсилювач дозволяє вимірювати як колектора і напруги бази.

В таблиці 1.1 наведено основні параметри приладу.

Таблиця 1.1 — Параметри резисторів

Маса	31 кг
Висота	40 см
Ширина	30 см
Товщина	60 см
Матеріал корпусу	Алюміній
Робочий діапазон температур	Від 0°C до +50°C
Допустимі вібраційні навантаження	15 хвилин уздовж кожної осі на 0,015 дюймовий з частотою змінюється від 10-50-10 c/s в 1-хвилинних циклів.
Підключення до електромережі	Інструмент призначений для роботи від джерела живлення з його нейтральним

	або поблизу землі (земля) потенціалу. Він не призначений для роботи від двох фаз багатофазної системи. Він забезпечений шнуром живлення для підключення до джерела живлення. Третій провід підключений безпосередньо до приладової рами, і призначений для заземлення приладу для захисту обслуговуючого персоналу, відповідно до рекомендацій національних і міжнародних правил техніки безпеки.			
Напруга живлення	Від 115 В до 230 В			
Режим розгортки	Нормальний режим: змінний струм (на частотний характеристиці); повне випрямлення змінного струму. Режим постійного струму: позитивний або негативний.			
Режим пульсації постійного струму	Без навантаження не більше 2% напруги, або не більше 0,1% повного діапазону напруги.			
Точність значення напруги	Пікові напруги розімкнутого ланцюга на всіх коливається в межах + 35% і -5%			
Діапазони значень напруг	15 В	75 В	350 В	1500 В
Максимальний піковий струм (Нормальний режим)	10 А	2 А	0,5 А	0,1 А
Піковий струм (Крок генератора в імпульсному режим)	20 А	4 А	1 А	0,5 А

Мінімальний опір	0,3 Ом	6,5 Ом	140 Ом	0,3 кОм
Максимальний опір	65 кОм	1,4 МОм	6,5 МОм	6,5 МОм
Точність кроку	2% від загального значення напруги на виході, в тому числі значення зміщення, або 1% амплітудної установки перемикача, в залежності від того, що більше.			
Діапазон зміни кроку струму	Від 50 нА до 200 мА			
Діапазон зміни кроку напруги	Від 50 мВ до 2 В			

Наведені електричні та екологічні характеристики дійсні для приладів, що працюють при температурі навколишнього середовища від +1°C до +40°C після початкового прогріву протягом 5 хвилин, коли попередньо відкаліброваних при температурі + 25°C ± 5°C [3].

Як видно з характеристик приладу до його недоліків можна віднести велику вагу та необхідність трьохфазного живлення, яке можна знайти лише у спеціальних виробничих приміщеннях. Однак він дозволяє проводити вимірювання широкого діапазону напівпровідникових приладів.

1.4 Характерограф ЭРБИЙ-7107

Прилад (рис. 1.6) призначений для випробувань, досліджень напівпровідникових двополюсників: резисторів, фоторезисторів, фотодіодів, термісторів, варисторів, наноплівок і т.п. Під управлінням спеціальної комп’ютерної програми прилад задає необхідні параметри випробування і вимірює контрольні параметри випробуваного зразка.

Під час вимірюванні можна задати наступні параметри:

- Напруга вимірюваного двополюсника;
- Температура вимірюваного двополюсника;
- освітленість.

Після встановлення заданої температури і освітленості починається вимірювання. Прилад дозоляє вимірювати наступні параметри:

- струм через двополюсник;
- струм короткого замикання;
- фото е.р.с.

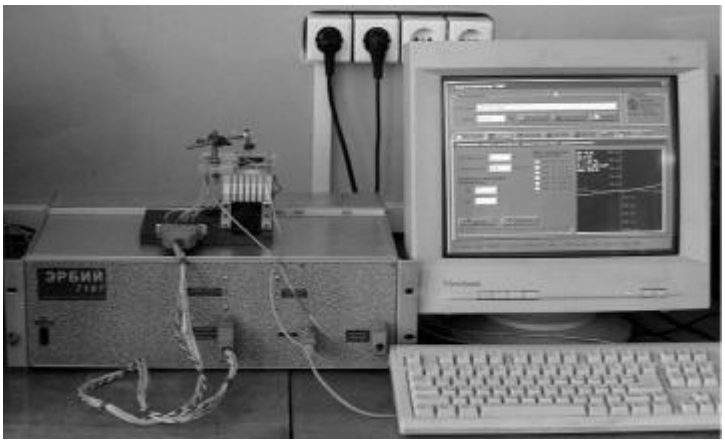

Рисунок 1.6 — Корпус приладу з вимірювальним стендом та персональним комп’ютером

Передбачено перерахунок виміряного фотоструму в потужність випромінювання (мВт).

Прилад має високу точність завдання і вимірювання параметрів. Програмно-апаратний комплекс дозволяє автоматично знімати набір різних характеристик, одержуваних зі змінними, з обраним кроком, параметрами (напруга, освітленість, час).

У приладі реалізована функція захисту досліджуваного зразка від струмового перевантаження. Також прилад і програма можуть оснащуватися додатковими функціями: іншими характеристиками (наприклад, залежностями від температури) та ін.

Основні характеристики приладу:

- Струм живлення зразка (будь-якої полярності): від 10 нА до 200 мА;
- Напруга на виводах зразка (будь-якої полярності): від 100 мкВ до 5 В;

- Температура тримача: в діапазону зоні температур від 0 до +90°C (Або ширше, в залежності від типу нагрівача і охолоджувача): точність підтримки температури: 0,1°C, стабільність 0,01°C, швидкість нагріву і охолодження тримача: 30°C на хвилину, час встановлення температури: близько трьох хвилин від моменту включення струму мікрохолодильників Пельтьє;
- Струм живлення (будь-якої полярності) мікрохолодильників Пельтьє: від 0,1 А до 3 А; напруга: до 15 В;
- Діапазон перестрочування освітленості: 1:10000;
- Точність і стабільність підтримання освітленості: від 0,1% до 1%;
- Точність завдання і вимірювання напруги: від 0,1% до 1,3% від вимірюваної величини;
- Точність вимірювання струму: від 0,3% до 3% від вимірюваної величини.

Програмна частина призначена для керування приладом, вигляд вікна користувацького інтерфейсу показаний на рисунку 1.7.

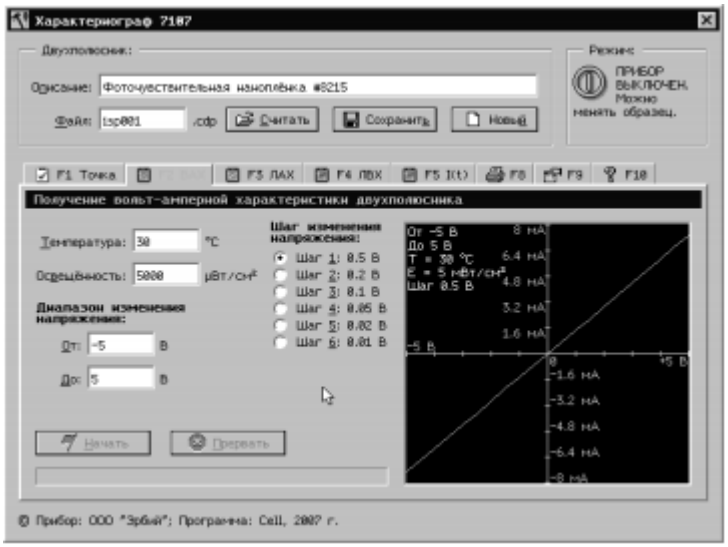

Рисунок 1.7 — Віно програми для керування приладом

Вона забезпечує отримання параметрів зразка «в одній точці» (при фіксованому впливі), а також організовує отримання характеристик за допомогою зручного інтерфейсу.

Отримані програмою характеристики зберігаються в файли, роздруковуються на принтері і передаються в зовнішні програми обробки даних. Інформація про кожного двополюсника (задані параметри і отримані характеристики) зберігається в окремий файл, що дає можливість пізніше відновити роботу з кожним конкретним зразком [4].

Даний прилад є слабким аналогом розроблюваної системи, оскільки він призначений для вимірювання виключно двополюсників. Однак його варто розглядати оскільки він використовує персональний комп'ютер як засіб відображення.

Прилад має значні розміри, що видно з рисунку 1.6, і не має можливості використання тривимірної графіки для відображення отриманих характеристик.

1.5 Характерограф TR-4805

Характерограф типу TR-4805 (рис. 1.8) є приладом загального призначення, що слугує для дослідження характеристик напівпровідникових приладів.

Прилад призначений для наочного дослідження напівпровідникових приладів: транзисторів малої та великої потужності, польових транзисторів, діодів, стабілітронів, тунельних діодів, тиристорів і т.д.

Він дозволяє легко і швидко вимірювати напівпровідникові прилади в лабораторіях та на виробництві.

За допомогою основного приладу можливо виконувати випробування двох елементів за допомогою порівняння при ручному переключенні параметрів в діапазоні 0–1000 В та 5 нА–2 А, при максимальному базовому струмі 200 мА. З використанням високоамперного адаптера типу 1576-3 (TR-4806-3) та високоамперного адаптера типу 1575-3 (TR-4806-3), що поставляються окремо в якості модулів розширення, діапазон вимірювань може бути розширений до 200 А колекторного струму та до 20 А струму бази (в імпульсному режимі).

Прилад також підходить для випробувань пасивних елементів, наприклад реле, перемикачів, роз’ємів при великих та малих струмах, до 200 А.

Розробники заявляють, що характерограф повністю зібраний на напівпровідникових елементах, та що використання інтегральних мікросхем підвищує надійність апарату. Прилад дуже легкий в обслуговування, що сприяє широкому використанню.

Рисунок 1.8 — Передня панель характерографу TR-4805

Основні характеристики приладу:

- Дозволяє вимірюваними одразу два напівпровідникові прилади та порівнювати результати при ручному переключенні;
- Живлення колекторного кола: 0–1 кВ;
- Полярність живлення: позитивна і негативна;
- Режим роботи: змінна напруга (випрямлена синусоїдальна напруга) постійна напруга;

- Максимальний струм колекторного кола: 2 А
- Потужність розсіювання: 0,1–0,5–2–10 Вт з автоматичною індикацією;
- Послідовний опір: 0–1,7 МОм;
- Захист від небезпечної напруги: на напівпровідникові прилади, що затиснуті у вимірювальні затискачі можна подавати напругу лише в закритому стані;
- Режим роботи генератора сходинок: дискретна напруга сходинки, циклічно повторювані сходинки напруги та струму;
- Кількість сходинок: 2–10
- Кроки струму: 0,2 мкА–20 мА;
- Кроки напруги: 0,1 В–2 В;
- Похибка при нульовому зміщенні: ±5%;
- Зміщення: не менше ±1 кроку;
- Перевірка струму заслону польового транзистора: за допомогою послідовного включення опору 100 кОм;
- Режим роботи підсилювача горизонтального відхилення: вимірювання напруги колектор-емітер або напруги база-емітер;
- Коефіцієнт відхилення: 0,1–100 В на поділку;
- Точність підсилювача горизонтального відхилення: ±5%;
- Режим роботи підсилювача вертикального відхилення: вимірювання колекторного струму;
- Коефіцієнт відхилення: 5 нА–0,2 А на поділку;
- Точність підсилювача горизонтального відхилення: ±5% ±10нА;
- Розмір зображення дисплея: 80х80мм (10х10 поділок);
- Положення зображення: звичайне для обох типів транзисторів;
- Необхідна напруга живлення: 100, 127, 200 В ±10% (з можливістю переключення);
- Частота живлення 50/60 Гц;
- Габаритні розміри: 252х262х343 мм;

- Вага 12,5 кг;
- Діапазон робочих температур:+10°C–+35°C [5].

Як видно з наведених характеристик даний прилад найбільш професіональними, оскільки в ньому передбачено декілька режимів вимірювання, можливість встановлення додаткових модулів, наприклад для вимірювання силових транзисторів зі струмами до 200 А. Можливо використання джерел живлення з різною напругою та частотою. Також даний прилад має можливість проводити вимірювання та порівняння характеристик двох транзисторів. Все це робить його більш досконалим ніж більшість перерахованих аналогів.

Згідно документації прилад випущений в 1978 році, а отже технології та елементна база, що використовується для виробництва, давно застаріли і потребують модернізації.

Слід також звернути увагу на значну вагу приладу, що не сприяє його поширеному використанню. Для зменшення ваги можна відмовитись від апаратної обробки та відображення характеристик і використовувати для цього засоби обчислювальної техніки.

1.6 Висновки

В даному розділі розглянуто аналоги різних років випуску та ступеню комерційності, від саморобних схем опублікованих у спеціалізованих виданнях до великих професіональних приладів що потребують промислового живлення. Таке порівняння дозволить вибрати успішні проектні рішенні, що були реалізовані в інших подібних системах, та уникнути помилок допущених іншими розробниками.

Більш старі прилади використовують вбудовані системи відображення результатів вимірювань. Це призводить до підвищення ціни приладів та до ускладнення модернізації системи обробки та візуалізації отриманих характеристик.

Новіші використовують засоби обчислювальної техніки, зокрема і персональні комп’ютери, в якості підсистем обробки результатів та керування вимірюванням.

Однак жоден з розглянутих аналогів не використовує засоби тривимірної графіки для відображення характеристик вимірюваних напівпровідникових приладів. Вони використовують традицій систему відображення характеристик за допомогою сімейств. А більш примітивні взагалі можуть відображати лише одну характеристику за раз.

2 РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ

В основі роботи системи лежить прилад, який дозволить отримувати дані реальних зразків транзисторів в необхідній, для подальшої обробки, формі. Тому даний розділ присвячено розробці схеми апаратної частини системи, вибору схемних рішень та розрахунку параметрів елементної бази.

2.1 Вибір схемних рішень

2.1.1 Вибір мікроконтролера

Яскравими представниками восьми розрядних мікроконтролерів є вироби компаній Motorola (68HC05, 68HC08, 68HC11) і Zilog (Z8).

Motorola довгий час не надавала засобів, які дозволяють дешево і швидко почати роботу з її контролерами, що явно не сприяло їх популярності серед некорпоративних розробників. Однак слід зауважити, що закордоном мікроконтролери від Motorola займають більшість ринку.

Мікроконтролери фірми Zilog, заснованої колишніми працівниками Intel, яка ще недавно виглядала багатообіцяюче, не витримала гонки на ринку і сьогодні система команд Z8 виглядає досить застарілою.

Мікроконтролери PIC можна придбати за досить низькими цінами, що дозолило їм швидко захопити значну частину ринку. До того ж кристали від фірми Microchip виявились не гірші, а часто навіть кращі за більшість аналогів за продуктивністю і не потребують дорогого програматора.

Також у продажу можна знайти дешеві комплекти PICSTART, які містять все необхідне, для того щоб не маючи ні засобів, ні навиків роботи з PIC-контролерами, швидко створити та налагодити на ньому продукт.

Ці контролери мають хороші порти, а все інше зроблено досить незручно. Архітектура залишає бажати кращого, система команд виявилась вкрай обмеженою. Тим не менш PIC-контролери залишаються популярними в тих випадках коли потрібно розробити недорогу систему, яка не пред’являє серйозних вимог до її керування.

Контролери фірми Scinex мають 52 команди (PIC має 33), інструкції для роботи з пам’яттю, покращена архітектура, кожна команда виконується за один машинний цикл, що при інших рівних умовах швидше ніж у Microchip, до того ж їх частота досягає 100 МГц.

Така висока швидкість контролера дозволила його розробникам відмовитись від периферії — таймерів, лічильників, регістрів зсуву в портах вводу–виводу, — це все рекомендується реалізовувати чисто програмними засобами. А з апаратних контролер має лише швидке ядро, пам’ять та порти вводу/виводу.

Корпорація Atmel в 1996 році спричинила справжню революцію у світі мікроконтролерів, представивши своє сімейство чипів на новому прогресивному ядрі AVR. Більш продумана архітектура AVR, швидкодія яких перевищувала контролери Microchip, приваблива цінова політика сприяла відтоку багатьох розробників від найбільш популярних, на той час, PIC-контролерів.

Мікроконтролери AVR мають більш розвинену систему команд, що налічує до 133 інструкцій, продуктивність, близьку до 1 MIPS/МГц, Flash ПЗУ програм з можливістю внутрішньо схемного перепрограмування. Багато чипів мають функцію само програмування. AVR-архітектура оптимізована під мову високого рівня Сі. Крім того всі кристали підтримують сумісність «знизу в гору».

Величезну роль зіграла доступність програмного забезпечення і засобів підтримки розробки. У Atmel багато програмних продуктів, що розповсюджуються безкоштовно. Добре відомо, що розвинуті засоби підтримки розробки при освоєнні та ознайомленні з будь-яким сімейством мікроконтролерів відіграють не менш значну роль ніж самі кристали. Фірма Atmel приділяє багато уваги цьому питанню. Також популярності сприяє наявність надзвичайно вдалого, безкоштовного, середовища розробки AVR Studio для ОС Windows.

Для розробника-початківця важливим є той факт, що для програмування AVR можна обійтись взагалі без програматора. Найбільш популярним способом програмування цих контролерів є безпосереднє підключення до паралельного порту персонального комп'ютера [6].

2.1.2 Вибір способу з'днання з ПК

Оскільки обробка даних повинна відбуватись на персональному комп'ютері. Для цього в схемі повинен бути передбачений спосіб передачі інформації на ПК.

Один з найпростіших способів це передача через COM або LPT порти. Пересилати дані можна на пряму — без використання інтегрованих схем. Але вони майже вийшли з вжитку и практично не зустрічаються на сучасних ПК.

Альтернативою є USB-порт, який є одним із найбільш уживаних способів обміну даними з ПК. Однак протокол передачі дуже складний, що може сильно загальмувати проектування. Тому більшість розробників використовує готові рішення у вигляді інтегральних мікросхем. Таким чином для з'єднання з ПК можна використати один з наступних варіантів.

Proiffilic Tehnologi PL2303X — цей вузол є конвертором USB/послідовний порт для USB 1.1. Дескриптори вмонтовані у внутрішні ПЗУ (ROM — Read Only Memory). Для власного PID або ID вендора, знадобиться зовнішній EEProm, який може бути підключений за допомогою дводротової лінії. Додатки обмежені сигнальними лініями послідовного інтерфейсу. Версія PL2303HX RevD має OTR-ROM (One time Programmable) і дозволяє використання послідовних сигналів в якості 8 біт вводу/виводу.

Silicon Labs CP2102/103 — на противагу Proiffilic має інтегрований EEProm с 1024 байтами та підтримує USB 2.0 Full Speed.

MOSCHIP semiconductor MSC7820 підтримує два послідовних інтерфейси та передачу даних SIR-IrDA (інфрачервоний), а також USB 2.0. Також є можливість підключати зовнішній EEProm за допомогою двох інтерфейсів I2C.

Фірма **maxim** пропонує USB мікросхеми **Max3421** і **Max342**, які мають два прямих з'єднання мікроконтроллера з перетворювачем USB 2.0 (Full Speed). Крім USB має також SPI та вісім інших входів і виходів. Мікросхеми підготовлені в якості USB-хоста для USB-OTG. Оскільки послідовний інтерфейс ПК не підтримується, то мікросхеми можуть працювати без додаткового драйвера, через вбудований USB-драйвер Windows.

FTDI FT232R — мікросхема дозволяє проводити емуляцію послідовних і паралельних інтерфейсів (рис. 2.1). Так званий режим Big Bang Mode, який робить мікросхему досить цікавою для створення побутової апаратури. FT232R відкриває можливості для перетворення з USB в інші послідовні інтерфейси в так званій області MPSSE (Multi-Protocol Synchronous Serial Engine). За допомогою режиму Big Bang мікросхема FT232R дає доступ до ПК без використання додаткового контролера [7].

Рисунок 2.1 — Зовнішній вигляд FT232R

Перевагою використання мікросхеми FT232R є також достатня кількість літератури, та демонстраційних прикладів реалізацій на її основі.

В комплект входять драйвери для найбільш популярних операційних систем. Драйвер дає API (Application Programming Interface), що дозволяють створювати програмне забезпечення для кінцевого продукту з використанням мов програмування високого рівня (наприклад Visual Basic).

До недоліків використання окремої мікросхеми в ролі USB-драйвера можна віднести необхідність використання мікроконтролера, а також додаткова інтегральна мікросхема суттєво підвищує вартість виготовлення виробу.

Вбудований модуль USB PIC18. Фірма Microchip випускає контролери з модулем USB, який дозволяє проводити обмін інформацією з персональним комп'ютером (рис. 2.2). Мікроконтроллер PIC18F4550 підтримує обмін даними через протокол USB 2.0 в режимі Full Speed [8].

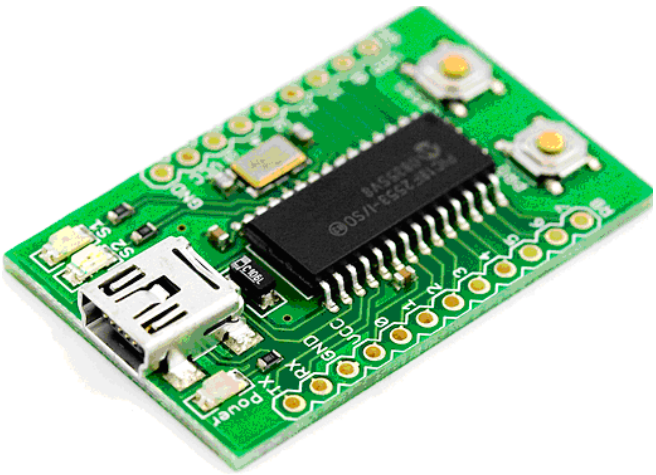

Рисунок 2.2 — Зовнішній вигляд USB PIC18

Превагою використання саме цього рішення є використання однієї мікросхеми для передавання і оброблення даних. Це спростить схему, що призведе до підвищення надійності і зниження вартості приладу. Microchip дає відкритий доступ до USB драйверів для операційної системи Windows. Драйвер верхнього рівня дає розробнику гранично простий API, що сильно спрощує написання прикладних програм для взаємодії з контролером.

До недоліків можна віднести відсутність перекладеної документації, особливо українською мовою. Такий мікроконтролер є досить потужним (а отже дорогим) і його ресурси не використовуватимуться повністю.

2.1.3 Вибір схеми керування напругою

Для побудови характеристик транзистора необхідно змінювати напругу в широких межах: від нуля до максимально допустимого, для даного

транзистора, значення. Для більшості транзисторів це значення не перевищує 50 В. Отже схема повинна регулювати напругу в діапазоні 0..50 В.

Найбільш простим і логічним способом є використання цифро-аналогового перетворювача (ЦАП). Мікроконтролер PIC18F4550 не має вбудованого, але має достатньо портів вводу-виводу для під'єднання зовнішнього ЦАП.

Мікросхеми ЦАП, як правило мають більше декілька каналів та невисоку максимальну вихідну напругу. Знайти мікросхему, що задовольняла б усім вимогам (максимальна напругою 50 В, одно канальний) не вдалося. А найкращі знайдені результати були занадто дорогі для використання — сильно підвищували вартість виробу. Тому довелося відмовитись від використання мікросхеми ЦАП.

Замість ЦАП можна використовувати конвертори постійної напруги з керуванням за допомогою широтно-імпульсної модуляції (ШІМ).

Для живлення всієї схеми можна скористатись блоком живлення з ПК, який дозволяє отримати напругу 12 В та струм в декілька ампер. Тому спочатку була розглянута можливість використання конвертора з підвищенням напруги. Основною превагою якого є те, що для приладу не потрібно буде використовувати окреме джерело живлення.

Інтегральна схема MAX16824 — високовольтний інтегрований драйвер світлодіодів високої яскравості с діапазоном вхідної напруги від +6.5 В до +28 В. Особливістю ІС є наявність трьох виходів постійного струму з відкритим витоком, до якого підключається джерело напруги номіналом 36 В та номінальним вихідним струмом 150 мА, для кожної з трьох окремих ліній світло діодів. Струм можна задавати незалежно, за допомогою зовнішніх резисторів. Особливістю MAX16824 є наявність трьох ШІМ, виводи яких керують світлодіодами. Виводи ШІМ також використовуються для вмикання і вимикання кожної лінійки діодів (рис. 2.3).

Регулювати напругу можна безпосередньо за допомогою мікроконтролера, використовуючи для цього чотирьох розрядний послідовний інтерфейс з пропускною здатністю 2 Мбіт/с [9].

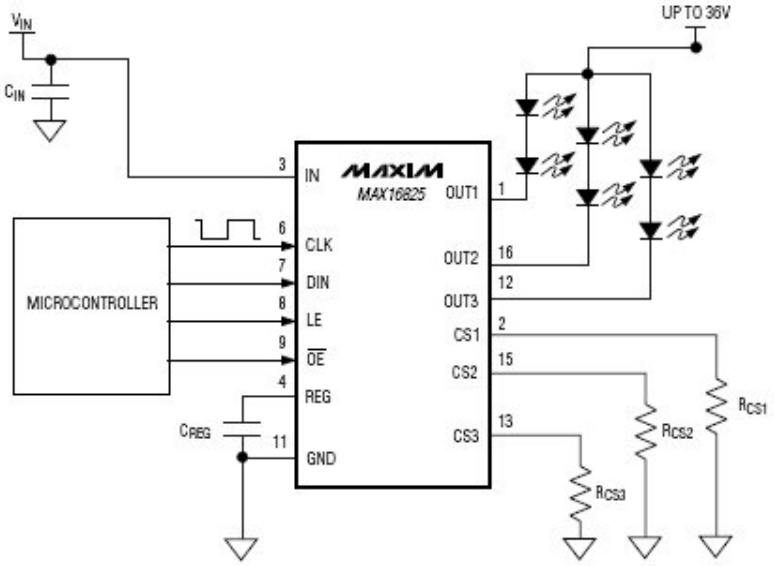

Рисунок 2.3 — Схема ввімкнення MAX16824

Перевагою використання мікросхеми є простота роботи з нею. А недоліки — її вартість і два непотрібні канали.

Наступним рішенням було зібрати конвертор на дискретних елементах, оскільки мікроконтролер PIC18F4550 має вбудований ШІМ.

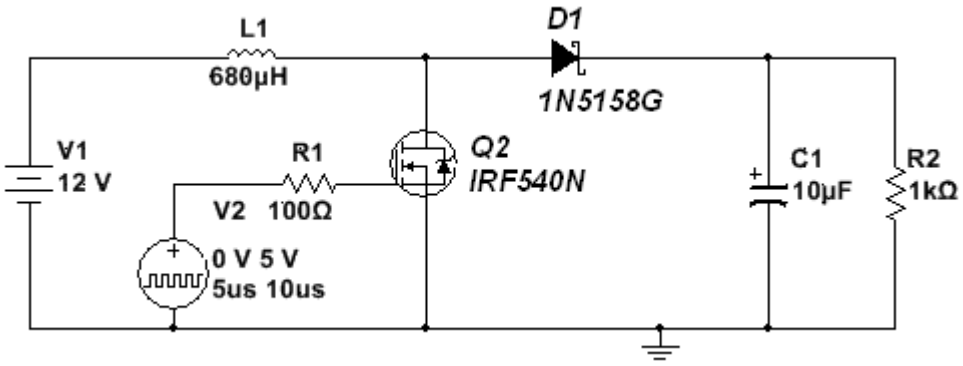

Рисунок 2.4 — DC/DC Step-up convertor

Схема зображена на рис. 2.4 є найбільш простим і дешевим рішенням та видає напругу від 12 до 40 вольт. Але підвищувальний конвертор не може дати напругу меншу ніж на його вході, але для дослідження транзистора, характеристика при напрузі колектор-емітер в діапазоні від нуля до кількох

вольт є найбільш цікавою. Отже в схемі даного приладу підвищувальний конвертор постійної напруги не може бути використаний [10].

Конвертора понижуючого типу на відміну від підвищувального може забезпечити напругу від нуля (фактично забезпечує від сотень мілівольт). Проте використання такого конвертора потребує окремого джерела живлення з вищою напругою.

Інтегральні схеми понижуючого конвертора через значну вартість не розглядалися.

В схемі на рис. 2.5. транзистор є ключем, для створення імпульсної напруги з постійної. При цьому амплітуда сформованих імпульсі рівна величині вхідної напруги. Для підвищення ефективності перетворення транзистор повинен перемикатись з високою частотою (чим вище частота тим ефективніше перемикання). В реальних схемах частота перемикання транзисторів може знаходитись в діапазоні від 80 кГц до 2 МГц.

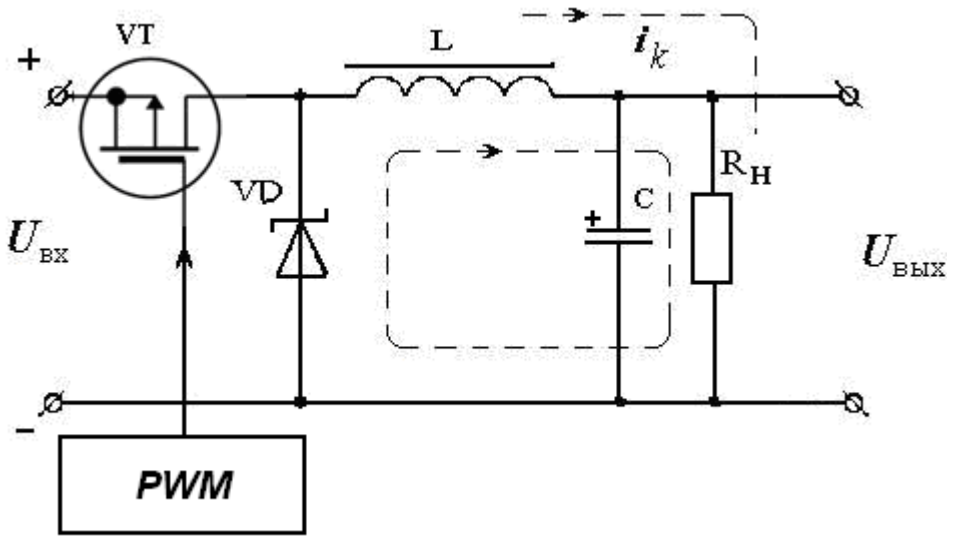

Рисунок 2.5 — DC/DC step-down converter

Далі, отримана імпульсна напруга згладжується дроселем і електролітичним конденсатором. В результаті на виході створюється постійна напруга, але меншої величини. При цьому величина вихідної наруги буде пропорційна ширині імпульсів отриманих на виході транзистора. Якщо він відкриється на більший проміжок часу то енергія яку накопичує дросель

збільшиться, що, в свою чергу, приведе до збільшення напруги на конденсаторі. І навпаки — при меншій тривалості відкритого стану транзистора напруга на конденсаторі зменшиться [11].

Важливим елементом схеми є діод. За його допомогою підтримується стум навантаження, що створений дроселем, в ті періоди часу коли транзистор закритий. Інакше кажучи коли транзистор відкритий струм дроселя і струм навантаження забезпечуються джерелом живлення, а дросель при цьому накопичує енергію. Після закриття транзистора струм навантаження підтримується за рахунок енергії, яку накопичив дросель. Цей струм протікає через діод, тобто енергія дроселя витрачається на підтримання струму навантаження.

Збільшити ККД можна якщо замість діода використовувати синхронний випрямляч (рис. 2.6), тобто замість діода знизу встановлюється такий самий польовий транзистор, він працює синхронно з першим але вони протифазні, тобто відкривається коли верхній польовий транзистор закритий і навпаки. Проте схема синхронізації двох силових транзисторів сильно ускладнюється тому для керування нею краще вирисовувати спеціальний синхронний інтегрований драйвер. Він має всі необхідні затримки між відкриттям і закриттям, щоб зменшити протікання небажаних струмів. Але додавання до схеми рішення ще однієї інтегральної схеми призведе до підвищення витрат на виробництво тому буде використана схема з одним польовим транзистором і діодом.

Однак така схема ввімкнення польового транзистора не буде працювати. Річ у тім що N-канальний польовий транзистор повністю відкриється якщо на заслін подати позитивну напругу відносно витоку.

В даному випадку якщо на заслін подати позитивну напруги то транзистор почне відкриватись і напруга на витоку також підніметься. В результаті транзистор не може бути повністю відкритий або закритий.

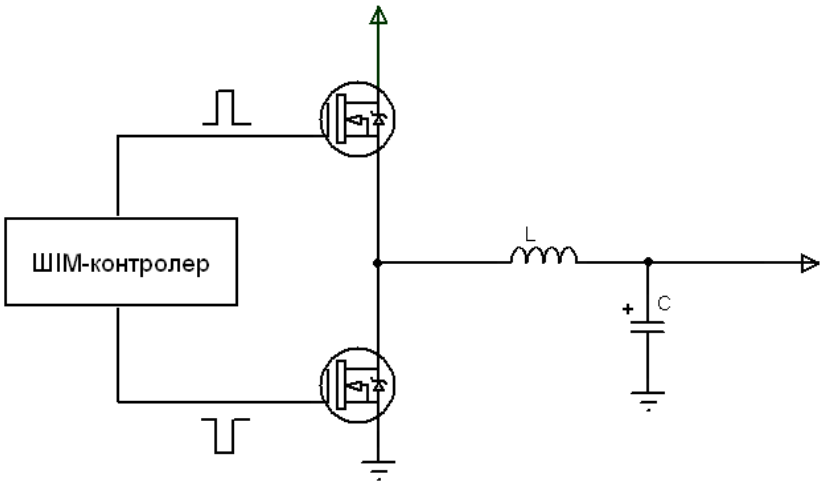

Рисунок 2.6 — Схема конвертора з синхронним випрямлячем.

Цю проблему можна вирішити за допомогою нескладної схеми, зображеної на рис. 2.7. Як видно силовим польовими транзистором (з права) керує біполярний драйвер. Однак він заведений відносно його Витоку. Транзистор що знаходиться зліва — малопотужний, він використовується для зсуву рівня. Сигнал на таку схему подається інвертований. Резистор Pull-Down потрібно поставити щоб схема не «висіла в повітрі». Принцип роботи схеми наступний: спочатку конденсатор CBOOT заряджається через діод DBOOT, напругою керування, оскільки транзистор закритий, на виході Витоку земля (після дроселя L іде навантаження, що заземляє Витік на час ввімкнення транзистора). Польовий транзистор зсуву рівня (зліва) навпаки, відкритий, щоб силовий польовий транзистор був закритий. Власне, це і є інверсія. Коли транзистор зсуву рівня закривається через резистор RLEVEL подається позитивна напруга на драйвер, який підсилює сигнал і подає «+» на Заслін силового транзистора. Він починає відкриватись і через деякий час відкривається повністю. Так як конденсатор CBOOT заряджений і прив’язаний до Витоку силового транзистора, то коли напруга на ньому зрівняються з напругою живлення то конденсатор CBOOT починає керувати силовим транзистором через драйвер. Виходить, що напруга в момент відкриття силового польового транзистора відносно землі дорівнює сумі напруги живлення та напруги на конденсаторі CBOOT. А діод не дозволяє напрузі виходити назад. Тому важливо знати різницю напруг і

використовувати діод з запасом на потрібну напругу. Коли конденсатор CBOOT розряджається то починає відкриватись транзистор зсуву рівня, на драйвері починає падати напруга і одночасно з цим Виток силового транзистора знову замикається на землю [12].

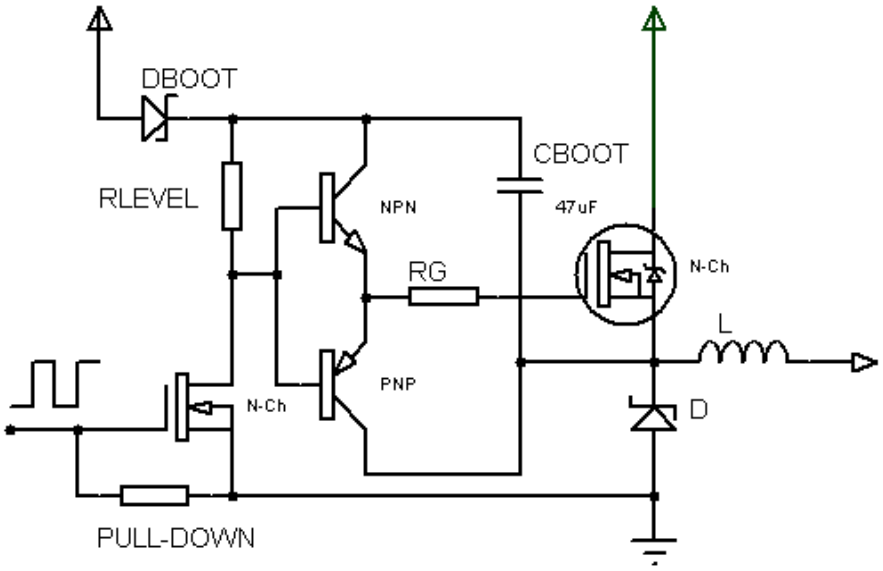

Рисунок 2.7 — Схема керування напругою

Можна також додати до схеми резистор після **Стоку** керуючого польового транзистора. Це потрібно для того, щоб захистити транзистор рівня коли драйвер відкритий і його «земля» має більший потенціал ніж реальна.

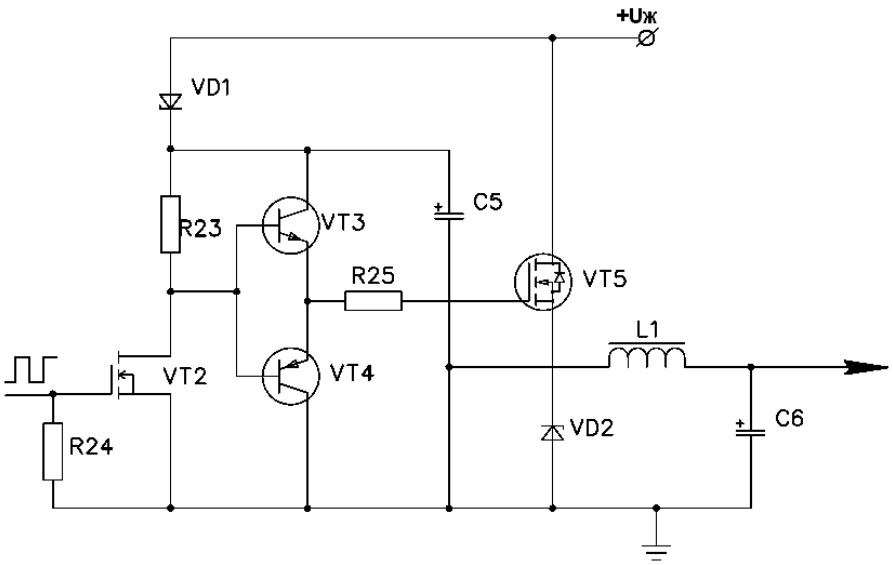

Рисунок 2.8 — Остаточний варіант схеми

В результаті було створено схему (рис. 2.8), працездатність якої підтверджено за допомогою моделювання в середовищі Multisim.

2.1.4 Вибір керованого джерела струму

В якості керованого джерела струму було обрано набір резисторів, що під’єднанні безпосередньо до виводів мікроконтролера. Хоча напруга на всіх виводах буде однаковою, але за рахунок різного опору буде протікати різний струм.

Для більш гнучкого регулювання струму можна вмикати декілька портів тоді опір буде визначатись паралельним з’єднанням резисторів, що підключені до цих виводів. Відповідно буде змінюватись і струм.

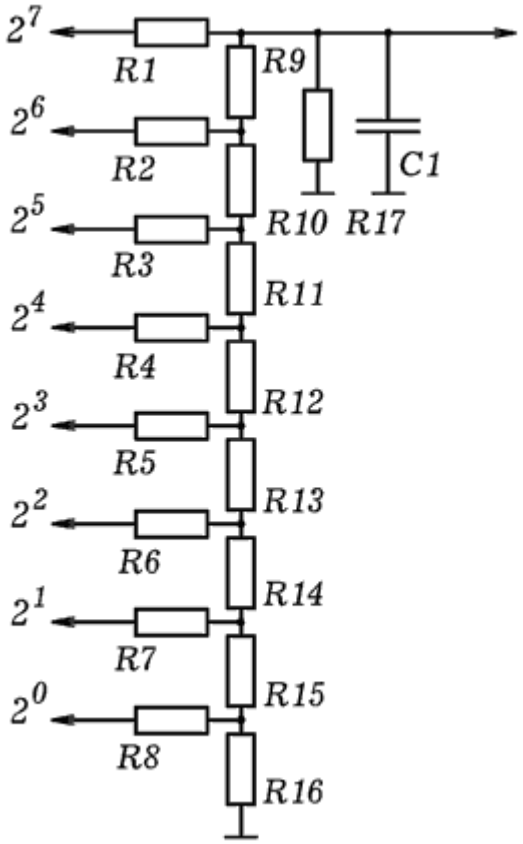

Рисунок 2.9 — R-2R-ЦАП

Але оскільки в даному випадку не вимагається висока точність перетворення то можна обійтись дешевим способом, без використання спеціальних мікросхем. На рис. 2.9 зображена схем 8-ми бітного ЦАП з використанням матриці резисторів R–2R. Основний недолік такої схеми в

тому що потрібно 8 портів вводу-виводу. Похибка такого перетворення складає один молодший розряд.

Тут використовується тільки два номінали опорів, що відносяться як 2:1. Кількість резисторів рівна 2N. Резистивний дільник можна використовувати в якості ЦАП двома способами, в режимі напруги і в режимі струму (нормальний та інверсний режим). Головна перевага ЦАП з виходом за напругою в тому що вихідний імпеданс такої схеми залишається постійним. Друга перевага — відсутність ємнісних струмів у навантаженні. До недоліків схеми можна віднести те що джерело повинно мати низький імпеданс. По-друге для регулювання підсилення не можна використовувати резистор послідовно з джерелом [13].

Найбільша перевага для використання такої схем в тому що резистори мають однаковий номінал, що сильно спрощує його проектування.

2.2 Аналіз структурної та принципової схеми

2.2.1 Аналіз структурної схеми

Структурна схема системи зображена на рис. 2.10.

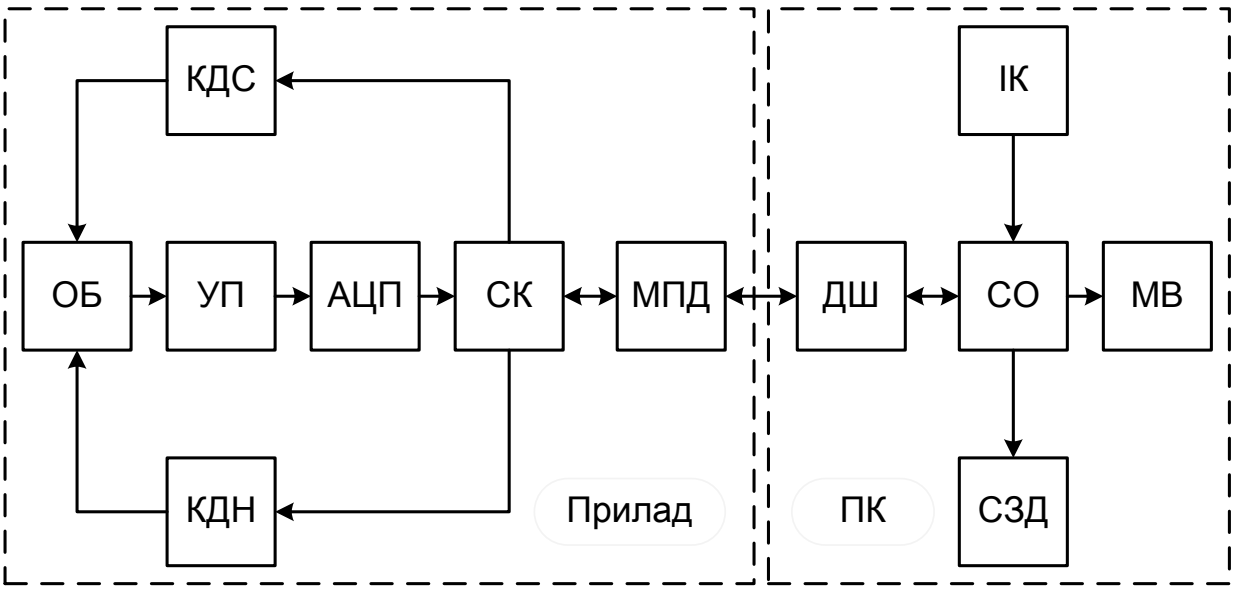

Рисунок 2.10 — Структурна схема системи

Для вимірювання воль-амперних характеристик транзистора, який позначено як об’єкт вимірювання (ОВ) необхідно змінювати стум бази та

напругу колектор-емітер, для цього в схемі передбачене кероване джерело струму (КДС) та кероване джерело напруги (КДН).

Отримане значення напруги через пристрій узгодження (УП) подається на аналогово-цифровий перетворювач (АЦП).

Схема керування (СК) визначає напруги колектор-емітер і струму бази транзистора. Також проводить первинну обробку даних перед відправкою на ПК. Отримані дані, за допомогою модуля передачі даних (МПД) пересилаються на персональній комп'ютер для подальшої оброки.

Драйвер шини (ДШ) — низькорівневий програмний модуль призначений для взаємодії з апаратною частиною системи.

Отримані дані повинні піддаватись обробці та аналізу, за допомогою системи обробки (СО). Вона проводить перевірку та, в разі необхідності, корегування результатів. В основі такого аналізу лежить співставлення отриманих даних з математичною моделлю.

Модуль відображення (МВ) виконує власне побудову тривимірного зображення. Інтерфейс користувача (ІК) призначений для керування системою.

Згідно структурної схеми після проведення огляду існуючих рішень створено схему електричну принципову.

2.2.2 Аналіз електричної принципової схеми

Для вимірювання воль-амперних характеристик транзистора необхідно змінювати стум бази та напругу колектор–емітер. Це реалізовано за допомогою ЦАП та керованого джерела напруги.

Цифро-аналоговий перетворювач підключається до восьми розрядного порту-В мікроконтролера, виводів RB0–RB7. Кожен із цих виводів може давати напругу 5 В, якщо він знаходиться в стані логічної 1, або 0 В в стані логічного 0. Тобто, в залежності двійкового від коду, що встановлений на виводах порту ЦАП може задавати напругу від 0...5 В, з кроком 0,02 В. Похибка такого перетворювача складаю один молодший розряд — 0,02 В.

Резистор R18 необхідний, щоб задати струм бази. Його опір було розраховано виходячи з того, що для більшості малопотужних транзисторів складає приблизно від 100 мкА до 10 мА.

В схемі також передбачений зворотній зв'язок, який дозволяє точніше задавати базовий струм транзистора. Напруга бази знімається за допомогою вбудованого аналогово-цифрового перетворювача мікроконтролера, на виводі AN2, через дільник утворений резистором R24 та вхідним опором АЦП.

Кероване джерело напруги складається з польового транзистора VT4, діода VD2, конденсатора C6 та індуктивності L1. Керування джерелом напруги відбувається за допомогою зміни шаруватості імпульсів які подає мікроконтролер на заслін силового транзистора, який працює в режимі ключа. Транзистор в свою чергу відкриваючись і закриваючись на відповідні періоди часу створює з постійної напруги імпульсну, амплітуда якої приблизно рівна величині напруги живлення. Далі імпульсна напруга згладжується за допомогою дроселя та конденсатора і подається на схему вимірювань.

Транзистори VT1–VT3 необхідні для того, щоб правильно задавати режим роботи потужного ключа.

Для отримання точного значення напруги, що подається на схему вимірювань реалізовано зворотній зв’язок утворений дільником напруги, що складається зі 100 кОм резистора R22 та 10 кОм вхідного опору вбудованого АЦП. Як видно цей дільник має коефіцієнт ділення 1/10, тобто зменшує максимально можливу напругу з 50 В до необхідних 5 В.

Режим роботи вимірюваного транзистора задається резистором R26, який обмежує струм колектора коли транзистор відкритий. А R23 є частиною дільника, що вимірює напругу колектор–емітер.

Відображення режиму роботи комплексу відбувається за допомогою світлодіодів HL1 та HL2. Переший відображає поточний стан вимірювань і

керується безпосередньо мікроконтролером. Він може знаходитись в одному з трьох станів:

- викинений — з'єднання з ПК не встановлено;
- ввімкнений — прилад готовий до роботи;
- блимає — запущено процес вимірювань.

Другий світлодіод контролює наявність напруги +60 В від зовнішнього джерела живлення, тобто світиться протягом всього часу роботи. Кожен світлодіод має послідовно ввімкнений резистор, що забезпечує коректний режим його роботи.

В схемі реалізоване апаратне скидання лічильника команд мікроконтролера при підключенні до ПК, оскільки він живиться від USB порту комп'ютера. За цю функцію відповідають резистор R27 і конденсатор C7.

Функції схеми курування та частково модуля передачі даних виконує мікроконтролер. Варіанти схем підключення та фрагменти коду прошивки для роботи з протоколом USB можна знайти в документації [14].

2.2.3 Алгоритм роботи мікроконторлера

Мікроконтролер отримує команди від ПК, виконує первинну обробку даних та керує процесом вимірювань.

Вимірювання починається після отримання команди з ПК. Перш за все мікроконтролер встановлює заданий з ПК струм бази транзистора за допомогою ЦАП. Далі він знімає значення напруги база-емітер $U_{Б}$ на виводі AN2 та вираховує струм за формулою:

$$I_{Б} = \frac{U_{БЕ} - U_{Б}}{R18},$$

де $U_{БЕ}$ — наруга між базою і емітером, яка знімаються за допомогою АЦП на виводі AN1.

Якщо розраховане значення струму відрізняється від заданого то проводиться корегування та знову проводиться вимірювання. Після

встановлення необхідного значення, струм бази залишається незмінним протягом всього циклу побудови характеристики.

Далі мікроконтролер починає змінювати напругу на схемі вимірювання за допомогою вбудованого модуля ШІМ на виводі CCP1. При цьому контролер може контролювати цю напругу за допомогою зворотного зв'язку. Після завершення перехідних процесів він за допомогою АЦП реєструє напругу на виводі AN1 та розраховує струм колектора за такою формулою:

$$I_K = \frac{U_{AN0} - U_{AN1}}{R26}.$$

Потім контролер збільшує напругу на заданий крок і процес повторюється знову, доки напруга колектор-емітер не досягне заданого значення. Величина кроку зміни визначається автоматично за крутістю характеристики.

Для уникнення простою під час очікування завершення перехідних процесів, мікроконтролер в цей час передає отримані дані на персональний комп'ютер. Дані для кожної точки складаються з пари чисел подвійної точності з плаваючою комою: наруга колектор-емітер у вольтах та струм колектора у міліамперах. В свою чергу програмне забезпечення ПК за цими даними будує характеристику та відображає її на екрані.

Увесь процес вимірювання триває не більше 20 секунд, після чого можна зберегти результат та проводити нові вимірювання.

2.3 Розрахунки параметрів схеми

Розроблюваний прилад призначений для вимірювань тому для отримання точніших результатів значення параметрів деяких елементів потрібно точно розрахувати.

Коли вимірюваний транзистор відкритий повністю відкритий то його опором можна знехтувати. В цьому разі вся напруга буде падати на резисторі R26. Максимальний струм через транзистор не повинен перевищувати 100 мА. Тоді можна знайти опір резистора:

$$R26 = \frac{U_{ж}}{I} = \frac{50}{0{,}1} = 500 \text{ Ом}.$$

А потужність, що розсіюється на резисторі:

$$P_{R26} = \frac{U_{ж}^{\ 2}}{R26} = \frac{50^2}{500} = 5 \text{ Вт}.$$

Для отримання значень напруги мікроконтролер має вбудовані аналогово-цифрові перетворювачі, які здатні реєструвати від 0 В до 5 В. Оскільки максимально напруга 50 В то потрібно використовувати дільник напруги з коефіцієнтом ділення 1/10. Такий дільник може бути утворений резистором R22 та вхідним опором АЦП контролера. Згідно з документацією на обраний мікроконтролер вхідний опір його АЦП дорівнює 10 кОм.

$$\frac{1}{10} = \frac{R_{\text{АЦП}}}{R22 + R_{\text{АЦП}}},$$

$$R22 = 10R_{\text{АЦП}} - R_{\text{АЦП}} = 9R_{\text{АЦП}} = 9 \cdot 10 = 90 \text{ кОм}.$$

Максимальна потужність на резисторі:

$$P_{R22} = \frac{U_{ж}^{\ 2}}{R22} = \frac{50^2}{100 \cdot 10^3} = 0{,}025 \text{ Вт}.$$

Оскільки резистори R23 та R24 мають однакове функціональне призначення з резистором R22 то наведені для нього розрахунки справедливі і для них.

Задавати напругу, що визначає струм бази вимірюваного транзистора, можна за допомогою цифро-аналогового перетворювача зібраного на резисторах R1–R17. Він дозволяє регулювати напругу від 0 В до напруги на виводах мікроконтролера, що дорівнює 5 В, з кроком:

$$\Delta U = \frac{U}{2^8} = \frac{5}{256} = 0{,}02 \text{ В}.$$

Для роботи малопотужних транзисторів потрібно щоб струм бази складав порядку сотень мікроампер. Тоді можна розрахувати значення опору R18, для забезпечення мінімального значення струму 100 мкА:

$$R18 = \frac{U}{I} = \frac{0,02}{0,1 \cdot 10^{-3}} = 200\,\text{Ом}.$$

Максимальне значення струму бази досліджуваного транзистора при розрахованому значенні опору:

$$I_{\text{Вамх}} = \frac{U}{R18} = \frac{5}{200} = 0,025\text{А} = 25\,\text{мА}.$$

Потужність резистора R18:

$$P_{R18} = \frac{U^2}{R18} = \frac{5^2}{200} = 0,125\,\text{Вт}.$$

Для забезпечення необхідного струму (30 мА) через індикаційні світлодіоди послідовно з ними вмикають резистори. Світлодіод HL1 потрібен для відображення сигналів мікроконтролера і буде підключений до його виводу, напруга на якому приблизно дорівнює 5 В. Звідки можна вирахувати необхідне значення опору резистора R19:

$$R19 = \frac{U}{I_{св}} = \frac{5}{0,03} \approx 160\,\text{Ом}.$$

Потужність цього резистора:

$$P_{R19} = \frac{U^2}{R19} = \frac{5^2}{160} \approx 0,15\,\text{Вт}.$$

Світлодіод HL2 призначений для індикації наявності живлення на схемі керованого джерела напруги. Тому вмикається безпосередньо до входу живлення, напруга на якому рівна 50 В, отже опір резистора R28 буде:

$$R28 = \frac{U_{ж}}{I_{св}} = \frac{50}{0,03} \approx 1,6\,\text{кОм}.$$

Потужність резистора:

$$P_{R28} = \frac{U_{ж}^{\ 2}}{R28} = \frac{50^2}{1600} = 1{,}56 \text{ Вт}.$$

Інші елементи є частинами готових схемних рішень, які використовуються в якості функціональних елементів приладу.

2.4 Вибір елементної бази

2.4.1 Вибір резисторів

Розміри резисторів залежать від потужності, яку вони здатні розсіяти. Найпоширенішими є виконання на потужності 0,25 Вт, 0,5 Вт, 1 Вт, 2 Вт, 5 Вт. Також резистори відрізняються допусками: ±10%, ±5%, ±1%. Доцільно обрати резистори фірми Hitano, зовнішній вигляд та габаритні розміри зображено на рис. 2.11.

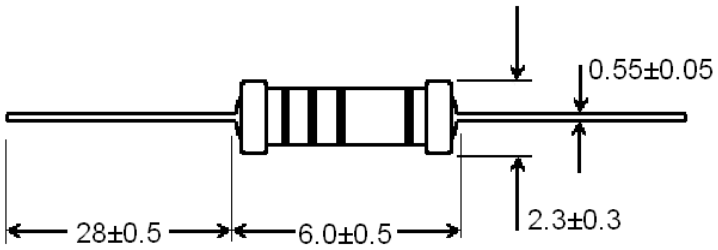

Рисунок 2.11 — Малопотужні резистори

Резистори R1–R8 мають номінальне значення опору 10 кОм. Вони входять до складу цифро-аналогового перетворювача, що задає струм бази, точність значення якого залежить від допуску резистора. Однак значення струму може корегуватись за допомогою зворотних зв’язків тому доцільно обрати ці резистори з досить великим допуском ±5%.

Резистори R9–R17, 20 кОм також є складовими ЦАП тому для них було обрано такі самі параметри.

Резистори R18 та R19 не потребують великої потужності і точності. Перший задає струм бази, який за звичай дуже малий, а другий обмежує струм індикаторний через світло діод.

Резистори R20–R24 використовуються для вимірювань напруги у відповідних точка, в якості складових для дільників. Оскільки фактичне значення опору кожного з них може бути занесене в конфігурацію то вони

також не потребують високої точності. Резистор R26 забезпечує рижим роботи вимірюваного транзистора. Якщо останній відкритий то резистор розсіюватиме значну потужність, яка відповідно до розрахунків становить приблизно 5 Вт. Його вигляд зображено на рис. 2.12.

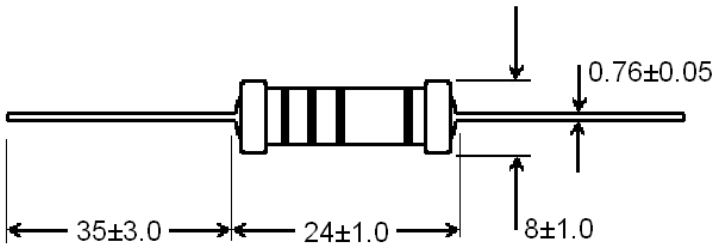

Рисунок 2.12 — Потужний резистор

До таблиці 2.1 зведено всі параметри обраних резисторів.

Таблиця 2.1 — Параметри резисторів

Позиційне позначення	Опір, кОм	Потужність, Вт	Допуск	Назва
R1–R8, R27	10	0,5	±5%	CR025SJTB-10K
R9–R17	20	0,5	±5%	CR025SJTB-20K
R18	0,2	0,5	±5%	CR025SJTB-200R
R19	0,16	0,5	±5%	CR025SJTB-160R
R20– R24	100	0,5	±5%	CR025SJTB-100K
R25	0,1	5	±5%	MOR500JTB-100R
R26	0,5	5	±5%	MOR500JTB-100R
R28	1,6	0,5	±5%	CR025SJTB-1R6

2.4.1 Вибір конденсаторів

Розміри та форма конденсаторів більшості випадків залежить від номінального значення ємності та напруги, яку вони можуть витримати. Найдоступнішими є конденсатори фірми Hitano. На рисунках 2.13 і 2.14 зображено вигляд звичайних та електролітичних конденсаторів відповідно.

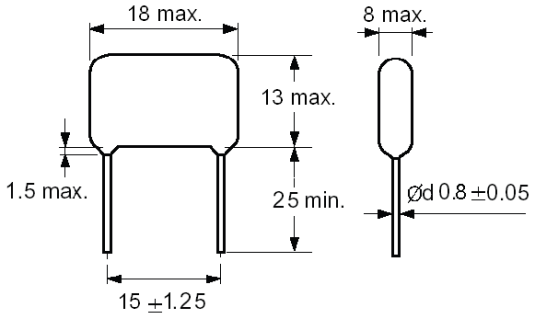

Рисунок 2.13 — Конденсатор фірми Hitano

Більшість конденсаторів даної схеми підключені до сигнальних кіл, напруги 5 В буде цілком достатньо.

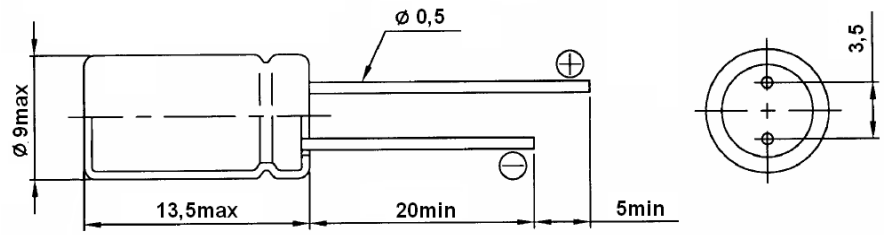

Рисунок 2.14 — Електролітичний конденсатор фірми Hitano

Конденсатори С5 та С6 заходяться в колах живлення, тому їх напруга повинна бути більшою за 50 В. До того ж ці конденсатори мають велику ємність 47 мкФ, тому їх доцільно обрати електролітичними.

В таблиці 2.2 наведено параметри обраних конденсаторів.

Таблиця 2.2 — Параметри конденсаторів

Позиційне позначення	Ємність, пФ	Напруга, В	Допуск	Назва
C1	$22 \cdot 10^3$	16	±20%	*MMK223J63*
C2, C3	22	16	±20%	*TCH2A220K-L515B*
C4	$100 \cdot 10^3$	16	±20%	*SF1E224Z-L215B*
C5, C6	$47 \cdot 10^6$	63	±20%	*ECR470M2AB*
C7, C8	$100 \cdot 10^3$	16	±20%	*R15W104K1HL2-L*

2.4.2 Вибір дроселя

Дросель L1 згладжує імпульсну напругу після силового транзистора. Згідно розрахунків індуктивність дроселя повинна бути близькою до

200 мкГн. Найближче значення 220 мкГн. До того ж дросель повинен бути розрахований на струм до 0,2 А. Тому було обрано RCH664NP-221K фірми Sumida, його вигляд зображено на рисунку 2.15.

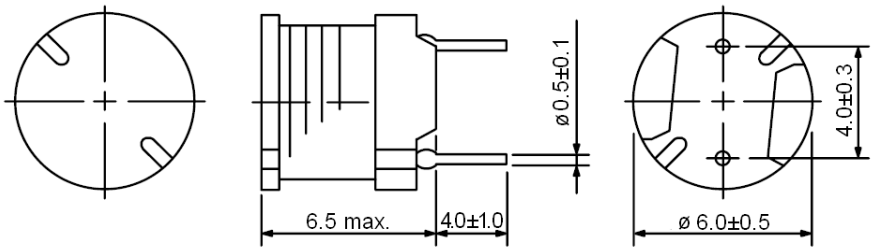

Рисунок 2.15 — Дросель RCH664NP-221K

2.4.3 Вибір резонатора

Для більшої стабільності тактової частоти, що потрібна для підключення USB потрібно використовувати зовнішній резонатор, який повинен видавати частоту 20 МГц. Для цього можна використати кварцовий резонатор HC49S (рис. 2.16).

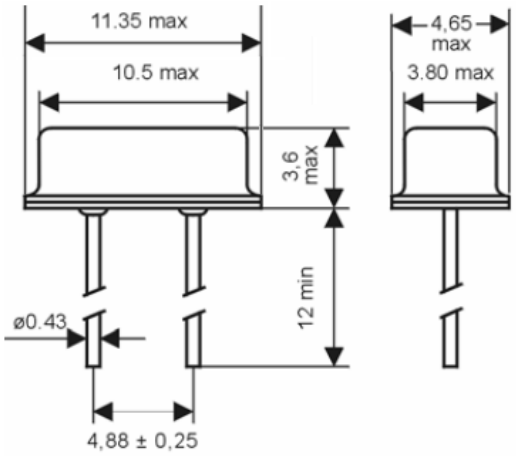

Рисунок 2.16 — Кварцовий резонатор HC49S

2.4.4 Вибір діодів

Діоди встановлені в силових колах схеми з максимальною напругою 50 В та струмом 100 мА. Для цього добре підходять діоди 1N4148 (рис. 2.17) витримують зворотною напругу 100 В то прямий струм 450 мА.

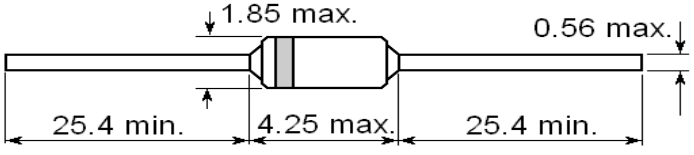

Рисунок 2.17 — Діод 1N4148

2.4.5 Вибір мікроконтролера

Мікроконтролер окрім основних функцій виконує також передачу даних на ПК. Тому однією з основних вимог є наявність вбудованого модуля USB. Контролер PIC18F2550 фірми Microchip, який має 28 виводів, що є цілком достатньо для даної схеми (рис. 2.18).

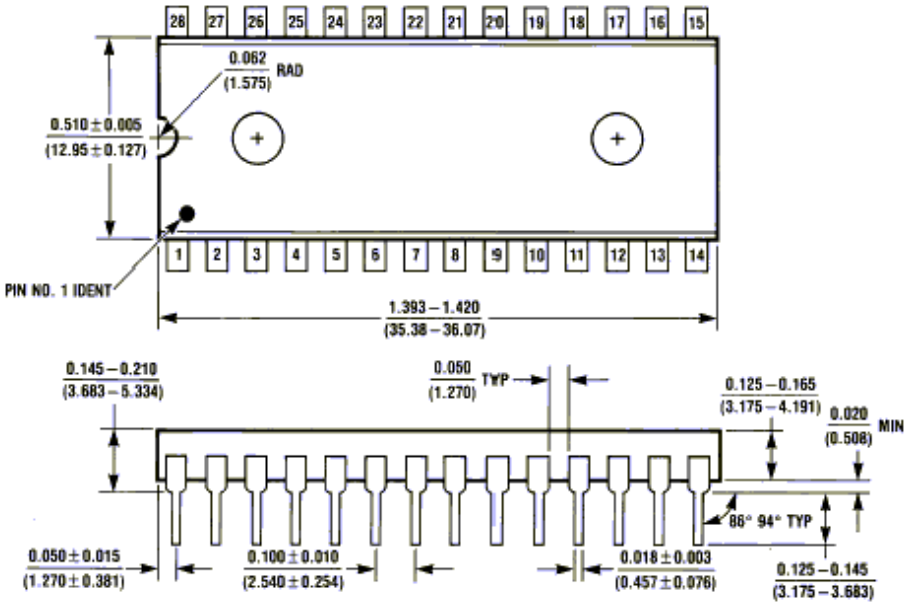

Рисунок 2.18 — Мікроконтролер PIC18F2550

2.4.6 Вибір транзисторів

Потужний польовий транзистор VT4 керує джерелом живлення 50 В, однак з огляду на те, що він працює в імпульсному режимі то необхідно обрати транзистор з деяким запасом. Транзистор IRF820 здатен працювати при напрузі стік-витік до 500 В та при струмі 2,5 А. Транзистор виконано в корпусі TO–220 (рис. 2.19)

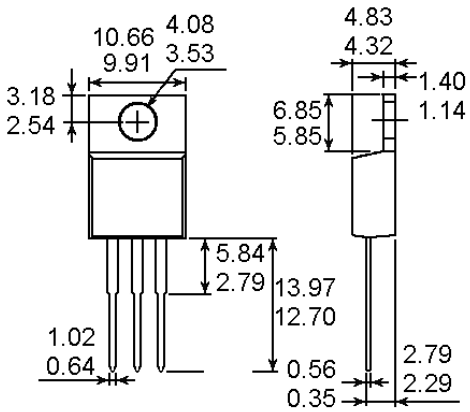

Рисунок 2.19 — Корпус TO–220

Транзистори VT2 і VT3 повинні бути комплементарними, обрано BC639 — NPN та DC640 — PNP з максимальною напругою база-емітер 80 В та максимальним струмом колектора 1 А, які виконано в корпусі TO–92 (Рис. 2.20).

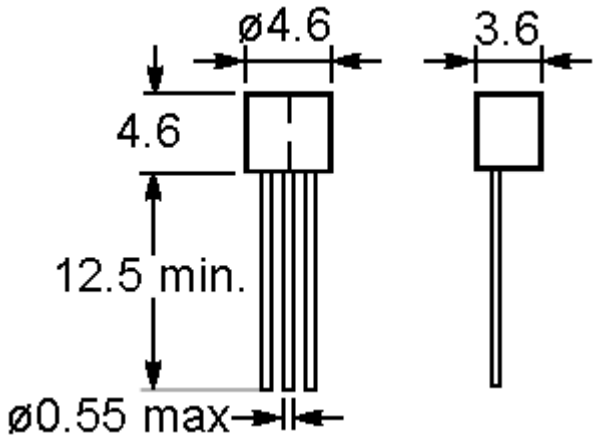

Рисунок 2.20 — Корпус TO–92

Малопотужний польовий транзистор VT1 необхідний щоб задавати правильний режим роботи силового польового транзистора. Для цього було обрано BF245A з максимальною напругою 30 В та максимальним струмом 0,1 А.

2.5 Розрахунок надійності

В даному розділі буде приведено розрахунок надійності схеми приладу за випадковими відмовами. Такий розрахунок виконується за схемою електричною принциповою та враховує не тільки інтенсивності відмов елементів кожного типу але й умови експлуатації в яких вони знаходяться, тобто вводиться поправочних коефіцієнтів які відображають механічні та кліматичні впливи на елементи. В цьому разі формула експлуатаційної інтенсивності відмов матиме такий вигляд:

$$\lambda_i = \lambda_{\text{Б}} \prod_{n=1}^{N} K_n ,$$

де $\lambda_{\text{Б}}$ — вихідна (базова) інтенсивність відмов елемента; N — кількість коефіцієнтів, що враховуються; K_n — n-тий коефіцієнт впливу.

В більшості випадків в розрахунку інтенсивності відмов присутній коефіцієнт, що враховує умови експлуатації, який буде однаковий для всіх елементів приладу:

$$K_E = K_1 K_2 K_3 = 16 \cdot 1,2 \cdot 1,05 \approx 20.$$

Значення відповідних коефіцієнтів вибирається з довідників згідно умов технічного завдання, відповідно до яких:

- Коефіцієнт умов використання $K_1 = 16$, для наземної апаратури;
- Коефіцієнт, що враховує температуру і вологість $K_2 = 1,2$ для відносної вологості повітря 80% та температури +25°С;
- Коефіцієнт, що враховує висоту використання апаратури $K_3 = 1,05$ для висоти 0–2 км.

Оскільки на схемі є досить багато елементів одного типу, у яких буде однакова інтенсивність відмов то доцільно згрупувати елементи з однаковою інтенсивністю. Також необхідно зауважити що всі формули та значення коефіцієнтів взяті з довідника [15].

Електролітичні конденсатори С5, С6 мають допустиму напругу 63 В, а максимальна напруга на схемі 50 В, звідки можна розрахувати коефіцієнт навантаження та інтенсивність відмов для них:

$$K_{\text{Н}} = \frac{U_{max}}{U_{\text{доп}}} = \frac{50}{63} = 0,79,$$

$$\lambda_i = \lambda_{\text{Б}} K_{\text{Н}} K_{\text{Е}} K_{\text{С}} = 0,19 \cdot 0,79 \cdot 20 \cdot 2,2 = 6,68.$$

Всі інші конденсатори в даній схемі використовуються в сигнальних колах, це означає, що максимальна робоча напруга не буде перевищувати 5 В, тоді їхній коефіцієнт навантаження буде:

$$K_{\text{Н}} = \frac{U_{max}}{U_{\text{доп}}} = \frac{5}{16} = 0,31.$$

А інтенсивність відмов:

$$\lambda_i = \lambda_{\text{Б}} K_{\text{Н}} K_{\text{Е}} K_{\text{С}} = 0,01 \cdot 0,31 \cdot 20 \cdot 0,84 = 0,053.$$

Інтенсивність відмов кварцового резонатора розраховується так:

$$\lambda_i = \lambda_Б K_Т K_Е = 0,025 \cdot 1 \cdot 20 = 0,5.$$

Малопотужні резистори, що працюють у сигнальних колах мають коефіцієнт навантаження який навіть у найгіршому випадку буде менший від 0,1. Тоді можна знайти інтенсивність відмов:

$$\lambda_i = \lambda_Б K_Н K_Е = 0,04 \cdot 0,1 \cdot 20 = 0,08.$$

Згідно розрахунків максимальна потужність резисторів, що працюють у колах живлення не перевищує 0,25 Вт, а отже коефіцієнт навантаження буде приблизно 0,5.

$$\lambda_i = \lambda_Б K_Н K_Е = 0,04 \cdot 0,5 \cdot 20 = 0,4.$$

Потужний резистор R26, як було розраховано, при відкритому транзисторі розсіює 5 Вт потужності, що є для нього максимальним значенням тому його коефіцієнт навантаження буде рівний 1.

$$\lambda_i = \lambda_Б K_Н K_Е = 0,04 \cdot 1 \cdot 20 = 0,8.$$

Інтенсивність відмов мікроконтролера визначається за формулою для інтегральних мікросхем:

$$\lambda_i = \lambda_Б K_{с,Т} K_Е K_{Корп} = 0,019 \cdot 4,9 \cdot 20 \cdot 1 = 1,87.$$

Коефіцієнт навантаження діода розраховується, як відношення максимального струму через нього в схемі до максимально допустимого струму. Максимальний струм на схемі через діоди не перевищить 200 мА, а допустимий — 450 мА:

$$K_Н = \frac{I_{max}}{I_{доп}} = \frac{200}{450} = 0,45.$$

Тоді можна розрахувати інтенсивність відмов для діодів VD1 і VD2:

$$\lambda_i = \lambda_Б K_Н K_Е = 0,1 \cdot 0,45 \cdot 20 = 0,89.$$

Потужність силового транзистора VT4, що розсіюється на схемі становить приблизно 5 Вт, а максимальна 125 Вт тоді можна знайти коефіцієнт навантаження:

$$K_{\text{Н}} = \frac{P_{max}}{P_{\text{доп}}} = \frac{5}{125} = 0,04.$$

Тоді з урахуванням навантаження та робочої температури коефіцієнт $K_{\text{Р}} = 0,1$, тоді:

$$\lambda_i = \lambda_{\text{Б}} K_{\text{Р}} K_{\text{Е}} = 0,886 \cdot 0,12 \cdot 20 = 0,2.$$

Таблиця 2.3 — Параметри надійності груп елементів

Позиційні позначення	Базова інтенсивність відмов $\lambda_{\text{Б}}\ 10^6$, 1/год	Коефіцієнт навантаження $K_{\text{Н}}$	Робоча інтенсивність відмов $\lambda_i \cdot 10^6$, 1/год	Кількість елементів в групі N_i
C1-C4, C7, C8	0,01	0,31	0,05	6
C5, C6	0,19	0,79	6,69	2
D1	0,02	4,90	1,88	1
VD1, VD2	0,10	0,44	0,90	2
VT1-VT3	0,06	0,10	0,12	3
VT4	0,09	0,12	0,21	1
R1-R20,R27	0,04	0,10	0,08	21
R21-R25	0,04	0,50	0,40	5
R26	0,04	1,00	0,81	1
ZQ1	0,03	1,00	0,50	1

Для малопотужних біполярних транзисторів, у яких допустима потужність складає 80 Вт, а робоча 35 Вт коефіцієнт навантаження буде:

$$K_{\text{Н}} = \frac{P_{max}}{P_{\text{доп}}} = \frac{35}{80} = 0,44.$$

За довідником підбираємо значення $K_{\text{Р}}$, який в при $K_{\text{Н}}$ = 0,44 та робочій температурі +25°С для кремнієвих біполярних транзисторів складає приблизно 0,1. А інтенсивність відмов відповідно:

$$\lambda_i = \lambda_{\text{Б}} K_{\text{Р}} K_{\text{Е}} = 0,06 \cdot 0,1 \cdot 20 = 0,12.$$

Для більшої наочності результати наведених вище розрахунків зведені до таблиці 2.3.

Сумарна інтенсивність відмов приладу визначається як сума інтенсивностей відмов кожного окремого елемента:

$$\lambda_c = \lambda_i \sum_{i=1}^{k} \lambda_i \cdot N_i = 16,48.$$

де N_i — кількість елементів в i-й групі.

Час напрацювання на відмову схеми відповідно буде:

$$T_0 = \frac{1}{\lambda_{\text{С}}} = \frac{1}{16,48 \cdot 10^{-6}} = 6,4 \cdot 10^4 \text{год}.$$

Далі можна розрахувати імовірність безвідмовної роботи протягом часу t:

$$P(t) = e^{-\frac{1}{T_0}}.$$

Графік залежності імовірності безвідмовної роботи від часу представлений на рисунку 2.21.

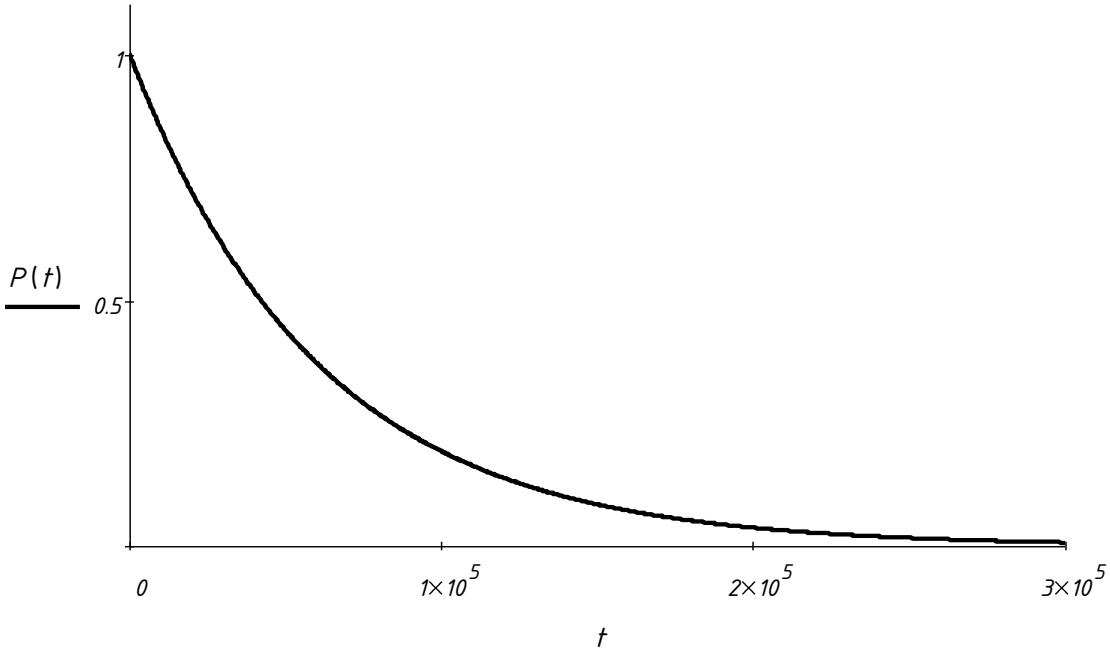

Рисунок 2.21 — Імовірність безвідмовної роботи

2.6 Висновки

Схема розроблена таким чином, щоб спростити та здешевити її виготовлення. Більшість вузлів можна замінити більш дорогими інтегральними рішенням, це може суттєво спростити розробку, але і підвищити вартість на виготовлення приладу.

На цьому етапі було проведено моделювання кожного окремого рішення та, в деякій мірі, взаємодію між ними. Тестування системи в цілому потребує наявності програмного забезпечення для керування систему, розробці якового присвячено наступний розділ.

3 РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ

Даний розділ присвячено розробці програмного забезпечення необхідного для керування системою. Воно складається з трьох відносно самостійних частини: програма для керування мікроконтролером, низькорівневий драйвер для взаємодії з приладом та модуль відображення. Кожна частина представляє собою окремий рівень роботи системи, вирішує певні задачі та має інтерфейс для взаємодії з іншими частинами.

3.1 Розробка програми керування мікроконтролером

Згідно структурної схеми, що зображена на рисунку 2.10 мікроконтролер виконує функції схеми керування (СК) та модуля передачі даних (МПД). Тобто керує процесом вимірювання та передачею результатів на ПК. В минулому розділі було обґрунтовано вибір мікроконтролера PIC18F4550, а цьому будуть описані особливості його використання та наведено алгоритм проведення вимірювання.

3.1.1 Вибір мови програмування

При написанні програм керування мікроконтролерами PIC існує можливість використання лише однієї з двох мов програмування: мови асемблера та С.

Програма розроблена на мові асемблера виконується швидше та потребує менше ресурсів ніж аналогічна програма написана на С. Написання програми для мікроконтролера на асемблері дозволяє більш тісно взаємодіяти із апаратними засобами мікроконтролера.

Другою превагою використання асемблера на етапі розробки є те, що деякі середовища схемотехнічного моделювання, наприклад Porteus ISIS, дозволяє використовувати не тільки скомпільовані програми (.hex), а і вихідний текст програми на асемблері (.asm). Це може суттєво прискорити розробку простих рішень за рахунок відмови від встановлення спеціальних інтегрованих середовищ розробки.

Однак за таку оптимізацію доводиться платити більшими затратами часу на розробку складніших програм. Оскільки для розробки на асемблері необхідно добре розуміти архітектуру конкретної моделі мікроконтролера, оскільки кожна серія має власну реалізацію мови асемблера. Проте фірма Microchip оснащує свої продукти детальною документацією в якій описано як апаратні особливості архітектури так і способи взаємодії з ними, тому досвідченим розробникам апаратури використання асемблера не завдасть значних труднощів.

Іншим можливим варіантом є використання мови С, переваги якої складно переоцінити. На відміну від асемблера мова С більш зрозуміла для розробники та дозволяє в деякій мірі абстрагуватись від апаратних особливостей архітектури мікроконтролера. Що в свою чергу зробить вихідний код більш гнучким та спростить його перенесення на інший тип мікроконтролера. За роки існування цієї мови в накопичилась велика кількість алгоритмів для рішення майже будь-якої задачі. Тому більшість розробників віддають перевагу саме цій мові.

Проте програмування на мові С потребує використання спеціальних інтегрованих середовищ розробки. На щастя фірма Microchip розробила середовище MPLAB спеціально для мікроконтролерів PIC. Яке включає всі інструментів для розробки та налагодження програмного коду. Для кожної серії мікроконтролерів необхідно використовувати різні компілятори, тому MPLAB створений з окремих модулів — він дозволяє встановлювати різні інструменти, залежно від ситуації.

Незважаючи на всі переваги та недоліки кожної з мов програмування вирішальне значення мала наявність бібліотеки для роботи з ПК через USB.

Універсальний драйвер Microchip для роботи з USB під керуванням операційної системи Windows (Microchip General Purpose USB Windows Driver), складається з трьох частин: бібліотеки для керування USB модулем мікроконтролера, драйвер для операційної системи та DLL бібліотека, що полегшує програмну взаємодію з драйвером.

Разом з бібліотекою поставляються вихідні коди програм для мікроконтролерів різних типів, та демонстраційні приклади, що показують основи роботи з драйвером. Вони побудовані таким чином, щоб можна було легко відокремити бібліотечні файли і використати їх як основу для створення власних програм.

Один з таких проектів і було взято за основу. Він має вже готову архітектуру, призначену для роботи з ПК та налаштовану конфігурацію мікроконтролера. Всі інші засоби є досить простими і можуть бути з легкістю використані з будь-якою архітектурою.

Перш ніж перейти до розробки слід зазначити особливості використання бібліотеки. Для роботи з USB потрібна висока стабільність тактових імпульсів, тому використання звичайної конфігурації є небажаним. Для стабілізації тактової частоти до мікроконтролера підключений кварцовий резонатор з частотою 1 МГц. З урахуванням заданого слова конфігурації мікроконтролера його тактова частота становить 5 МГц. Це дуже важливий параметр, оскільки від нього залежить частота переключення таймерів, а отже і значення, що необхідно задати, щоб отримати потрібні інтервали часу.

3.1.2 Таймери та переривання

Коли необхідно виконати дію через точно зазначені інтервали часу можна поставити функцію затримки, що буде виконуватись заданий час а потім передасть керування потрібному коду. Проте такий підхід має один недолік — під час очікування мікроконтролер не може виконувати жодних інших дій. Цю проблему вирішують таймери.

Використання таймерів, як і більшості периферійних модулів засновано на перериваннях. Переривання це механізм, що дозволяє перервати виконання програми та передати керування іншій ділянці коду. По завершенню виконання цієї ділянки керування буде передано туди де було перервано виконання основного коду. Це дозволяє швидко реагувати на настання певних подій.

Мікроконтролери сімейства PIC18 мають дворівневу систему переривань, тобто переривання можуть мати два пріоритети: високий і низький, відповідно і дві функції обробники [16].

Компілятор мови С надає досить зручний засіб для роботи з обробниками переривань. За документацією, щоб визначити функцію як обробник необхідно додати ключове слово *interrupt*, та за необхідності пріоритет (наприклад *low_priority*) у визначення функції, яке матиме наступний вигляд: *void interrupt low_priority low_isr (void)*. Це означає що функція буде викликана при появі переривання з низьким пріоритетом.

На жаль в нових версіях компілятора таке визначення обробників переривань не працює. Тому адреси потрібних функцій потрібно «вручну» занести у відповідні вектори переривань. Вектор — це адреса в програмній пам'яті, за якою відбувається автоматичний перехід при виникненні будь-якого дозволеного переривання.

Як було зазначено даний мікроконтролер має два типи переривань, а отже і два вектора:

- 0x08 — вектор переривань з високим пріоритетом;
- 0x18 — вектор переривань з низьким пріоритетом.

Слід пам’ятати, що вразі відключення пріоритетів буде використовуватись лише один вектор 0x08.

Щоб налаштувати переривання необхідно дозволити переривання глобальні та периферійні встановленням біті GIE та PEIE регістру INTCON відповідно та пріоритети встановленням IPEN регістру RCON [17].

3.1.3 Програмне керування зміною напруги

Керування напругою колектор-емітер відбувається за допомогою схеми понижуючого імпульсного перетворювача постійного струму, описаної в попередньому розділі. Керування таким перетворювачем відбувається за допомогою широтно-імпульсної модуляції (ШІМ), тобто результуюча напруга залежить від шпаруватості імпульсів що подаються на вхід схеми керування. Це створює необхідність у відносно високій стабільності

генерованих імпульсів, яку не можна забезпечити при керуванні з основного циклу програми.

Найкращий варіант керування таким перетворювачем це використання апаратних засобі ШІМ (PWM, Pulse-Width Modulation) мікроконтролера. Для цього в мікроконтролері передбачено модуль CCP. Абревіатура розшифровується як Capture/Compare/PWM (Захват/Порівняння/ШІМ). Обрана модель мікроконтролера має два незалежних модулі CCP1 та CCP2. Однак вони, є не зовсім незалежними, оскільки працюють на одній частоті та використовують однаковий спосіб підключення, проте є можливість задавати шпаруватість для кожного з них окремо.

Для роботи модуля потрібно налаштувати виводи RC2 (CCP1) та RC1 (CCP2) на вихід, та активувати таймер 2. При використанні модулів CCP цей таймер буде рахувати такти та керувати довжиною імпульсів, а отже не доступний для інших функцій [18][19].

Ініціалізація модуля CCP режимі ШІМ включає наступні етапи. Перш за все потрібно встановити режим роботи модуля та встановити відповідні виводи порту C на вихід. За режим роботи відповідають біти 0–3 регістру CCP1CON. Вибір режиму ШІМ відбувається встановленням бітів 2 та 3 (останні два ігноруються і можуть мати будь-яке значення) [20]

За період імпульсів відповідає регістр PR2, що має 8 розрядів, а отже його значення не може перевищувати 255. Фактичне значення періоду розраховується за такою формулою:

$$P_{PWM} = 4T_{osc}N_{prescale}(PR2+1),$$

де P_{pwm} — період імпульсів, T_{osc} — тактова частота роботи мікроконтролера, $N_{prescale}$ — значення переддільника таймера. Цей параметр може приймати три значення 1, 4 або 16. Для цього в біти 0–1 регістра T2CON потрібно встановити в значення 00, 01 або 11 відповідно.

За формулою на основі потрібної частоти імпульсів можна розрахувати значення PR2. Краще спершу взяти значення переддільника рівним 16 та

розрахувати PR2, якщо значення перевищить 255 то взяти менше значення переддільника та повторити розрахунок. Використання регістра PR2 та таймера для роботи ШІМ можна побачити на рисунку 3.1.

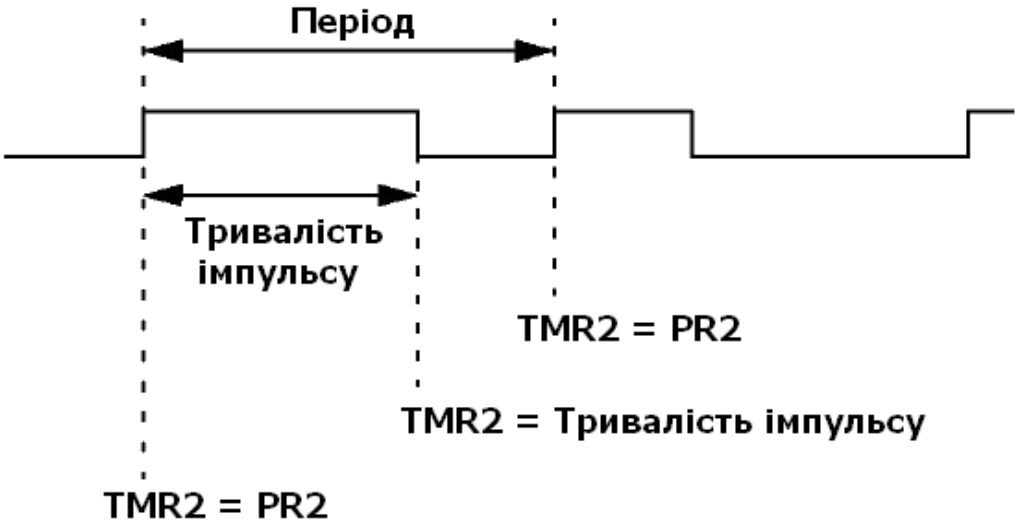

Рисунок 3.1 — Часова діаграма циклу ШІМ

Видно, що зміна значення напруги з високого на низьке і навпаки відбувається коли таймер досягне деякого значення. Це означає, що частота імпульсів кратна частоті переключень таймера, який в свою чергу залежить від тактової частоти мікроконтролера. Очевидно, що підвищення частоти можна досягти зменшенням граничного значення лічильника. Однак таке зменшення призведе до того, що імпульс триватиме меншу кількість тактів і поділити його можна буде на меншу кількість частин. Параметр, що визначає на скільки частин можна поділити імпульс називається розрядністю ШІМ. Від дорівнює кількості біт, в яку можна записати число відношення частоти роботи мікроконтролера до необхідної частоти імпульсів. Наприклад, якщо тактова частота роботи мікроконтролера 20 МГц, а необхідна частота імпульсів 10 кГц то їх відношення буде 2000 або 11111010000, це 11 двійкових розрядів при максимальній можливій розрядності 10. Тому в цьому випадку розрядність ШІМ дорівнює 10 [19].

Забезпечення правильної роботи перетворювача не потребує конкретної частоти, тому було обрано максимальний період 255 та значення переддільника 1:1 в результаті буде отримано період:

$$P_{PWM} = 4T_{osc}N_{prescale}(PR2+1) = \frac{4}{5\cdot 10^6}\cdot(255+1) = 205\cdot 10^{-6}\text{ с.}$$

Це забезпечить досить високу частоту та максимальну розрядність ШІМ — 10 біт. При цьому значення шпаруватості може бути в діапазоні від 0 до 1023, отже можна змінювати вихідну напругу з кроком менше 0,1% від напруги живлення, що в даному випадку становить 0,05 В.

3.1.4 Отримання результатів вимірювання

Як відомо мікроконтролер є цифровим пристроєм та зазвичай працює з цифровими сигналами. Проте часто виникає необхідність отримувати величину аналогових сигналів. Не зважаючи на те, що існує велика кількість схем та інтегральних рішень аналогово-цифрових перетворювачів (АЦП) фірма Microchip намагається забезпечити якомога більше своїх мікроконтролерів вбудованим модулем АЦП (ADC, Analog-to-Digital Converter), навіть найпростіші PIC10, що мають всього шість виводів.

Обраний мікроконтролер має 13 виводів (AN0–AN12) на які можна подавати аналогові сигнали, але лише один фізичний модуль АЦП, який підключається до цих виводів за допомогою мультиплексора.

Найважливішою характеристикою АЦП є його розрядність, вона показує з яким мінімальним кроком можна вимірювати величину аналогової напруги, яку можна визначити за наступною формулою:

$$\Delta U = \frac{U_{ADC}}{2^{N_r}},$$

де U_{ADC} — опорна напруга АЦП, N_r — кількість розрядів АЦП [21].

Мікроконтролер має 10 розрядів АЦП, отже мінімальний крок аналогового сигналу при використанні напруги живлення як опорної становитиме приблизно 5 мВ.

Роботу модуля АЦП контролюють три восьми розрядні регістри ADCON0, ADCON1 та ADCON2. Перший дозволяє вмикати модуль та обирати який вивід буде підключено до нього. Другий встановлює джерело

напруги та режими роботи виводів (аналоговий чи цифровий). Третій контролює час перетворення та частоту тактів модуля [22].

Більшість бітів можна встановити як показано в документації, однак деякі потрібно розглянути детальніше.

Біт ADFM регістру ADCON2 визначає вирівнювання результату. Річ у тім, що 10 розрядів результату роботи АЦП знаходяться у 16-ти розрядному регістрі ADRES (ADRESH та ADRESL) і їх потрібно там певним чином розмістити (рис 3.2).

Рисунок 3.2 — Вирівнювання результатів

Якщо планується використовувати лише 8 розрядів результату, наприклад коли не потрібна велика точність вимірів або опорна напруга не досить стабільна рекомендується встановити лівостороннє вирівнювання (ADFM = 0), а результат брати з регістру ADRESH. Якщо ж планується використовувати всі 10 біт результату то краще вирівняти в право та скористатися ADRESH і ADRESL як одним 16-розрядним регістром.

Інша особливість використання модуля АЦП полягає в тому, що для своєї роботи він використовує переривання. Це потрібно, оскільки процес отримання цифрового значення сигналу займає певний час. В цей час мікроконтролер може виконувати інші дії, а коли перетворення буде завершено буде викликано переривання. Тож при ініціалізації модуля необхідно також настроїти переривання: біт ADIE регістру PIE1 дозволяє переривання від АЦП, біт ADIP регістру IPR1 встановлює пріоритет переривань від АЦП, біт ADIF регістру PIR1 скидає прапорець переривань.

Для запуску власне перетворення потрібно встановити біт GO_DONE регістру ADCON0 та скинути ADIF, щоб переривання не було викликано одразу.

Щоб переконатись, що переривання викликане саме завершенням аналогово-цифрового перетворенням у функції обробнику потрібно перевірити, що встановлено одночасно ADIF та ADIE регістру PIE1, та скинути перший, що переривання не було викликано повторно.

3.1.5 Загальний алгоритм роботи мікроконтролера

Вище було описано налаштування та принципи роботи кожної необхідної частини окремо. Далі буде показано як вони взаємодіють в процесі вимірювання.

Під час запуску апаратної частини відбувається ініціалізація вказаних модулів з описаною конфігурацією.

При отриманні сигналу до початку вимірювань відбувається скидання всіх лічильників та встановлення значення напруги колектор-емітер і струму бази у початкове значення. Відбувається розрахунок початкового значення шпаруватості імпульсів для отримання необхідної напруги. Це значення контролюється за допомогою зворотного зв’язку. Через резистивний дільник воно подається на вивід AN1. Коли значення напруги між двома запусками АЦП буде менше ніж задана похибка — вважається, що напруга встановлена і починається вимірювання струму колектора. Після отримання цього значення напруга збільшується на заданий крок та все повторюється.

Кожна характеристика це досить великий об’єм даних. Максимальна розрядність ШІМ генератора 10, тобто можна отримати 1024 значення напруги. Розрядність керованого джерела струму 8, тобто 256 значень. Отже результуюча може містити:

$$N_t = 2^{N_{PWM}} \cdot 2^{N_{\text{КДС}}} = 1024 \cdot 256 = 262144 \text{точок},$$

де N_{PWM} — розрядність ШІМ, $N_{\text{КДС}}$ — розрядність керованого джерела струму.

Кожна точка представлена числом з плаваючою комою з одинарною точністю, тобто 4 байти на точку. Отже в результаті вся характеристика займатиме:

$$V_{res} = V_{float} \cdot N_t = 4 \cdot 262144 = 1048576 \text{ байт.}$$

Тобто в найгіршому випадку повний результат вимірювання потребує більше 1 МБ пам’яті, тоді як обрана модель мікроконтролера має всього 2048 байт пам’яті даних, а отже збереження повного результату є неможливим. З огляду на це всі виміряні дані потрібно одразу ж передавати на ПК. Це можливо зважаючи на те, що всі використовувані модулі (таймери, АЦП та ШІМ) працюють у фоновому режимі, а мікроконтролер може виконувати передачу на ПК між перериваннями.

Слід зазначити, що прийом та передача інформації на ПК має обмеження в часі. Тому, щоб уникнути переривання передачі даних, перед початком передачі потрібно відключити переривання на глобальному рівні (INTCON.GIE = 0), а після завершення знову дозволити.

Коли лічильник напруги досягне максимального значення він буде скинутий, а лічильник струму збільшений на заданий крок струму та процес повторено.

Для індикації стану приладу використано світлодіод, що буде блимати ти на протязі всього циклу вимірювань. Для цього використано апаратний таймер 0 мікроконтролера.

3.2 Розробка драйвера

Для взаємодії з апаратною частиною системи необхідно спеціальне програмне забезпечення — драйвер. Який візьме на себе функції передачі сигналів керування та отримання результатів вимірювань. Згідно структурної схеми (рис 2.10) це — драйвер шини (ДШ).

Драйвери взаємодіють з досить примітивними пристроями, що, зазвичай, керуються мікроконтролерами. В таких пристроях може бути досить не типова організація пам’яті. В цьому випадку основна функція

драйвера полягає у тому, щоб конвертувати дані з формату ПК у формат мікроконтролера і навпаки. Тому для написання драйверів необхідно використовувати мову, з одного боку, достатньо низького рівня, щоб мати можливість прямого доступу до пам’яті, а з іншого — достатньо високого, щоб не створювати труднощі при розробці. Саме такою мовою є C++.

Як було зазначено зі сторони мікроконтролера передачу даних виконує вбудований USB-модуль. Проте, протокол USB досить складний і навіть з урахуванням його підтримки на апаратному рівні розробка драйвера «з нуля» потребує багато часу та ресурсів. Тому для полегшення розробки було використано вихідні коди універсального драйвера фірми Microchip для операційної системи Windows (Microchip General Purpose USB Windows Driver).

Організація програмної взаємодії з драйвером відбувається за допомогою DLL бібліотеки, в якій реалізовано функції для з’єднання з пристроєм, прийому і передачі даних та закриття з’єднання.

Однак ця бібліотека є лише посередником, що спрощує взаємодію з системним драйвером, який і виконує безпосередню передачу даних. Щоб успішно встановити з’єднання з пристроєм, при першому підключенні, коли операційна система запитає драйвер необхідно вказати шлях саме до нього.

З огляду на те, що універсальний драйвер Microchip призначений для використання лише з операційною системою Windows, а розробка драйверів часто потребує тісної взаємодії з операційною системою на низькому рівні то інші операційні системи, для керування приладом, не розглядались. Очевидно, що для розробки драйвера приладу доречно використати середовище від того ж виробника що і операційна система — Microsoft Visual Studio.

3.2.1 Створення і використання бібліотек

Як було зазначено використання універсального драйвера відбувається через за допомогою DLL бібліотеки. Драйвер системи також планується організувати у вигляді бібліотеки. Це, згідно принципу модульності,

дозволить логічно відділити програму виводу тривимірної графіки від підсистеми керування приладом і полегшить майбутнє використання останньої для інших цілей. Таким чином драйвер та апаратна частина стануть лише джерелом даних для підсистеми відображення. А її саму можна використовувати окремо, для виводу збережених раніше даних. Тому робота з DLL бібліотеками є дуже важливою і її потрібно розглянути детальніше.

Зазвичай розділення програмного коду на бібліотеки потрібно для забезпечення принципу модульності. Він зобов’язує організовувати програми як сукупність невеликих незалежних блоків, які називаються модулями, структура і поведінка яких підкоряються певним правилам. Використання модульного програмування дозволяє спростити тестування програми і виявлення помилок. Апаратно-залежні частини можуть бути жорстко відділені від інших частин, що покращує мобільність створюваних програм.

Модуль — функціонально закінчений фрагмент програми. У багатьох мовах (але далеко не обов'язково) оформляється у вигляді окремого файлу з вихідним кодом або пойменованої безперервної її частини. Деякі мови передбачають об'єднання модулів в пакети [23].

Створення бібліотеки динамічного зв’язування (Dynamic Link Library, DLL) необхідно виконати деякі дії.

В середовищі Microsoft Visual Studio необхідно створити новий проект типу Win32. У вікні створення проекту вибрати «Пустий проект». Після компіляції буде отримано DLL, бібліотеку імпорту (.lib) та бібліотеку експорту (.exp).

Перш за все необхідно створити файл заголовку (.h) з сигнатурами функцій, який може бути використаний при підключенні DLL з бібліотекою імпорту. В цьому файлі повинні бути описані усі сигнатури функцій, що мають бути викликані зовнішніми програмами.

Всі ці функції мають бути помічені спеціальними ключовими словами. Наприклад, якщо функцію з сигнатурою «*void Start();*» необхідно викликати з

іншої програми то записати так «*extern "C" __declspec(dllexport) void Start();*». Так само необхідно записати функцію і в фалі з реалізацією.

Перший варіант використовувати DLL разом з бібліотекою імпорту (.lib), яка виходить при компіляції проекту бібліотеки. Цей метод дуже простий, так як в такому випадку потрібно просто включити заголовки бібліотеки та саму бібліотеку в проект та викликати необхідні функції.

Це буде чудово працювати, якщо заголовки і бібліотека імпорту знаходиться в каталозі, прописаному в бібліотечних шляхах. Перед запуском програми, потрібно переконатись, що DLL знаходиться в каталозі, прописаному в системній змінній PATH або в тому ж каталозі, що і виконуваний файл, інакше буде отримано повідомлення про помилку. Якщо декілька програм використовують цю DLL, потрібна всього одна її копія, що лежить, наприклад, в каталозі Windows\System.

Другий варіант — завантажити DLL «на льоту». Це потрібно в разі, якщо DLL створено сторонніми розробниками які не надали доступ до заголовків і бібліотеки імпорту [24].

На відміну від першого, де бібліотека підключається ще на етапі компіляції, тобто якщо відповідні файли не знайдено буде видано помилку компіляції і проект не буде побудований. Тут бібліотека та всі її функції завантажуються динамічно на етапі виконання програми.

Другий метод є більш гнучким, оскільки він дозволяє завантажувати бібліотеки вибірково, або при виконанні деяких умов. Таким чином можна організувати динамічне завантаження лише тих частин програми які необхідні в даний момент користувачу, і вивантажувати коли вони вже не потрібні. В той же час як при першому варіанті всі бібліотеки гарантовано будуть завантажені перед початком виконання.

Далі описано кроки, що необхідно заробити щоб завантажити DLL-бібліотеку. Всі описані функції є функціями прикладного інтерфейсу Windows — WinApi, тому для їх використання необхідно включити заготовочний файл windows.h.

Для завантаження бібліотеки потрібно викликати функцію **LoadLibrary.** Вона приймає єдиний аргумент — строку зі шляхом до бібліотеки, яку необхідно завантажити. Шлях може бути абсолютний, відносний або містити тільки назву файлу. В останньому випадку пошук буде проводитись спочатку в каталозі з виконуваним файлом, а потім у всіх каталогах що описані в системній змінній PATH. Виклик функції поверне дескриптор бібліотеки типу HMODULE або значення NULL, якщо бібліотеку не знайдено.

Тепер бібліотека є завантаженою в пам’ять і доступна для використання. Далі потрібно отримати адреси функцій, які будуть викликатись з неї. Для цього використовують функцію **GetProcAddress**, яка отримує на вхід дескриптор бібліотеки та ім’я функції яку необхідно завантажити. Функція GetProcAddress повертає значення типу FARPROC, це і є вказівник на потрібну функцію. Отримане значення за допомогою приведення типів необхідно привести до правильної сигнатури. Найпростіший варіант це визначити новий тип вказівника на функцію з необхідною кількістю і типами параметрів та типом значення, що вона повертає, визначити змінну цього типи та виконати приведення до нього. Тепер з отриманою змінною вказівника можна поводитись як зі звичайною функцією — викликати з необхідними їй списком параметрів та отримувати результат її роботи.

Якщо функцію з заданим ім’ям не знайдено в бібліотеці то GetProcAddress поверне значення NULL. Це свідчить про те, що сталась помилка і подальше використання бібліотеки може привести до невизначеної поведінки всієї програми, оскільки з великою імовірністю завантажено неправильну бібліотеку. Тому необхідно контролювати отримане значення вказівника і вразі помилки переривати роботу програми.

Системна функція **FreeLibrary** вивантажує бібліотеку коли вона вже непотрібна. В якості параметру вона отримує дескриптор бібліотеки, що необхідно вивантажити, та повертає логічне значення, що показує чи успішно була вивантажена бібліотека.

Технічно функція FreeLibrary не вивантажує бібліотеку, а лише зменшує кількість вказівників на неї. Власне вивантаження з пам’яті буде виконано лише тоді коли лічильник вказівників досягне нуля. Це так званий механізм підрахунку посилань. Він дозволяє економити ресурси системи коли одну бібліотеку використовують декілька додатків. Перший виклик LoadLibrary завантажує бібліотеку в пам’ять, а всі інші лише збільшують лічильник посилань.

Тому дуже важливо викликати FreeLibrary кожен раз при завершенні програми, навіть при аварійному, інакше лічильник посилань ніколи не дійде до нуля і бібліотека не буде вивантажена, аж до перезавантаження комп’ютера. В цей час її файл неможливо перемістити, перейменувати або перекомпілювати проект, оскілки файл вважається зайнятим.

З DLL бібліотеки можна викликати лише окремі функції. А при розробці програмного коду бібліотеки в об’єктно-орієнтованому стилі весь функціонал розділений між об’єктами, які досить часто повинні бути в пам’яті на протязі всієї роботи програми. Виникає ситуація при якій об’єкт повинен бути створений перед викликом функції та видалятись після кількох викликів різних функцій. Навіть якщо створювати об’єкт «ліниво», тобто при кожному виклику кожної функції перевіряти чи створений об’єкт і створити його якщо ні, залишається невирішеною проблема видалення.

Найбільш простий варіант це в бібліотеці створити функції, що необхідно викликати до початку роботи та після її завершення. Але може ускладнити використання бібліотеки, оскільки розробники часто забувають викликати такі функції, особливо ті що пов’язані з очисткою ресурсів, адже їх влив не роботу програми не завжди очевидний.

Для вирішення цієї проблеми можна додати спеціальну функцію **DllMain**, яка буде автоматично викликана при виникненні однієї з описаних нижче подій. Щоб визначити, що це за подія необхідно порівняти значення параметру fdwReason з однією з наступних констант:

- DLL_PROCESS_ATTACH — бібліотека була завантажена в віртуальний адресний простір поточного процесу, як результат запуску процесу або виклику LoadLibrary.
- DLL_PROCESS_DETACH — бібліотека вивантажена з віртуального адресного простору процесу, що викликав через невдалу спробу завантаження або тому, що лічильник посилань досяг нуля.
- DLL_THREAD_ATTACH — Поточний процес створює новий потік. Коли це відбувається, система викликає функцію точки входу всіх бібліотек DLL, що в даний час приєднані до цього процесу. Виклик виконується в контексті нового потоку.
- DLL_THREAD_DETACH — Потік завершився повністю. Якщо DLL зберігає вказівник на виділену пам'ять, потрібно використовувати цю можливість, щоб звільнити її. Система викликає функцію точки входу всіх завантажених в даний момент бібліотек DLL з цим значенням [25].

Найбільш цікавими є перші дві події, тому, що саме при їх виникненні можна виконати ініціалізацію і очистку ресурсів в середині бібліотеки.

При додаванні до порожнього проекту функції DllMain можна отримати помилку, що така функція уже визначена. Щоб позбавитись від неї необхідно просто іще раз перекомпілювати проект.

3.2.2 Алгоритм та структура програми

Структура програмного коду цієї частини системи організована в об’єктно-орієнтованому стилі. Такий підхід є більш гнучким оскільки він дозволяє розділити функціонал програми між окремими класами. Кожен клас містить дані і функції призначені для рішення однієї конкретної підзадачі. Наприклад якщо клас використовує бібліотеку то він сам її завантажує, використовує та вивантажує коли це йому необхідно, так щоб іншим класам, які використовують перший не було необхідності знати про це.

Все що відомо про клас це його інтерфейс, тобто набір функцій, які можна викликати у об’єкта цього класу, щоб змусити його виконати деякі дії.

В термінах об’єктно-орієнтованого програмування функції класу називаються методами. Різниця в тому, що метод виконує дії для конкретного об’єкта. Тобто в метод на відміну від функції неявно передається вказівник на цей об’єкт. Надалі для позначення функції-члену класу буде використовуватись термін метод.

USBDevice основний клас бібліотеки драйвера. Він є об’єктним відображенням підключеного пристрою з яким можна обмінюватись інформацією, тобто до якого можна підключитись, послати запит та отримати відповідь. Цей клас є найбільш низькорівневим, він безпосередньо взаємодіє з універсальним драйвером, за допомогою засобів, що надає операційна система.

Слід зауважити, що даний клас є абстрактним тобто не представляє конкретного пристрою і має лише функціонал доступний для будь-якого USB пристрою, що може бути функціонувати подібним чином. Додавати реалізацію функціоналу для конкретного пристрою можна за допомогою механізму наслідування. Тобто похідні класи повинні додати можливість відправляти конкретні повідомлення, що можуть бути оброблені на стороні мікроконтролера.

Ініціалізація пристрою проходить в два етапи. На першому завантажується бібліотека з універсальним драйвером та проводиться пошук в ній необхідних для роботи функцій. На другому проводиться пошук підключених до USB-порту пристроїв. В разі якщо файл бібліотеки, одну або більше функцій з неї не знайдено або немає підключених пристроїв буде отримано повідомлення про помилку та викликане аварійне завершення роботи, а всі спроби послати повідомлення будуть завершуватись з помилкою. Для спрощення обидві ці дії об’єднані в одному методі Connect, однак для більшої гнучкості кожну с цих дій можна викликати окремо.

Всі помилки, що можуть повернути виклики методів USBDevice мають унікальний код, а отже однозначно вказують на причину виникнення помилки.

Коди помилок записані у вигляді констант у спеціальному файлі заголовку (USBDeviceError.h), який можна підключити при використанні бібліотеки драйвера характерографа із програми на мові C або C++. Однак можливі варіанти використання бібліотек з іншими мовами програмування в які неможливо підключити згаданий файл заголовку. В цьому випадку доведеться працювати безпосередньо з числовими значеннями кодів. В таблиці 3.1 наведено константи можливих помилок, їх числові значення та короткий опис.

Коди помилок від 0 до 1000 є зарезервованими використання при подальшій розробці даного класу. Похідні класи можуть додавати нові коди помилок починаючи зі значення, що не входять в визначений діапазон.

Для передачі даних використовується метод SendReceive, який посилає дані вказаної довжини та приймає дані у відповідь. Якщо обмін не відбувся за час, що вказується при виклику, то метод завершиться з помилкою.

Передача даних реалізована у вигляді, так званих, сесій. Тобто фактичне з’єднання з пристроєм відкривається безпосередньо перед відправленням даних та закривається одразу після отримання відповіді, або в разі збою при передачі. Це дозоляє зменшити затрати ресурсів.

Власне передача даних відбувається за допомогою іменованих каналів Windows (Named Pipes). Це засіб дозволяє організувати передачу даних між локальними процесами, а також між процесами, запущеними на різних робочих станціях в мережі.

Канали типу Pipe найбільше схожі на файли, тому вони досить прості у використанні.

Таблиця 3.1 — Коди помилок

Константа	Код	Опис
STATE_OK	0	Виклик функції завершися успішно.
NO_DLL	1	Помилка при завантаженні бібліотеки — файл бібліотеки не знайдено.
NO_FUNCTION	2	Помилка при завантаженні бібліотеки — неможливо отримати адресу однієї або більше функцій, що необхідно завантажити. Тобто в завантаженому файлі немає функції з відповідним ім’ям.
NO_DEVICE_CONNECTED	3	Пристрій не знайдено. Необхідно пересвідчитись, що підключено до USB порту комп’ютера.
NOT_FULL_DATA	4	Помилка при отриманні даних з мікроконтролера — прийнято менше ніж очікувалося.
INVALID_PIPE_HANDLE	5	Не вдалось відкрити з’єднання.
CALL_GET_LAST_ERROR	6	Виклик однієї з системних функцій Windows завершився з помилкою. Щоб отримати більше інформації необхідно викликати функцію GetLastError().
SESSION_ALREADY_OPEN	7	Спроба відкрити другу сесію, коли перша ще не закрита. Виникає при використанні кількох потоків, або внаслідок неуважного використання методів при ручному відкритті сесій.
COMMUNICATION_ERROR	8	Помилка передачі. Код відповіді від мікроконтролера не співпадає з очікуваним.

Через канал можна передавати дані тільки між двома процесами. Один з процесів створює канал, інший відкриває його. Після цього обидва процеси можуть передавати дані через канал в одно або двосторонньому режимі, використовуючи для цього добре знайомі функції, призначені для роботи з файлами, такі як ReadFile і WriteFile. Слід зауважити, що додатки можуть виконувати синхронні або асинхронні операції з каналами Pipe, аналогічно тому, як це можна робити з файлами [26].

В класі USBDevice є ще один метод призначений для передачі даних — SendRequest. На відміну від попереднього він сам відкриває та закриває сесії. Цей метод отримує параметром вказівник на об’єкт класу DeviceRequest або похідного від нього.

Клас DeviceRequest є об’єктним представленням запиту. Він є абстрактним та розроблений для спрощення рутинних операцій, таких як створення та видалення буфера, слідкування кількістю даних у буферах та убезпеченням від переповнення. Похідні класи можуть реалізовувати більш складну логіку для наповнення буфера даними для відправки та обробки прийнятої відповіді. Таким чином реалізовано логічне розділення обов’язків при якому кожен клас похідний від DeviceRequest є конкретною дією, що може бути виконана пристроєм.

За завантаження бібліотеки універсального драйвера відповідає клас LibraryLoader. Він містить методи для завантаження, вивантаження та отримання шляху до цієї бібліотеки. Потреба у ньому з’явилась з огляду на те, що при розробці кожна частина програми знаходиться у власній папці проекту, а після завершення всі частини готового продукту будуть знаходитись в одній. Іншими словами цей клас знає де потрібно шукати необхідні бібліотеки. З цієї ж причини йому необхідний метод, що дозволяє дізнатись з якого місця було завантажено бібліотеку.

3.2.3 Асинхронні операції

Метод SendRequest має одну проблему, яка полягає в тому, що він синхронний. Це означає, що при виклику цього методу керування не буде

повернуто доки не буде отримано повну відповідь від пристрою. При використанні простого програмного забезпечення, наприклад консольних програм, така поведінка є допустимою і навіть бажаною, оскільки в цьому випадку очікування найлегший спосіб відслідкувати момент завершення операції, щоб почати обробку результатів. Необхідно просто помістити алгоритм оброки після виклику методу. Однак і в цьому випадку, якщо операція виконується надто довго, є ризик то що користувач вважатими, що програма «зависла».

У випадку ж графічних додатків, всі дії виконують послідовно у, так званому, основному циклі. В цьому циклі, що безперервно виконується, проводиться оновлення графіки (кожен разу вся графічна частини повністю перерисовується), перевірка надходження нових запитів від користувача та виконання дій у відповідь на останні (наприклад відправлення запиту до пристрою). Якщо ж хоч одна частина буде виконуватись надто довго, що це помітив користувач, а для цього досить навіть пів секунди, виникле ефект короткочасного зависання.

Щоб подолати цей недолік було додано метод SendRequestAsync, який є аналогічним методу SendRequest, за винятком того, що він є асинхронним, тобто відправлення запиту та очікування відповіді виконується в іншому потоці (Thread).

Щоб пояснити цей термі необхідно розглянути модель процесів операційної системи. Процесом називається екземпляр завантаженої в пам'ять програми, що виконується в системі. Цей екземпляр може створювати потоки, які представляють собою послідовність машинних інструкцій для виконання. А процес є лише своєрідним контейнером для об’єднання потоків.

Будь-який процес має хоча б один потік (основний), що був створений разом з ним. Якщо в системі є достатня кількість процесорів то всі вони можуть виконуватись одночасно, однак це рідко буває так. Тому зазвичай потоки виконуються не одночасно, по черзі. Однак перемикання між ними

відбувається настільки швидко, що створюється ілюзія паралельного виконання. Послідовність перемикання, як і час роботи потоку розраховується динамічно і жодним чином не гарантується.

В Windows реалізовано «витісняючу» багатозначність, тобто система може перервати будь-який потік та запустити інший. Саме це може створити найбільше проблем. Кожен процес має спільні ресурси до яких може отримати доступ кожен потік. Якщо потік, що використовує ресурс буде перервано і запущено інший, що потребує доступу до того самого ресурсу то результат роботи програми буде не визначений і залежатиме від послідовності включення потоків. Такий стан називається «станом гонки» (race condition) [27].

Тому необхідним є механізм, що дозволить потокам координувати свою роботу — синхронізація потоків.

Найпростіший механізм синхронізації називається «критичними секціями». Він дозволяє виділити ділянки коду де необхідно отримати доступ до спільних ресурсів і гарантувати, що в кожен момент часу в кожній критичній секції буде не більше одного потоку.

Критична секція це об’єкт ядра операційної системи. Щоб її створити необхідно викликати функцію InitializeCriticalSection. Перед входом в ділянку коду, що має бути синхронізованою, необхідно викликати EnterCriticalSection, після виходу з неї LeaveCriticalSection.

В даному рішенні використовується дві критичні секції. Одна синхронізує відкрита сесій, щоб унеможливити їх повторне відкриття в іншому потоці. Цей прапорець встановлюється в методах OpenSession, CloseSession та SendReceive. Останній синхронізується щоб запобігти хаотичному запису з декількох потоків в буфер пристрою.

Інша синхронізація відбувається на рівні доступу до об’єкта поточного запиту. Якщо запиту не має то можна посилити новий, інакше спроба завершиться помилкою.

Власне створення потоку відбувається за допомогою системної функції **CreateThread**, інформація про яку може бути отримана із документації. Тут потрібно пам’ятати про те, що запускати в паралельному потоці можна лиши функцію, що не є членом класу. Для вирішення цієї проблеми можна скористатися так званою «дружною» функцією, тобто функцією, що може отримати доступ до прихованих членів класу, проте сама не є його членом. При запуску потоку буде передано вказівник на об’єкт класу USBDevice для взаємодії з ним.

3.2.4 Створення проекту.

Як було зазначено для створення драйверу використовується середовище розробки Microsoft Visual Studio. В якому при створенні та налаштуванні проекту можуть виникнути наступні труднощі.

Розташування файлів заголовків може не збігатись з заданим за замовчуванням. Щоб вказати ці шляхи необхідно у головному меню виконати: Проект – Властивості – Властивості конфігурації – C/C++ – Загальне – Додаткові каталоги включення. Щоб забезпечити легкий запуск проекту на різних комп’ютерах в цьому полі необхідно вказати саме відносний шлях до папки з файлами заголовків відносно кореневої папки проекту.

Далі необхідно встановити правильне кодування текстових даних. Для цього у головному меню виконати: Проект – Властивості – Властивості конфігурації – Загальне. У полі Набір знаків встановити значення «Використовувати багатобайтне кодування».

3.2.5 Результати

Створений драйвер пристрою представляє собою DLL бібліотеку, що бере на себе усю роботу з ініціалізації та закриття з’єднання, передачі та отримання даних. А керування пристроєм відбувається за допомогою кількох простих функцій, що наведені в таблиці 3.2.

Таблиця 3.2 — Функції бібліотеки драйвера

Функція	Опис
int GetMajorVersion()	Отримати старшу частину версії.
int GetMinorVersion()	Отримати молодшу частину версії.
int IsVersionConfirmed()	Чи співпадає версія драйвера з версією прошивки приладу (0 — ні).
int GetLastDeviceError()	Отримати код останньої помилки.
int GetPersentComplete()	Прогрес вимірювання в процентах.
int SetBuffer(SurfaceBuffer * buffer)	Передача вказівника на буфер, який потрібно заповнити даними. Виклик цієї функції розпочинає процес вимірювань.

3.3 Розробка модуля відображення

Окрім власне відображення графіки ця частина системи відповідає за відображення інтерфейсу користувача, отримання запитів від нього та оброку результатів. Розробка модуля на низькому рівні потребує багато часу та зусиль, тому можна скористатись одним з готових рішень. Ідеальне рішення повинно мати засоби для роботи з тривимірною графікою, мати можливість взаємодіяти з машинними бібліотеками, використовувати просту але функціонально розвинену мову програмування, бажано об’єктно-орієнтовану та мати засоби для побудови інтерфейсу користувача.

Подібні вимого часто пред’являються інструментам призначеним для розробки тривимірних ігор. Найбільш популярним таким інструментом є Unity3D від компанії Unity Technologies. Він представляє собою бібліотеку, що має всі необхідні для роботи компоненти, та графічний редактор для спрощення позиціонування елементів тривимірної сцени.

Додатки, що розроблені з використанням Unity3D можуть бути запущені на різних операційних системах, в тому числі і на мобільних. З огляду на те,

що апаратна частина лише одне з можливих джерел даних, а всі необхідні, для взаємодії з нею, алгоритми знаходяться у низькорівневій бібліотеці, що має досить вузький інтерфейс, можна легко створити мобільну версію графічного модуля програми.

Розробляти додатки на Unity3D можна використовуючи мови високого рівня C# або JavaScript (остання сильно відрізняється від версії для браузерів). Мова JavaScript є не дуже зручною для створення великих проектів тому зазвичай використовують саме C#. Це об’єктно-орієнтована мова програмування. Такий підхід є більш гнучким оскільки він дозволяє розділити функціонал програми між окремими класами. Кожен клас містить дані і функції призначені для рішення однієї конкретної підзадачі. Наприклад якщо клас використовує бібліотеку то він сам її завантажує, використовує та вивантажує коли це йому необхідно, так щоб іншим класам, які використовують перший не було необхідності знати про це. Все що відомо про клас це його інтерфейс, тобто набір функцій, які можна викликати у об’єкта цього класу, щоб змусити його робити деякі дії.

Іще однією особливістю мови C# є робота з так званим керованим кодом. Цей термін було введено корпорацію Microsoft по відношенню до їхньої системи .Net, що першою почала використання цієї мови. Він означає що після компіляції буде отримано не машинний код, а код для середовища виконання. Це програмний посередник між розроблюваною програмою та операційною системою, що в свою чергу дозоляє абстрагуватись від особливостей останньої і розробляти програми, що можуть бути запущені не різних операційних системах.

3.3.1 Взаємодія з бібліотекою драйверу пристрою

Реалізація взаємодії графічного модуля з приладом, перш за все потребує взаємодії з його драйвером, який, в свою чергу, організує реальний обмін даними з апаратною частиною. Драйвер, який описано раніше, розроблений у вигляді бібліотеки з використанням мови програмування C++

на основі некерованого (машинного) коду. Це може викликати деякі труднощі при взаємодії з керованим кодом мови C#.

На щастя Unity3D має засоби, що дозволять взаємодіяти з такою бібліотекою, через механізм так званих «Native plugging». Для цього необхідно додати файл бібліотеки (.dll) в проект, як звичайний файл ресурсів, але в особливу папку Plugins в корені проекту, оскільки там відбувається пошук всіх файлів бібліотек.

Щоб прив’язати метод який може бути викликаний з керованого коду до функції з бібліотеки обхідно визначити його як зовнішній статичний (*static extern*), тобто для виклику з бібліотеки функції *void Test()*, необхідно записати так: «*public static extern void Test();*». А щоб вказати з якої саме бібліотеки буде викликана функція потрібно використати спеціальний атрибут **DllImport** зі строковим параметром — ім’ям бібліотеки [28].

Цей спосіб використання бібліотеки дуже легкий і, як стверджують розробники, він працює на всіх платформах, що підтримуються. Щоб її підключити та використовувати функції необхідно всього дві стрічки коду на кожну, а додати до проекту можна простим перетягуванням в папку. Всю іншу роботу система зробить сама. Однак він має один досить суттєвий недолік, який полягає в тому що завантажена одного разу бібліотека вже не буде вивантажена аж до перезавантаження редактора Unity. Тобто вона буде зайнята, а файл захищено від модифікації. Іншими словами, щоб перебудувати бібліотеку необхідно перезапустити редактор.

Зважаючи на те, що перезавантаження редактора займає кілька хвилин, використання цього способу при розробці, коли перебудовувати бібліотеку доводиться постійно, є практично неможливим, навіть з усіма його перевагами.

Альтернативою є динамічне завантаження бібліотеки засобами операційної системи Windows, за допомогою системних функцій LoadLibrary, GetProcAddress та FreeLibrary, які були детально описані у попередньому підрозділі разом із алгоритмом завантаження.

Ці функції завантажуються за допомогою першого способу зі системної бібліотеки «kernel32.dll». Зважаючи на те, що kernel32.dll є частиною ядра операційної системи вона не буде перебудована, тому немає необхідності її вивантажувати, а значить використання першого способу є цілком виправданим [29].

Завантаження бібліотеки відбувається так само як і в мові C++, за винятком того, що замість вказівників на об’єкти використовується спеціальний тип **IntPtr**, який по суті є вказівником на область некерованої пам’яті.

Враховуючи це сигнатури функцій, як і перелік кроків, що необхідно виконати буде трохи змінений. Функція LoadLibrary приймає ім’я бібліотеки яку буде завантажено та повертає вказівник у форматі IntPtr. При відсутності вказаного файлу раніше функція порожня значення (NULL), однак згідно з правилами мови програмування змінній типу IntPtr не може присвоєне це значення. Тому для перевірки успішності операції необхідно порівнювати результат зі спеціальним значенням IntPtr.Zero, яке є еквівалентом порожнього вказівника [30].

Функція GetProcAddress, що використовується для пошуку адреси функції отримує на вхід вказівник на завантажену бібліотеку (IntPtr) і строковий параметр з ім’ям функції, для завантаження та повертає IntPtr вказівник на функцію. Однак в мові C# не можна викликати функцію через IntPtr вказівник. Для цього передбачений спеціальний тип делегат (Delegate) який може містити адресу методів та дозволяє їх викликати. Щоб конвертувати IntPtr вказівник в делегат слід використати метод **Marshal.GetDelegateForFunctionPointer**, який приймає вказівник та тип (об’єкт класу Type) делегату та повертає його, щойно створений об’єкт.

Функція вивантаження (FreeLibrary) змінила лише тип параметра-вказівника, а в іншому залишилась без змін [31] [32].

Залишилось згадати лише про одну особливість такого використання бібліотеки, яка полягає в тому, що абсолютний шлях до неї не повинен

містити кириличних символів. В іншому випадку можуть виникнути проблеми з кодуванням і завантаження не відбудеться.

3.3.2 Обмін даними між драйвером та модулем відображення

При вимірюванні характеристик виникає необхідність передачі значних об’ємів даних. Механізм передачі потребує окремого розгляду оскільки така передача може зайняти багато часу, що неодмінно позначиться на швидкості реакції всієї програми. Річ у тім, що графічна частина системи працює в одному потоці, тобто всі дії виконуються по черзі в так званому головному циклі, а якщо якась операція триватиме значний час (навіть секундна затримка буде досить помітною) то програма весь цей час не реагуватиме на дії користувача, більше того буде схоже, що вона «зависла».

Не зважаючи на те, що найбільші затрати часу спричиняє саме передача даних між апаратною частиною та комп’ютером, даний пункт присвячений обміну даними між керованим кодом Unity на мові C# і некерованим кодом бібліотеки, що написана з використанням мови C++.

Як уже сказано найпростіший варіант — виклик функції з бібліотеки, що очікує отримання даний від пристрою та повертає весь масив. У випадку якщо всі дані ще не прийнято функція може повернути порожнє значення (NULL). Він має багато недоліків, яких можна віднести утруднення контролю пам’яті на гранці між керованим та некерованим кодом, особливо якщо взяти до уваги те, що у некерованому коді функція може повернути масив лише у вигляді вказівника на його перший елемент, а в керованому коді робота з вказівниками має деякі складнощі. Другий недолік — складність встановлення розмірів отриманих даних.

Щоб пришвидшити відгук програми при передачі інформації можна розбити масив даних на менші частини. Однак і це не вирішує граничних проблем з виділенням та звільненням некерованої пам’яті та потребує окремої передачі розміру переданої частини масиву.

З огляду на недоліки було прийнято рішення відмовитись від прямого повернення масивів даних з некерованого коду.

Щоб уникнути очікування прийняття всіх даних можна використати механізм обробки подій, що заснований на застосуванні функцій зворотного виклику — тепер функція керованого коду буде викликана з некерованого.

При відправленні запиту на отримання масиву даних до некерованого коду драйверу передається вказівник на функцію, що буде викликана після завершення отримання даних від апаратної частини. Іншими словами відбудеться подія звершення прийняття і всі модулі що «підписані» на неї отримають (в даному випадку це лише графічний модуль) повідомлення і зможуть обробити цю подію. Проте, прийняття даних відбувається в додатковому потоці, і очевидно, функція-обробник буде викликана в тому ж таки додатковому потоці. Тобто окрім синхронізації потоків на рівні драйвера необхідна додаткова синхронізація на рівні керованого коду [33].

Найбільш оптимальне рішення полягає спільному використанні пам’яті до якої мають доступ обидва модулі програми. Некерований модуль не може отримати доступ до керованої пам’яті. А в мові C# є всі необхідні засоби для роботи з некерованою пам’яттю яку використовує C++.

Спільно використовувати можна будь-яку структуру даних, що може включати в себе масив з даними, його розміри та, за необхідності, іншу інформацію.

Для реалізації такого обміну необхідно створити однакові структури на C++ та C#. У випадку останньої використовуються спеціальні атрибути, що допоможуть правильно розташувати поля керованої структури. В некерованій бібліотеці створити функцію що отримує вказівник на цю структуру(SetBuffer), зберігає його та записує туди дані по мірі їх отримання від приладу та функцію, що показує чи завершена передача даних. В цьому випадку використовується функція GetPersentComplete, яка повертає прогрес вимірювання у відсотках.

Для початку процедури отримання даних потрібно виконати наступні кроки. Спершу потрібно створити об’єкт керованої структури. Потім виділити некеровану пам’ять під структуру та скопіювати туди щойно

створений об’єкт цієї структури. Після копіювання буде отриманий вказівник на цю область пам’яті, який треба передати до бібліотеки викликом відповідної функції, що також означатиме початок очікування передачі.

Далі на кожній ітерації основного циклу необхідно перевіряти чи вже отримано всі дані. Після того як це станеться можна отримати новий об’єкт керованої структури з цієї області некерованої пам’яті та вивільнити її.

Основною перевагою цього варіанту передачі є те, що необхідна синхронізація потоків лише на ріні драйверу. Іншою превагою є необхідність лише двох функцій для організації такої передачі та прозора робота з пам’яттю [34].

3.3.3 Взаємодія з драйвером приладу

За взаємодію з драйвером відповідає клас USBDevice. Він є об’єктним відображенням підключеного пристрою з яким можна обмінюватись інформацією, тобто до якого можна підключитись, послати запит та отримати відповідь.

Даний клас є абстрактним тобто не представляє конкретного пристрою і має лише функціонал доступний для будь-якого USB пристрою, що може бути розроблений за подібною схемою. Додавати реалізацію функціоналу для конкретного пристрою можна за допомогою механізму наслідування.

Ініціалізація пристрою проходить в два етапи. На першому відбувається динамічне завантаження бібліотеки драйвера та проводиться пошук в ній необхідних для роботи функцій, за допомогою функцій kernel32.dll. На другому проводиться пошук підключених до USB-порту пристроїв. В разі якщо файл бібліотеки, одну або більше функцій з неї не знайдено або немає підключених пристроїв буде отримано повідомлення про помилку та буде завершено процес вимірювань, а всі спроби послати повідомлення будуть завершуватись з помилкою. Слід зауважити, що це не вплине на роботу системи відображення, оскільки прилад є лише одним з можливих джерел даних.

Керований клас USBDevice не генерує помилок, а лише відображує ті, що отримані від драйвера. Для отримання інформації про помилку можна звернутись до таблиці 3.1.

Конкретний клас TracerDevice є об’єктно-орієнтованим представленням приладу характерографа та має набір методів для взаємодії з конкретним типом пристрою. Більше того він може працювати лише з пристроєм прошивка якого має відповідну йому версію.

3.3.4 Вибір способу побудови поверхні

Графік функції двох змінних відображається у вигляді поверхні. Вихідні дані для побудови такої поверхні зручно зберігати у вигляді так званої карти висот (Heightmap). Вона представляє собою двовимірну матрицю, довільної розмірності. Кожен її елемент є значенням висоти поверхні в даній точці. Тобто вона є представленням масиву тривимірних векторів, в яких координати x та y пропорційні індексам елемента i та j по горизонталі та вертикалі відповідно, а координата z дорівнює значенню цього елементу — висоті. Між кожними чотирма сусідніми точками, що утворюють квадрат будуються по два трикутники, з яких і буде складена вся поверхня.

В термінах комп’ютерної графіки ці трикутники називаються полігонами (Polygon), на основі яких будуються всі тривимірні фігури. Зазвичай полігони мають трикутну форму, оскільки три це найменша кількість точок необхідна для побудови площини. Крім того з курсу елементарної математики відомо, що площину не можна провести більше ніж через три довільні точки.

Як і більшість сучасних засобів відображення тривимірної графіки, Unity зазвичай використовує готові тривимірні моделі, які попередньо завантажені з файлу.

З огляду на це для побудови поверхні можна було б використати заздалегідь заготовлені полігони у вигляді окремих моделей. Проте, перепад висоти між сусідніми точками може сильно відрізнятись, а отже різними будуть розміри та форма трикутників.

Впоратись з цією ситуацією можна двома шляхами. Або підготувати багато полігонів для всіх можливих випадків, що неодмінно призведе до невиправданих витрат пам’яті. Або написати досить складний алгоритм масштабування кожного полігону в залежності від необхідних розмірів, що неодмінно призведе до появи помилок. З цього можна зробити висновок, що дане рішення не є оптимальним.

Знаючи, що карти висот часто використовуються для моделювання ландшафтів місцевості, розробники включили в стандартну поставки спеціальний інструмент Terrain, що дозволяє динамічно створювати ландшафти прямо редакторі, або навіть під час роботи програми. Він містить досить розвинений функціонал наприклад для пошуку точок перетину інших об’єктів з поверхнею ландшафту, або динамічного нанесення текстур. Це все може призвести до збільшення затрат ресурсів.

З огляду на все це найбільш оптимальним способом для відображення поверхні є використання низькорівневих засобів побудови графічних примітивів, що майже напряму використовують засоби графічних адаптерів.

3.3.5 Побудови тривимірного примітиву

Звичайний опис тривимірних моделей складається зі структури даних до якої входять наступні.

Масив вершин містить координати точок з координатами (x, y, z), які є вершинами полігонів-трикутників.

Масив нормалей. Нормалі зазвичай використовуються для розрахунку освітленості поверхні — чим більший кут між вектором нормалі та вектором освітлення тим більш яскраво буде освітлений полігон. Якщо є вектори з координатами (x_1, y_1, z_1) та (x_2, y_2, z_2), то кут між ними може бути знайдений за наступною формулою:

$$\cos\phi = \frac{x_1x_2 + y_2y_2 + z_1z_2}{\sqrt{x_1^2 + y_1^2 + z_1^2} \cdot \sqrt{x_2^2 + y_2^2 + z_2^2}}.$$

Скалярний добуток цих векторів розділений на добуток їх довжин. Для спрощення розрахунків у процесі побудови зображення на екрані, нормалі зазвичай мають одиничну довжину. Очевидно, що нижню частину можна відкинути та отримати досить просту формулу:

$$\cos\phi = x_1x_2 + y_2y_2 + z_1z_2 .$$

Нормалі завжди рахуються до побудови зображення, щоб зекономити ресурси комп'ютера [35].

Масив координат текстури, або так званий uv-масив. Він містить відносні (x, y) координати текстури, що можуть приймати значення від 0 до 1. Вектор (0, 0) відповідає лівому нижньому куту зображення текстури, а (1, 1) верхньому правому. На рисунку 3.3 зображено три точки з відповідними координатами (x1, y1), (x2, y2), (x3, y3). Заштрихована трикутна область буде «натягнена» на полігон у вершинах якового вказані відповідні uv-координати.

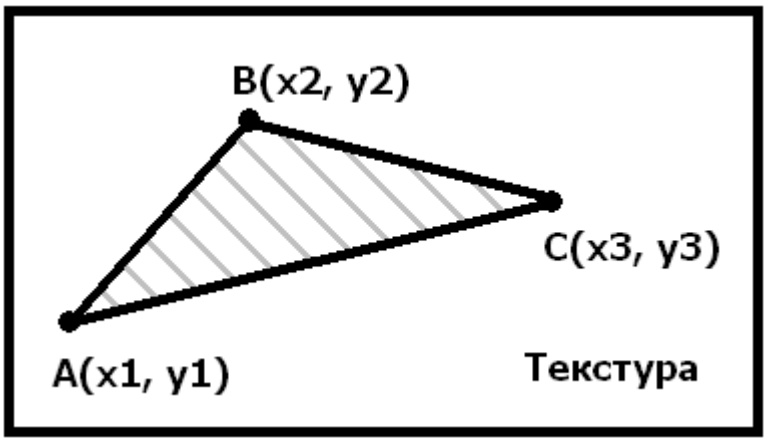

Рисунок 3.3 — Координати текстури

Масив кольорів. Для кожної вершини задає колір зазвичай в 24 бітному форматі, хоча формат кольору може змінюватись залежно від системи. Колір вказується для кожної вершини окремо, та буде поступово змінюватись при переході між точками з різними кольорами.

Масив індексів. На відміну від усіх вищезгаданих масивів, де кожний елемент відповідає одній точці, даний масив має три елементи для кожного трикутника. Наприклад для побудови квадрату (рис 3.4) необхідно 2

трикутники, тобто шість вершин. Однак при цьому можна отримати надлишкові дані, пов’язані з тим, що дві вершини мають однакові координати (B1 = B2, D1 = D2). Інший спосіб передбачає використання 4 вершин, однак з’являється необхідність використання деяких засобів, що дозволять однозначно встановити до яких трикутників належить вершина.

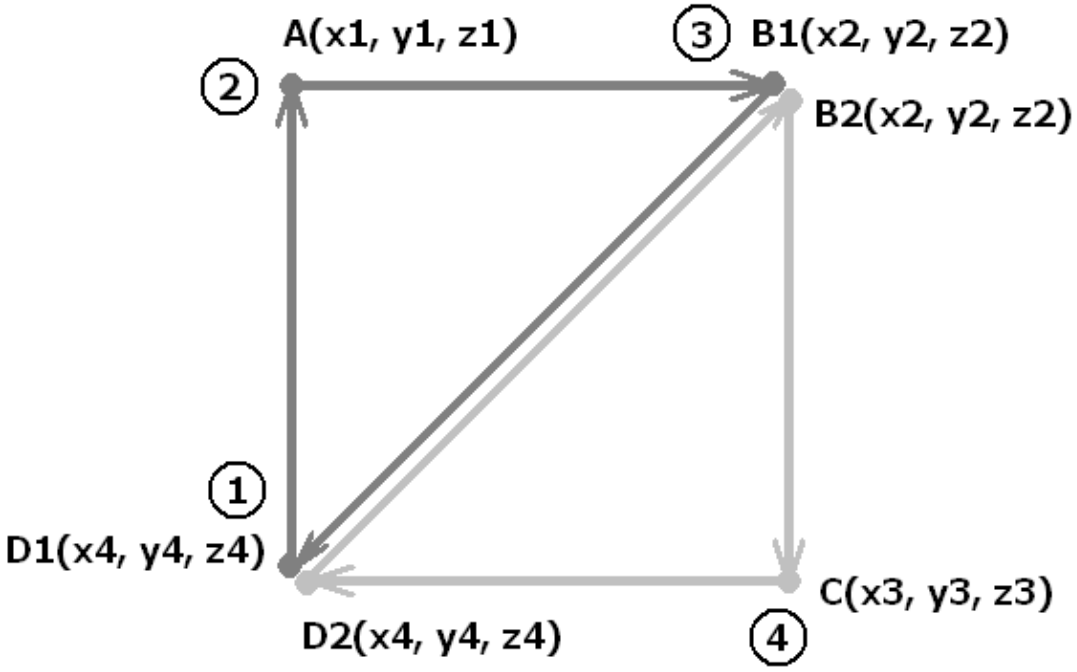

Рисунок 3.4 — Побудова квадрату за допомогою індексів

Кожен елемент масиву індексів це номер вершини в масиві вершин (цифри у кружечках). За кожними трьома елементами в масиві індексів буде побудований один трикутник (індекси 1, 2, 3 для першого і 1, 3, 4 для другого трикутника). Тут існує одна особливість: всі трикутники необхідно описувати за годинниковою стрілкою, якщо дивитись з камери (рис 3.4). Справа в тому, що це дозволяє відображати лише ті полігони, що «дивляться» у камеру. Трикутники описані за стрілками будуть видимі у такому положенні, а якщо описати їх в зворотному порядку то вони будуть видмі лише з іншого боку.

Інша особливість використання індексів полягає в тому, що нормалі вказуються для кожної точки і при використанні однієї точки в декількох полігонах вони обидва матимуть лише одну нормаль. У випадку з квадратом

цього не помітно, однак якщо між площинами нормалей є значний кут може виникнути несподіваний ефект освітлення (рис 3.5).

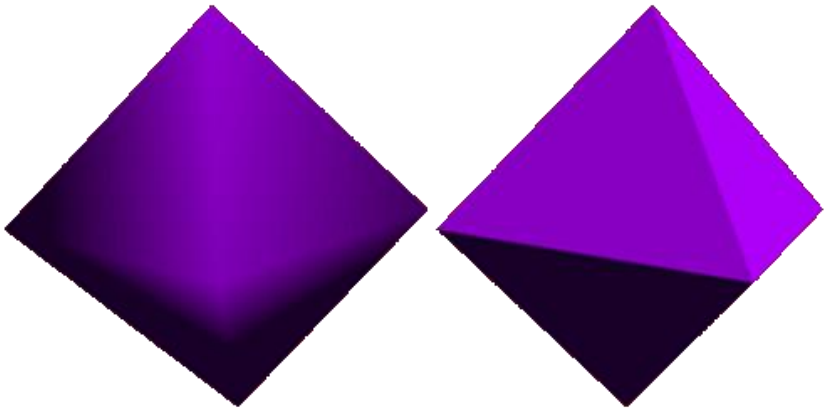

Рисунок 3.5 — Спільні та окремі вектори нормалей

Зліва (рис 3.5) зображено октаедр зі спільними вершинами, з права — для кожного трикутника вказано власні координати вершин, тобто деякі дублюються. У другому випадку нормалі кожного полігону перпендикулярні площині цього полігону, тому добре видно чіткі грані. У першому — вказано одну нормаль і рендеру доводиться інтерполювати нормалі для кожної точки. Це призводить то ефекту візуального згладжування різких переходів і зазвичай використовується для сфер або поверхонь з плавним переходом [36].

3.3.6 Алгоритм побудови поверхні

Для побудови та відображення поверхні створено два класи Surface та Plotter.

Перший призначено для зберігання та видачі в потрібній формі інформації про поверхню. Його основою є двовимірний масив, в якому зберігаються дані, для побудови карти висот, розміри масиву, масштаб та посилання на об’єкт Mesh, що може бути побудований на основі цієї поверхні. Також клас містить набір методів, які спрощують додавання, редагування даних та звернення до елементів за допомогою одного або двох індексів. Тобто в залежності від ситуації поверхня може бути представлена

як одновимірний або двовимірний масив. Це необхідно для спрощення генерації окремих масивів тривимірної моделі.

Другий клас Plotter виконує власне побудову моделі на онові об’єкту поверхні. Основний його метод Generate, отримує в якості параметру Surface, та повертає тривимірне зображення — Mesh. Цей метод є основним тому його алгоритм слід розглянути детальніше.

Преш за все необхідно визначити розміри кожного масиву (vertices, normals, colors, triangles) та виділити для них пам’ять. Нормалі та кольори встановлюються для кожної вершини, а значить розміри кожного масиву дорівнюють кількості елементів у вихідній матриці.

На другому етапі відбувається заповнення масивів вершин, кольорів та нормалей. В цьому випадку зручно представити поверхню як одновимірний масив та послідовно обробляти дані кожної вершини: створювати вектор, розраховувати нормаль та колір. Для більшої наочності відображення висота кожної точки підкреслюється її кольором — чим вище точка тим яскравіший колір. Значення кольору нормується за максимальною висотою точки на поверхні.

Заключним етапом є заповнення масиву індексів, кількість елементів якого розраховується за наступною формулою:

$$S = 12(N-1)(M-1),$$

де S — сумарна кількість елементів масиву індексів, N та M ширина і висота масиву відповідно (рис 3.6). Для кожної точки з індексами (i, j) будуються два трикутники для кожного квадрату, що утворений точками (i, j), (i+1, j), (i+1, j+1), (i, j+1). Щоб поверхню було видно згори та знизу необхідно побудувати два такі квадрати. Два трикутники згори, два знизу, по три точки на кожний, всього 12 значень для кожної точки.

На відміну від вихідного масиву індекси розташовуються послідовно, в одновимірному масиву, тому для кожної точки потрібно розрахувати її порядковий номер:

$$n = jN + i,$$

де n — номер індексу в масиві, N — горизонтальна розмірність вихідної матриці, i, j — індекси поточної точки.

Індекс наступної точки ліворуч буде n+1, а наступної точки вгору буде зміщений на розмір рядка, тобто n+N+1. З цього неважко зробити висновок в якій послідовності слід записувати точки. В кінці функції готова модель буде повернута в точку виклику і її можна помістити в спеціальний компонент для відображення.

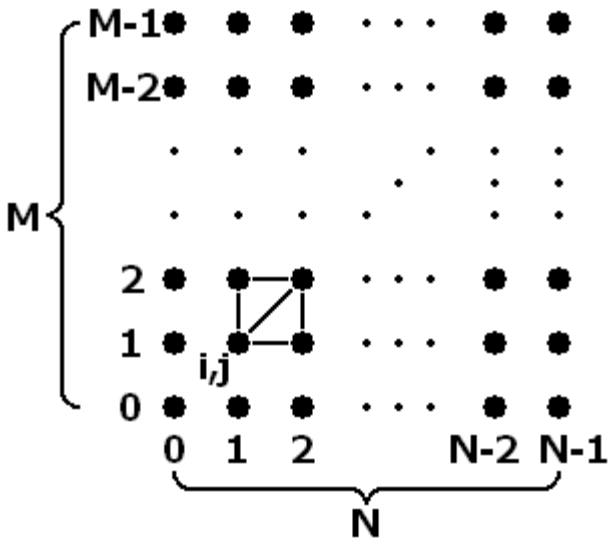

Рисунок 3.6 — Побудова полігонів поверхні

Для більшої наочності та полегшення сприйняття тривимірна поверхня буде підсвічена різними кольорами в залежності від висоти. В даному випадку використано перехід кольору, який нормується по найбільшому перепаду висоти поверхні, від зеленого, в найнижчій точці поверхні, до червоного — в найвищій. Для реалізації цього потрібно щоб формат структури даних вершин підтримував роботу з кольором. Зазвичай цього не потрібно, оскільки дуже часто модель повинна мати не просто колір а й рисунок — текстуру. За обробку вершин відповідають шейдери — програми, що виконуються безпосередньо відеоадаптерами. Тому для відображення кольорів було використано специфічний, хоч і дуже простий шейдер [37].

3.4 Інтерфейс користувача та використання програми

Для роботи з системою необхідно підключити прилад до USB порту комп’ютера та запустити програму для відображення графіки. Програма сама виконає завантаження драйвера та ініціалізацію приладу.

Після завантаження буде відображено головне вікно програми. Спершу воно буде порожне, оскільки жодних даних ще не завантажено. На рисунку 3.7, показано вигляд вікна програми із уже завантаженими тестовими даними. Для тесту було взято характеристики транзистора ГТ309.

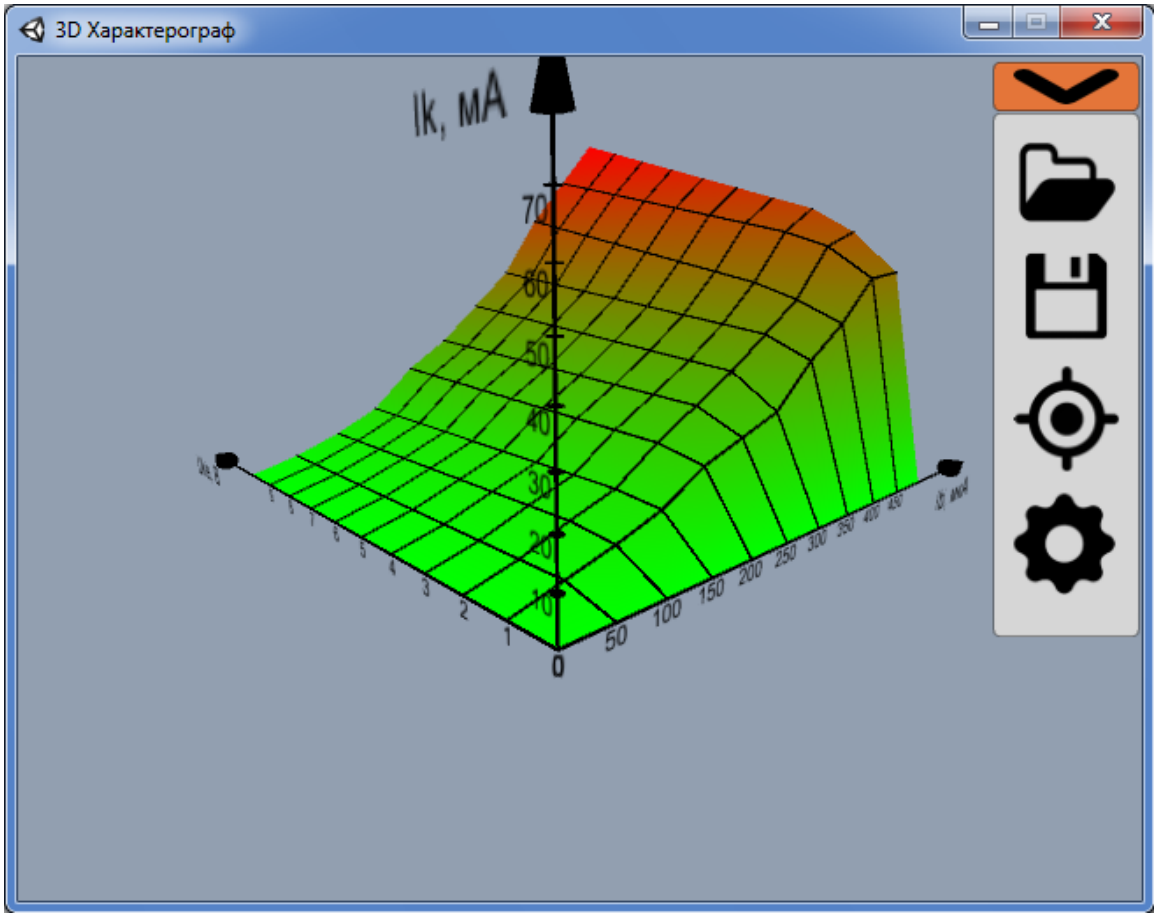

Рисунок 3.7 — Головне вікно програми

Це так званий режим огляду — основний режим який призначено для відображення готових характеристик. В цьому режимі можна обертати характеристику за допомогою правої клавіші миші та масштабувати її за допомогою колеса миші.

У верхньому правому кутку вікна знаходиться кнопка для відкриття та закриття головного меню. Це меню реалізовано у вигляді випадаючого

списку, що меню знаходиться в правій частині вікна (рис 3.7) та має наступні пункти:

– Відкрити. Дозволяє виконати завантаження даних у програму;
– Зберегти. Призначено для збереження даних у файл;
– Маркер. Вмикає або вимикає режим маркера;
– Налаштування. Відкриває вікно з налаштуваннями програми.

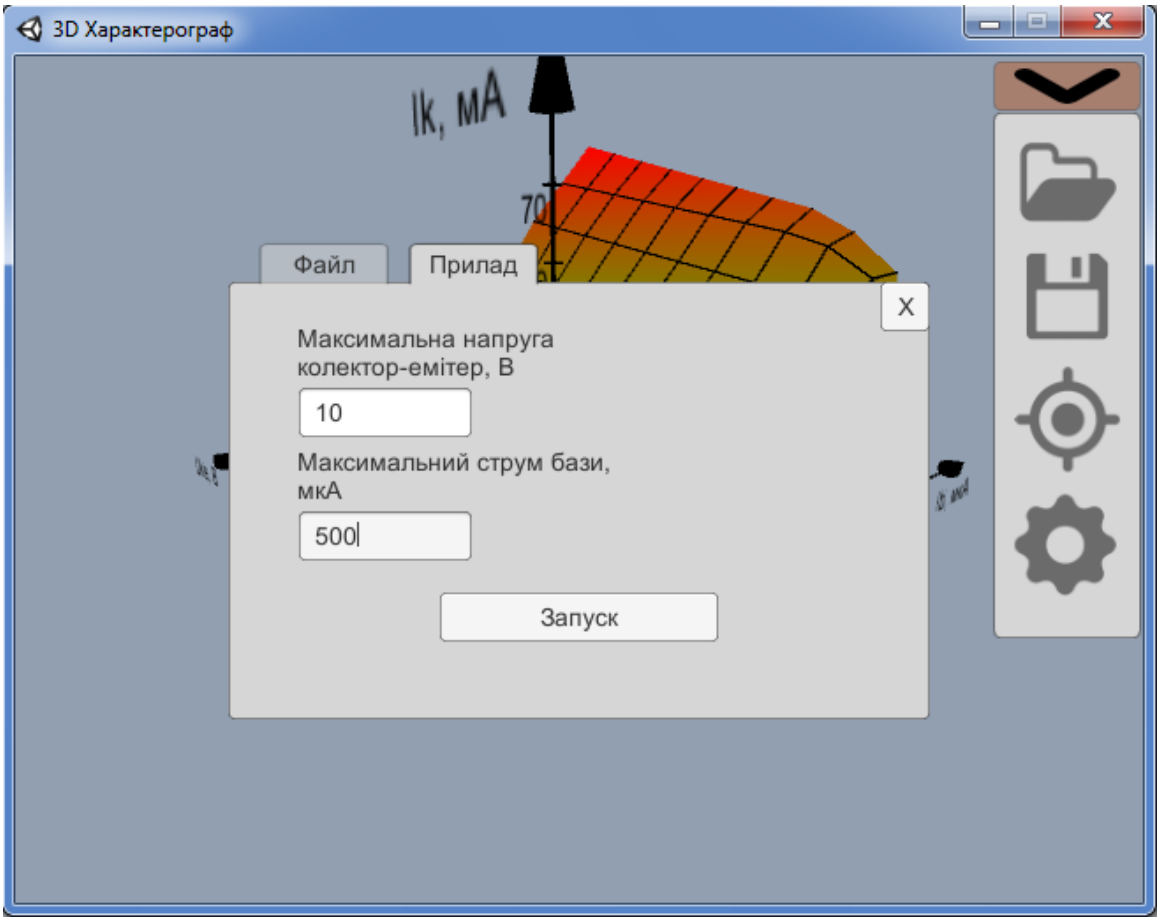

Рисунок 3.8 — Діалог роботи з приладом

Пункти не мають підписів, однк вони позначені іконками, що однозначно вказують на їх призначення. При відкритті кожного меню всі функції та кнопки попереднього меню будуть заблоковані, і відображені менш яскравим кольором.

Щоб розпочати роботу з програмою потрібно вибрати в головному меню пункт «Відкрити», після чого з’явиться діалгове меню що показане на рисунку 3.8.

Це меню має дві вкладки «Файл» та «Прилад», для отримання даних відповідно з файлу або з приладу. Слід зауважити, що архітектура програми розроблена таким чином, що апаратна частина є лише одним з можливих джерел даних. Це зроблено для того, щоб можна було додавати нові джерела даних, та способи збереження. Наприклад планувалось додати можливість зберігати дані на сервер та завантажувати їх з нього. Це дозволило б використовувати підсистему відображення на інших операційних системах, наприклад на мобільних. Засоби використані для її розробки дозволяють це зробити, однак підключення апаратної частини може викликати значні труднощі.

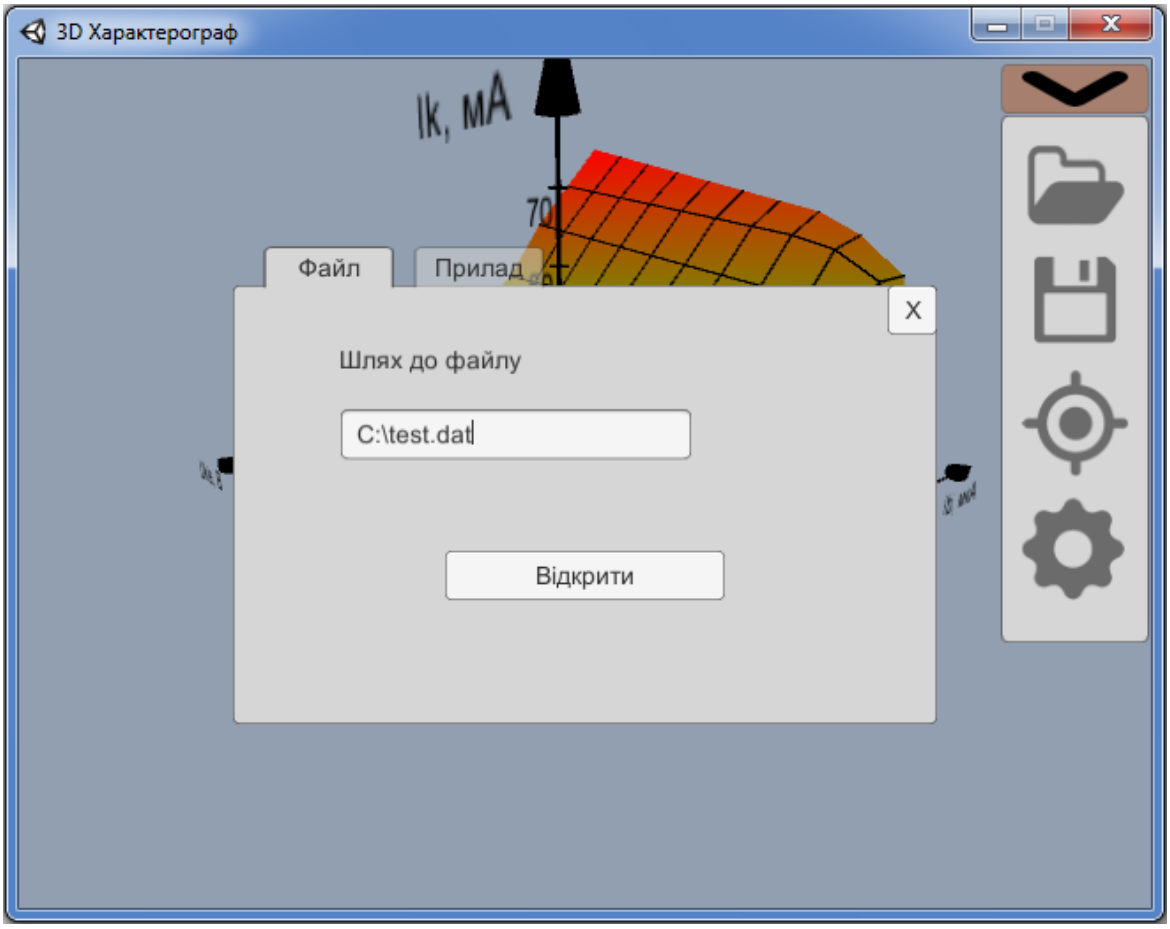

Рисунок 3.9 — Діалог відкриття файлу

Щоб виміряти характеристику потрібно перейти на вкладку «Прилад» та вказати максимальні безпечні значення напруги колектор-емітер та струму бази у відповідні поля. Це зроблено для того, щоб захистити транзистор від надмірного навантаження під час вимірювань. Після введення відповідних

даних кнопка «Запуск» стане доступною. Щоб розпочати власне процес вимірювання необхідно натиснути цю кнопку, після чого вона зникне і з’явиться шкали з прогресом процесу вимірювання у відсотках.

Коли значення шкали дійде до 100% меню завантаження буде закрито та повернуто в режим огляду.

Другий спосіб отримання даних в програму це завантаження їх з файлу (рис 3.9). Для цього потрібно відкрити меню завантаження та перейти на вкладку «Файл» та у відповідному полі вказати повне ім’я файлу, що обов’язково має включати його адресу, який необхідно відкрити, та натиснути копку «Відкрити». Файл буде відкрито і автоматично побудована характеристика, після чого меню буде закрито та повернуто в режим огляду.

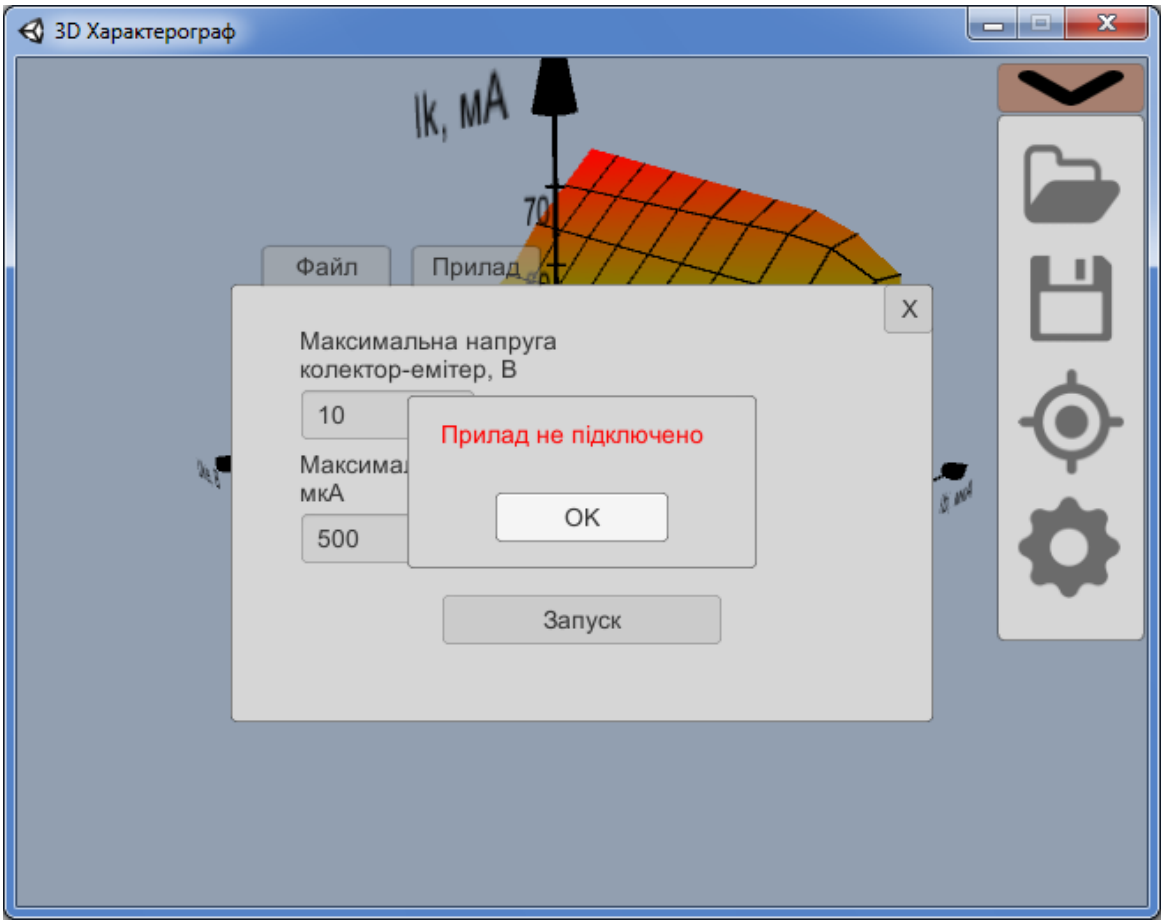

Рисунок 3.10 — Повідомлення про помилку

Якщо такого файлу не існує буде виведено повідомлення «Такого файлу не існує», після чого можна змінити ім’я файлу та спробувати знову.

Наступний пункт меню «Зберегти» веде до відповідного меню збереження. Воно має лише одну вкладку «Файл», оскільки прилад не призначено для збереження даних. Вигляд цього вікна аналогічний вигляду вікна завантаження з файлу (рис 3.9), крім того, що в цьому випадку буде відображено кнопку «Зберегти».

Для збереження необхідно вказати повне ім’я файлу, що включає його адресу на диску, та натиснути цю кнопку. Відповідний файл буде перезаписаний, і відновити його попередній зміст буде неможливо. Якщо такого файлу не існує то перед записом він буде створений.

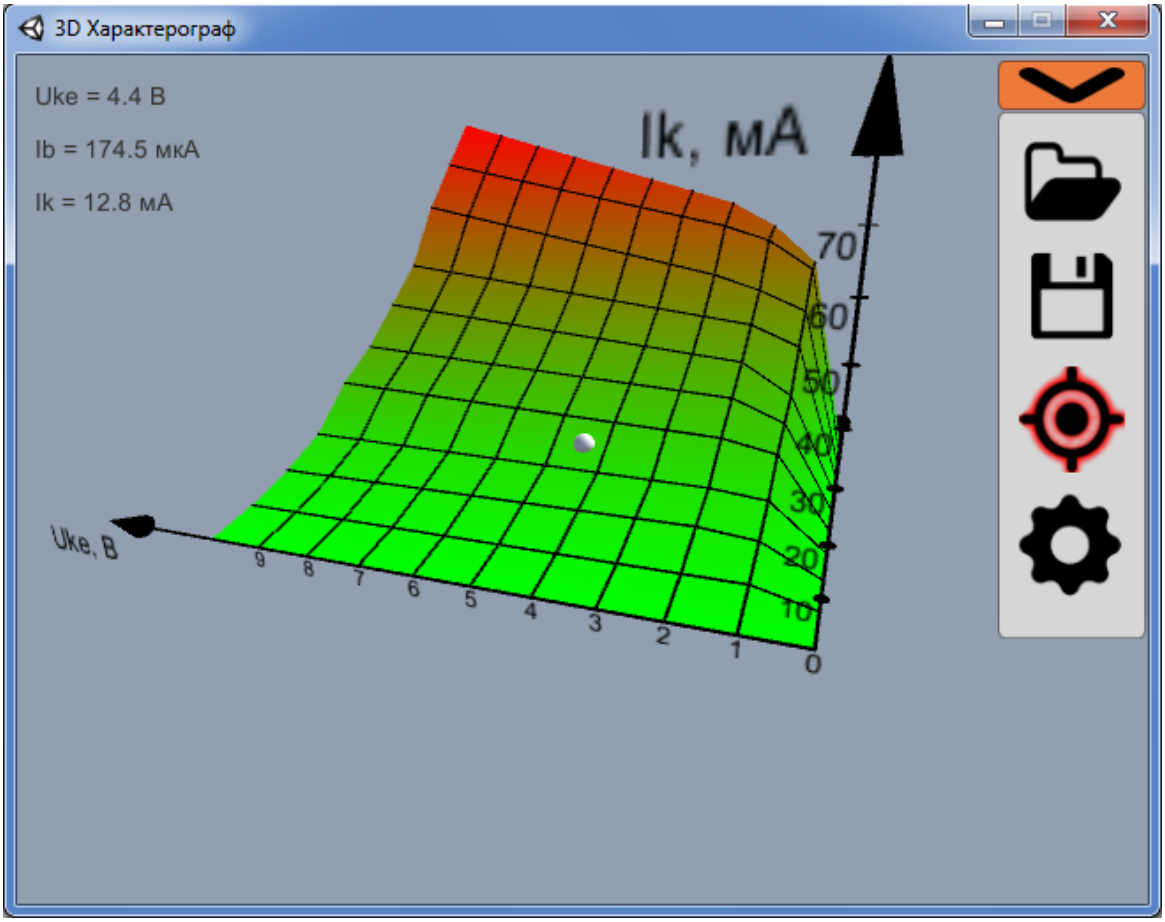

Рисунок 3.11 — Режим маркера

Якщо при запуску вимірювання сталась помилка то буде показано повідомлення з відповідним текстом (рис 3.10). На відміну від бібліотеки драйвера, яка у разі помилки повертає числовий код, відмінний від нуля, дана програма розшифровує цей код та виводить текс помилки. Окрім розшифрування стандартних помилок драйвера можуть виникнути і інші.

Наприклад може не бути знайдено файл бібліотеки цього драйвера в результаті чого буде отримано помилку «Файл бібліотеки CurveTracer3DDriver.dll не знайдено». Це дуже гнучкий спосіб відображення повідомлень, оскільки він може бути викликаний з будь-якого місця програми.

Третій пункт головного меню вмикає або вимикає режим маркера (рис 3.11). Цей режим дозволяє відслідковувати значення параметрів (струм бази, напругу колектор-емітер та струм колектора) у заданій точці. Для встановлення маркера в потрібну точку клацнути лівою кнопкою миші по поверхні характеристики. Вибрана точка відображається за допомогою спеціальної позначки — білої кульки. Числові значення параметрів точки будуть відображені у верхньому правому кутку вікна. Їх наявність також можна сприймати, як індикацію стану даного режиму.

При ввімкненні режиму маркера іконка в головному меню буде підсвічуватись червоним.

Останній пункт меню — налаштування. При виборі даного пункту буде відкрито меню, що містить різні налаштування програми для відображення графіки. Наприклад там є пункти, що дозволяють ввімкнути або вимкнути відображення осей, підписів та іншого.

Після завершення роботи програма сама виконає всі необхідні дії, зокрема очищення ресурсів, відключення приладу та вивантаження бібліотек. Важливо лише слідкувати за тим, щоб кожного разу відбувалось коректне завершення роботи.

3.5 Симуляція роботи системи

Згідно технічного завдання для підтвердження працездатності запропонованого рішення потрібно провести симуляцію симуляцію роботи схеми у середовищі моделювання, яке дозволяє їй взаємодіяти з програмною частиною.

Для проведення такої симуляції необхідно встановити та спеціальним чином налаштувати декілька програмних продуктів, кожен з яких відповідає за використання окремої частини системи:

- Середовище схемотехнічного моделювання Proteus ISIS в якому відбувається симуляція схеми апаратної частини приладу та розширення Proteus Virtual USB для організації взаємодії схеми з програмною частиною через USB.
- Інтегроване середовище MPLAB з компілятором C18 для компіляції програми керування мікроконтролером.
- Для розробки драйвера схеми на мові C++ використано Microsoft Visual Studio.
- Unity3D для роботи з тривимірною графікою.

Система схемотехнічного моделювання **Proteus** базується на основі моделей електронних компонентів, прийнятих в PSpice. Відмінною рисою пакета PROTEUS VSM є можливість моделювання роботи програмованих пристроїв: мікроконтролерів, мікропроцесорів, DSP та інших. Бібліотека компонентів містить довідкові дані. Додатково в пакет PROTEUS VSM входить система проектування друкованих плат. Пакет Proteus складається з двох частин, двох підпрограм: ISIS — програма синтезу та моделювання безпосередньо електронних схем і ARES — програма розробки друкованих плат [38].

Схему принципову розроблено та налагоджено з використанням Proteus 7.6 фірми Labcenter Electronics.

Для моделювання передачі даних через USB необхідно використовувати Proteus Virtual USB.

Основна мета USB моделювання Proteus VSM виконувати повне моделювання мікроконтролерів, які мають периферійний USB модуль, зокрема до моделювання наступних класів USB пристроїв:

- Запам’ятовуючі пристрої (Mass Storage Device Class, MSD);
- Пристрої введення (Human Interface Device, HID);

– Пристрої зв’язку (Communications Device Class, CDC);

Підтримка додаткових класів (і додаткових варіантів мікроконтролерів) можуть бути додані в більш пізніх версіях в залежності від попиту [39].

Розширення Virtual USB не встановлюється разом з основним пакетом програм Proteus. Для цього необхідно запустити Installer.exe з папки USB Drivers, що знаходиться у папці зі встановленим програмним забезпеченням Proteus. Або викликати Пуск – Proteus 7 Professional – Virtual USB – Install USB Drivers. Якщо цього не зробити то середовище видасть попередження про те, що драйвер не встановлено.

При першому віртуальному підключенні пристрою до комп'ютера він запросить драйвер як і при реальному з'єднанні. Необхідно вказати шлях до папки з драйвером. При встановленні на Windows 7 можуть виникнути труднощі з встановленням, тому необхідно встановити драйвер через «Диспетчер пристроїв».

Також в операційній системі Windows 7 скомпільовану програму драйвер необхідно запускати з правами адміністратора, інакше можуть виникнути труднощі, що пов’язані з неможливістю встановлення з’єднання з пристроєм.

Слід пам’ятати, що файл схеми Proteus не зберігає в собі файл прошивки, а містить лише абсолютний шлях до нього, тому при відкритті схеми на іншому комп’ютері можна отримати помилку, пов’язану з відсутністю файлу за вказаним шляхом.

Для розробки програми прошивки для мікроконтролера з використанням мови високого рівня необхідно спеціальне середовище розробки. Таким середовищем є **MPLAB** фірми Microchip. Він має набір інструментів для розробки та налагодження програмного коду для керування мікроконтролерами, такі як компілятори, лінкувальники, засоби для слідкування за ходом виконання коду. Для кожної серії мікроконтролерів необхідно використовувати різні компілятори, тому MPLAB створений з

окремих модулів, що дозволяє встановлювати різні інструменти, залежно від ситуації.

Для розробки даного проекту використано MPLAB 8.46, всі описані дії стосуються саме цього середовища. При відкритті проектів можуть виникнути наступні помилки.

Найбільш поширеною помилкою при використані MPLAB є розміщення проектів у папках, шлях до яких які містять кириличні символи. При намаганні відкрити файли з таких проектів MPLAB видасть помилку: «Вказаний шлях до файлу не може бути знайдений». Про це слід завжди пам’ятати, зважаючи на те що це повідомлення є малоінформативним.

Інша можлива проблема полягає в тому, що MPLAB зберігає абсолютні шлях до заготовочних файлі. В найпростішому випадку в проекті використовуються файли заголовків, що постачаються з компілятором та файли, що знаходяться в самому проекті. Розташування обох можуть відрізнятись на різних комп’ютерах тому необхідно переконатись, що в проекті вказані правильні шляхи.

Для цього необхідно клацнути правою кнопкою миші на проекті та вибрати пункт меню «Build options...», після цього відкриється вікно «Build options» (рис. 3.12). У ньому необхідно вибрати вкладку «Directories». У полі «Show directories for:» вибрати значення «Include Search Path». Натиснути кнопку «New» та вказати шляхи до папки проекту та до файлів заголовків в папці з компілятором.

Також можуть виникнути проблеми пов’язані зі зміною версії компілятора. Часто можна отримати помилки пов’язані зі зміною конфігураційних констант. Наприклад, константа FCMEM в більш пізній версії замінена на FCMEN.

При необхідності вказати шляхи до компілятора лінкувальника та інших інструментів. Для цього необхідно клацнути правою кнопкою миші на проекті та вибрати пункт меню «Select Language Toolsuite...»

Для відкриття проекту необхідно двічі клацнути на файл з розширенням .mcp або .mcw, в іншому випадку, якщо відкривати файл меню MPLAB можуть виникнути труднощі.

Для написання драйверів зазвичай використовують мову відносно низького рівня таку як C++.

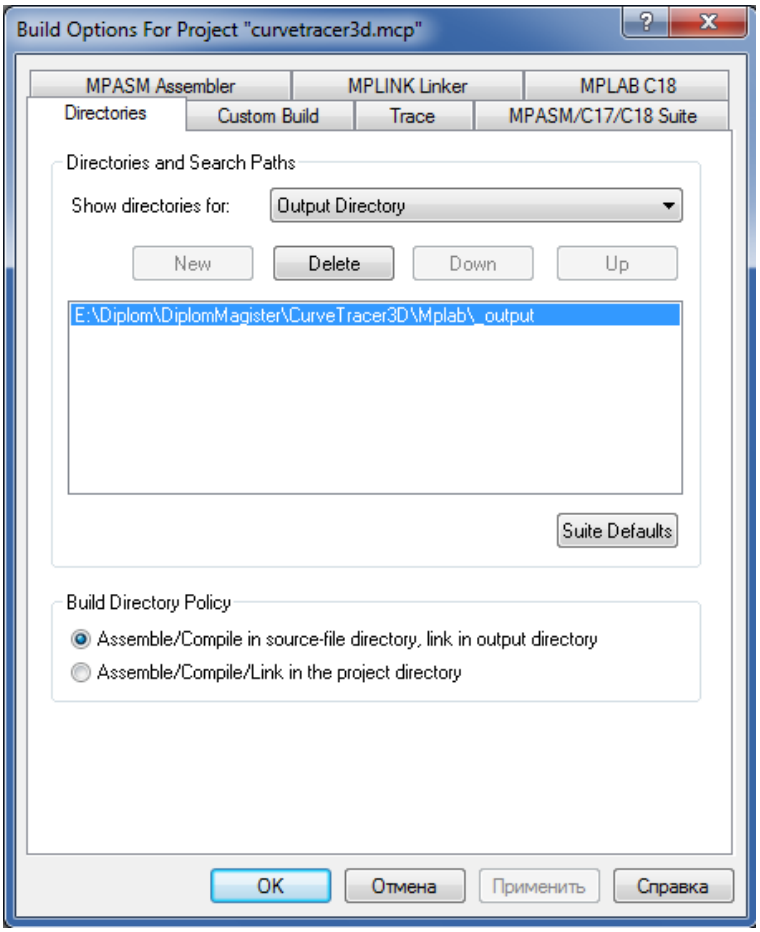

Рисунок 3.12 — Вікно «Build options»

З огляду не те, що операційна система Windows є найбільш популярною серед користувачів персональних комп'ютерів, а написання програм драйверів потребує взаємодії з операційною системою на низькому рівні інші операційні системи, для керування приладом, не розглядались. А отже для розробки драйвера керування приладом доречно використати середовище від того ж виробника що і операційна система — **Microsoft Visual Studio**.

Налаштування Visual Studio та запуск проектів не потребує особливих дій, окрім того, що при роботі під керуванням операційної системи Windows 7 запускати його необхідно з правами адміністратора.

Проект повністю сумісний з Microsoft Visual Studio 2008, або більш пізньою версією

Редактор Unity3D без проблем відкриває проекти, для цього потрібно лише вказати кореневу папку проекту. Якщо у обраній папці не має проекту то редактор заблокує кнопку відкриття.

Проблеми можуть виникнути лише з бібліотекою драйвера характерографа, оскільки вона завантажується на етапі виконання та лише при спробі звернутись до приладу. Отже потрібно слідкувати, щоб шлях завантаження співпадав розміщенням бібліотеки. Або можна просто скопіювати файл бібліотеки в кореневу папку проекти чи папку з побудованою програмою.

3.6 Висновки

Розроблене програмне забезпечення є модульним і універсальним. Драйвер приладу логічно відділений від модуля відображення. Фактично вони обидва можуть використовуватись окремо.

Виходячи з апаратних особливостей мікроконтролера та способу їх використання було встановлено точність та роздільну заданість системи:

- Керована напруга від 0 до 50 В з кроком 0,05 В;
- Керований струм бази від 0 до 10 мА з кроком 40 мкА;
- Похибка вимірювання струму колектора не перевищує 250 мкА;
- Максимальна роздільна здатність — 262144 точок.

4 ОХОРОНА ПРАЦІ

Метою даного розділу дипломного проекту виявлення потенційне шкідливих і небезпечних виробничих факторів, які можуть мати місце при розробці, налагодженні та експлуатації системи вимірювання та візуалізації характеристик транзисторів, що розробляється, а також розробка заходів щодо зменшення впливу цих факторів на людину й навколишнє середовище.

У першу чергу, необхідно провести аналіз та оцінку небезпечних та шкідливих виробничих факторів, які мають місце при проведенні технологічного процесу пайки, розглянути вимоги до електробезпеки розробленого пристрою та питання електробезпеки та пожежної безпеки в робочому приміщенні, де виконувалася ця робота.

4.1 Технічні заходи щодо зменшення впливу ЕМВ при налагоджуванні спроектованого виробу

Якщо на організм людини виявити тривалий вплив НВЧ випромінювання високої інтенсивності то це може призвести до незворотних змін в її організмі.

Джерелом такого випромінювання в робочому приміщені може бути як розроблений пристрій, так і інша радіоелектронна апаратура, яка в ньому використовується.

Гранично припустимі значення щільності потоку енергії ЕМП розраховуються виходячи з вимог ГОСТ 12.1.006–84 та ДСНіП№476. Згідно з цими документами гранично припустима напруженість ЕМВ на робочих місцях жорстко нормується та неповинна перевищувати заданих значень,.

Для зменшення впливу ЕМВ на технічний персонал передбачається використовувати додаткове екранування джерел випромінювання, використання еквівалентів навантаження випромінюючих елементів, а також захист часом та відстанню.

4.1.1 Небезпечні та шкідливі виробничі чинники при монтажі друкованих плат

Найбільша кількість небезпечних факторів сконцентрована на виробництві апаратної частини комплексу, а саме при його складанні.

Згідно СНиП 2.09.04—87* АДМИНИСТРАТИВНЫЕ И БЫТОВЫЕ ЗДАНИЯ площа виробничого приміщення на одного робітника дорівнює 4,5 м^2, об'єм — 15 м^3, висота виробничого приміщення — 3 м.

До складу небезпечних факторі, що шкодять здоров'ю працівників можна віднести наступні:

- запиленість і загазованість робочої зони;
- наявність інфрачервоного випромінювання;
- незадовільна освітленість робочих місць чи підвищена яскравість;
- незадовільні метеорологічні умови в робочій зоні;
- небезпека ураження електричним струмом;
- вплив бризок і крапель розплавленого припою;
- психофізіологічні перевантаження.

4.1.2 Аналіз біологічного впливу небезпечних та шкідливих факторів на людину в процесі пайки

Процеси пайки супроводжуються забрудненням повітряного середовища аерозолем припою, флюсу, парами різних рідин, застосовуваних для флюсу, при змиванні і розчиненні різних лаків, що застосовуються для покриття друкованих плат та ін. (табл. 4.1).

Таблиця 4.1 — Біологічна дія, клас небезпеки і ГДК (гранично допустима концентрація) у повітрі робочої зони шкідливих речовин

Компоненти	Характер токсичності і дія	Клас небезпеки	ГДК у повітрі роб. зони, мг/м^3
Олово	Ураження бронхів, викликає поліферативно-креточну реакцію в легенях. При тривалому впливі можливий пневмоконіоз	3	10
Свинець	При отруєнні спостерігається ураження нервової системи, крові, серцево-судинної системи, шлунково-кишкового тракту, статевої системи, порушення плину вагітності	1	0,01
Вісмут	Подібно дії інших металів викликає пригнічення активності ферментів, викликає ембріонотропну і гонадотропіну дію	2	0,5
Сурма	При гострому отруєнні - ураження дихальних шляхів, травного тракту, а при хронічному отруєнні - поразка ще і нервової системи, серцевого м'яза, пневмоконіоз, гінекологічні захворювання, порушення вагітності	2	0,5
Каніфоль Соснова	Має дратівну дію. При тривалому впливі на шкіру викликає дерматит	4	140
Спирт етиловий	Має наркотичну і дратівну дію. Викликає зміни в печінці, серцево-судинній системі, нервовій системі, сухість шкіри при тривалому контакті	4	1000
Етилацетат	Помірно дратує слизову оболонку очей, верхніх дихальних шляхів. Викликає дерматит і екземи	4	200
Кислота ортофосфорна	Володіє загально токсичною дією. Пари викликають атрофічні процеси слизової носу, запальні захворювання шкіри	2	1
Бензин	Подразнює і діє як наркотик. Функціональні нервові розлади, що супроводжуються м'язовою слабістю, млявістю, чи сонливістю, безсонням. Розлад травлення, печінки, тремтіння пальців і мови, ураження шкіри. Характерний розвиток судорог, знижується кров'яний тиск, пульс уповільнюється	4	300 (у перерахуванні на вуглець)

4.1.3 Біологічна дія інфрачервоного випромінювання на організм людини при проведенні процесу пайки.

При монтажі друкованої плати робітники, в основному, піддаються впливу теплового (інфрачервоного) випромінювання, що викликане нагрітою поверхнею паяльника.

ІЧ випромінювання створює, в основному, тепловий вплив на людину. Ефект дії ІЧ променів залежить від довжини хвилі. ІЧ випромінювання підрозділяється на три області: А, В, С. До області А відноситься випромінювання з довжиною хвилі 760–1500 нм, В — 1500–3000 нм, С — більш 3000 нм. Перша область має велику проникність через шкіру. Дія ІЧ променів при поглинанні їх у різних шарах шкіри приводить до її перегрівання, що обумовлює переповнення кровоносних судин кров'ю і посилення обміну речовин. Збільшується зміст фосфору і натрію в крові, посилюється секреторна функція шлунка, підшлункової і слинної залоз, розвиваються гальмівні процеси, зменшується нервово-м'язова збудливість. Підвищується серцебиття, відбувається підвищення максимального і зниження мінімального кров’яних тисків, підвищується температура тіла, підвищується кількість захворювань серцево-судинної системи й органів травлення.

Найбільш важкі ураження викликаються коротким ІЧ випромінюванням. Допустима щільність потоку енергії ІЧ випромінювання складає відповідно до ГОСТ 12.1.005–88 п.1.8 та ДСН 3.3.6.042–99:

Інтенсивність теплового опромінення працюючих від нагрітих поверхонь технологічного устаткування, освітлювальних приладів, інсоляції на постійному і непостійному робочому місцях не повинна перевищувати 35 Вт/м^2 при опроміненні 50% поверхні тіла і більш, 70 Вт/м^2 — при величині поверхні, що опромінюється, від 25% до 50% і 100 Вт/м^2 — при опроміненні не більш 25% поверхні тіла (табл. 4.2).

Інтенсивність теплового опромінення працюючих від відкритих джерел (нагрітий метал, скло, «відкрите» полум’я й ін.) не повинне перевищувати

140 Вт/м^2, при цьому опроміненню не повинно піддаватися більш 25% поверхні тіла й обов'язковим є використання засобів індивідуального захисту обличчя й очей.

Таблиця 4.2 — Допустимі щільності потоку енергії інфрачервоного випромінювання

Області інфрачервоного випромінювання	Довжина хвилі, нм	Допустима щільність потоку енергії, Вт/м^2	Допустима інтегральна щільність потоку енергії ІЧ випромінювання, Вт/м^2
A	760–1500	100	100
B	1500–3000	120	
C	3000–4500	140	
	>4500	120	

4.1.4 Визначення концентрації аерозолю свинцю в повітрі робочої зони при пайці ЕРЕ

Кількість аерозолю свинцю, що виділяється при пайці в атмосферу складає 0,02÷0,04 мг на 100 пайок.

Відповідно до організації робочого місця і вибору виробничого приміщення, приймаємо:

- кількість робочих місць — 4;
- розміри приміщення — 4,7х4х3,5 м;
- кількість пайок у хвилину — 10;

Визначаємо концентрацію в атмосфері аерозолю свинцю при ручній пайці по формулі:

$$C = 0,6\frac{y \cdot n \cdot t \cdot N}{v},$$

де: y — питоме утворення аерозолю свинцю, мг/100 пайок, n — кількість пайок на хвилину, шт., t — тривалість зміни, год., N — кількість робочих місць, на яких ведеться пайка, шт., v — об'єм приміщення, $м^3$.

Тоді:

$$C = 0,6(0,04 \cdot 10 \cdot 8 \cdot 4)/65,8 = 0,116 \text{ мг/м}^3.$$

Порівнюючи отримане фактичне значення з ГДК, зазначеними в таблиці 4.2, робимо висновок, що концентрація свинцю в повітрі робочої зони на порядок перевищує ГДК, тому необхідно передбачити місцеву вентиляцію, розрахунок якої буде приведений нижче.

Визначення інтенсивності інфрачервоного випромінювання при проведені технологічного процесу пайки

Інтенсивність опромінення Е від нагрітої поверхні визначаємо по формулі (для $r \geq \sqrt{S}$, 100 мм > 17.3 мм):

$$E = \frac{0.91 \cdot S \cdot \left[\left(\frac{T}{100} \right)^4 - A \right]}{r^2},$$

де r — відстань до джерела теплового випромінювання ($r \approx 100$ мм), S — площа випромінюючої поверхні ($S \approx 300$ мм ^{2}), $A = 85$ для шкіри людини і бавовняної тканини, T — температура випромінюючої поверхні ($T \approx 573$ К температура плавлення припою 240°С плюс запас 50÷60°С).

$$E = \frac{0,91 \cdot 3 \cdot 10^{-4} \cdot \left[\left(\frac{573}{100} \right)^4 - 85 \right]}{0,1^2} = 27,1 \text{ Вт/м}^2.$$

Визначимо, до якої області ІЧ випромінювання відноситься випромінювання тіла з температурою 573 К. За законом Віна:

$$\lambda_{max} = \frac{2,88}{T} = \frac{2,88}{573} \approx 5000 \text{ нм}.$$

Отже, дане випромінювання відноситься до області С. Так як отримана при розрахунку щільність Е = 27,1 Вт/м ^{2} менше припустимої 120 Вт/м ^{2}, то можна сказати, що ІЧ випромінювання не буде робити шкідливої дії на організм людини.

4.1.5 Вимоги до системи місцевої вентиляції і її розрахунок

Оскільки концентрація парів свинцю в повітрі перевищує гранично припустиму норму, то необхідно застосувати місцеву вентиляцію.

Вентиляційні установки включаються до початку роботи і виключаються після її закінчення. Робота вентиляційних установок контролюється за допомогою світлової сигналізації.

Розведення вентиляційної мережі і конструкція місцевих витяжок забезпечують можливість регулярної очистки повітропроводів. Електропаяльник у робочому стані знаходиться в зоні дії витяжної вентиляції.

Широке застосування при пайці має витяжна місцева вентиляція, що умовно розділяється на місцеві витяжки відкритого і закритого типу.

У даному випадку, для уловлювання шкідливих парів, що виділяються при пайці, використовуємо місцеву витяжку у вигляді прямокутного отвору

Визначаємо кількість повітря, що відсмоктується прямокутним отвором:

$$L = v_x \ \ S + 7,7E^{0,63}X^{1,4} \ \ ,$$

де S — площа усмоктувального отвору, м^2, E — більша сторона отвору, м, X — відстань від площини усмоктувального отвору до зони пайки, v_x — швидкість повітря в зоні пайки, v_x = 0,6 м/с.

Величину E вибираємо приблизно рівною найбільшій стороні друкованої плати. Габарити друкованої плати 250×160 мм.

Приймемо E = 0,2 м, X = 0,1 м. Витяжку треба максимально наблизити до зони пайки.

Визначимо оптимальний розмір найменшої сторони усмоктувального отвору :

$$B = E \cdot 0,24 \cdot \left(\frac{X}{E} \right)^{0,33} = 0,2 \cdot 0,24 \cdot 0,5^{0,33} \approx 0,04 \text{ м}.$$

Площа *S* усмоктувального отвору:

$$S = B \cdot E = 0,04 \cdot 0,2 = 0,008 \text{ м}^2.$$

Кількість повітря, що відсмоктується отвором 0,2 м х 0,04 м:

$$L = 0,6 \ \ 0,008 + 7,7 \cdot 0,2^{0,63} \cdot 0,1^{1,4} \ \ = 0,07 \left[\frac{\text{м}^3}{\text{с}} \right] = 257 \left[\frac{\text{м}^3}{\text{год.}} \right].$$

Визначаємо концентрацію в атмосфері аерозолю свинцю при ручній пайці по формулі:

$$C_{\text{ФАКТ}} = 0,6 \frac{y \cdot n \cdot t \cdot N}{v + L \cdot t'},$$

де: *y* — питоме утворення аерозолю свинцю, мг/100 пайок, *n* — кількість пайок у хвилину, шт., *t* — тривалість зміни, год., *t'* — тривалість роботи витяжки, год., *N* — кількість робочих місць, на яких ведеться пайка, шт., *v* — об'єм приміщення, м ^{3}.

Тоді:

$$C_{\text{ФАКТ}} = 0,6 \frac{0,04 \cdot 10 \cdot 8 \cdot 4}{65,8 + 1028 \cdot 8} = 9,2 \cdot 10^{-4} \ \ \frac{\text{мг}}{\text{м}^3}.$$

Так як $C_{\text{доп}} > C_{\text{факт}}$, то в застосуванні спеціальних заходів щодо охорони навколишнього середовища немає необхідності.

4.2 Електробезпека

Згідно ГОСТ 12.2.007.0–75 спроектований комплекс має II клас оскільки має подвійну ізоляцію.

При монтажі друкованих плат, які входять до складу даного пристрою, найбільш ймовірною причиною поразки робітників електричним струмом, є дотик до струмоведучих частин у результаті руйнування ізоляції. Електрична мережа, що підведена до робочого місця — однофазна мережа змінного

струму частотою 50 Гц і напругою 220 В з заземленою нейтраллю, зануленням та з застосуванням автоматів струмового захисту.

Для підвищення електробезпеки в робочому приміщені застосовується понижена напруга для електропаяльника (потужністю 25 Вт та напругою 36 В). Для вимірювальної апаратури використовується електромережа — 220 В. Клеми введення електроенергії до робочого місця обгороджені кожухом щоб уникнути випадкового дотику. Роз'єми, а також закріплення проводів і кабелів в електроінструментах відповідають технічним вимогам і мають елементи заземлення. В аварійному режимі використовуємо подвійну ізоляцію проводів, що живлять електропаяльник. Оскільки електронний блок містить напівпровідникові прилади і мікросхеми, що можуть бути піддані впливу статичної електрики, то необхідно заземлити руки радіомонтажника, жало електропаяльника та корпус апаратури.

Зібрану схему, електроапаратуру підключають до джерел живлення через запобіжники з відповідними по струму і напрузі нормованими плавкими вставками.

Дане виробниче приміщення, згідно ПУЕ, відноситься до приміщень без підвищеної небезпеки.

Для запобігання ураження електричним струмом необхідно здійснювати періодичний контроль ізоляції.

4.2.1 Розрахунок електромережі з зануленням на здатність до вимикання

Занулення - це навмисне електричне з'єднання з нульовим запобіжним проводом металевих не струмоведучих частин, які можуть бути під напругою в разі пробою ізоляції струмоведучих частин.

При з'єднанні металевих не струмоведучих частин електрообладнання з нульовим проводом живлячої мережі замикання фази на корпус стає однофазним коротким замиканням. Виникаючий струм однофазного короткого замикання повинен забезпечити надійне спрацювання автомату

максимального струмового захисту та автоматичного відключення від живлячої електромережі несправного електрообладнання.

Розрахунок однофазного короткого замкнення.

$$I_{\text{КЗ}} = \frac{U_{\Ф}}{\sqrt{(r_{\Ф} + r_{\text{Н}} + \frac{r_{\text{Т}}}{3})^2 + (x_{\Ф} + x_{\text{Н}} + \frac{x_{\text{Т}}}{3})^2}}.$$

Для кабельних ліній індуктивним опором петлі фазний провід — індуктивний провід можливо знехтувати:

$$I_{\text{КЗ}} = \frac{U_{\Ф}}{r_{\Ф} + r_{\text{Н}} + \frac{r_{\text{Т}}}{3}}.$$

де $r_{\Ф}$ — активний опір фазного проводу, $r_{\text{Н}}$ — активний опір нульового проводу, $r_{\text{Т}}/3$ — активний опір обмоток трансформатору, Активний опір для $r_{\Ф}$ та $r_{\text{Н}}$ визначають за формулою:

$$r = \rho \frac{l}{S},$$

де ρ — питомий опір, l — довжина провідника, S — поперечний переріз проводу.

Найбільша довжина проводу $l_{\Ф} = l_{\text{Н}} = 15$ м, поперечний переріз $S_{\Ф} = S_{\text{Н}} = 1{,}5$ мм^2. Матеріал провідника мідь $\rho = 0{,}0175$ Ом·мм^2/м.

$$r = 0{,}0175 \frac{15}{1{,}5} = 0{,}175 \text{ Ом}.$$

$r_{\text{Т}}/3$ — визначаємо з табл. 10.11.

$r_{\text{Т}}/3 = 0{,}162$ Ом.

$$I_{\text{КЗ}} = \frac{220}{0{,}175 + 0{,}175 + 0{,}162} = 492{,}7 \text{ А}.$$

Стум максимального токового захисту для автоматичного роз’єднувача 25 А, $K_{\text{Т}} >> K_{\text{Т доп.}}$

$$K_{\text{Т}} = 429{,}7/25=17{,}2,$$

$$K_{\text{Т доп .}} = 1{,}25.$$

Кратність струму перевищує значення необхідне для безпечного відключення токового захисту.

Розрахуємо напругу на корпусі приладу при короткому замкненні відносно землі. Напруга не повинна перевищувати 42 В.

$$U_{\text{Н}}=I_{\text{КЗ}}Z_{\text{Н}}.$$

Для кабельної лінії $Z_{\text{Н}} = r_{\text{Н}}$.

$$U_{\text{Н}}=429{,}7\cdot 0{,}175=75{,}2 \text{ В}.$$

Для зниження напруги на корпусі приладу при короткому замкненні необхідно збільшити поперечний переріз нульового проводу, тому для нульового проводу візьмемо поперечний переріз 4,5 мм^2. Проведемо розрахунок з початку.

$$r_{\text{Н}} = 0{,}0175\frac{15}{4{,}5} = 0{,}06 \text{ Ом},$$

$$I_{\text{КЗ}} = \frac{220}{0{,}175+0{,}06+0{,}162} = 554{,}2 \text{ А},$$

$$U_{\text{Н}}=554{,}2\cdot 0{,}06=33{,}3 \text{ В}.$$

Підвести фазні проводи необхідно проводом поперечного перерізу 1,5 мм^2, нульовий провід необхідно підвести проводом поперечного перерізу не менше ніж 4,5 мм^2. Це приведе до зниження напруги на корпусі приладу до безпечної позначки.

4.3 Заходи щодо пожежної безпеки

На ділянці монтажу застосовуються деякі речовини і матеріали, що є пожежо- та вибухонебезпечні (табл. 4.3).

Таблиця 4.3 — Пожежовибухонебезпечні речовини, що застосовуються при виробництві друкованого вузла.

Найменування речовини	Темпер. запален. °С	Темпер. самозапалення, °С	Межа вибухання		Засоби пожежогасіння
			нижня	Верхня	
Каніфоль	–	850	12,6 г/м^3	–	Хімічна и повітряно-механічна піна, розпилена вода
Спирт етиловий	18	104	3.6%/68г/м^3	19%/340г/м^3	Хімічна піна, вода, пар, інертні гази
Бензини	17÷44	255÷474	0,76÷1.1%	5,16÷8,12 %	піна, водяний пар, інертні гази
Склотекстоліт	–	–	–	–	вода, хімічна піна

Для того щоб визначити категорію приміщення по вибухо-пожежній і пожежні небезпеці відповідно до НАПБ Б.03.002-2007, необхідно розрахувати надлишковий тиск вибуху в приміщенні. Визначимо його за формулою:

$$\Delta P = P_{max} - P_0 \frac{mZ}{V_{св}\rho_{гп}} \cdot \frac{100}{C_{СТ}} \cdot \frac{1}{K_н},$$

де P_{max} — максимальний тиск вибуху стехіометричної газоповітряної чи пароповітряної суміші в замкнутому просторі визначається за довідниками (P_{max} = 750 кПа); P_0 — початковий тиск (P_0 = 101 кПа); m — маса горючої речовини, кг; Z — коефіцієнт участі горючої речовини (Z = 0,3); $V_{св}$ — вільний об'єм приміщення, м^3; $\rho_{гп}$ — щільність газу і пару ($\rho_{п\ етил.\ спирт}$ по повітрю = 1,6 кг/м^3); $C_{СТ}$ — стехіометрична концентрація горючого газу чи парів ЛЗР (легко запальних речовин), %; $K_н$ — коефіцієнт, що враховує негерметичність приміщення і не адіабатичність процесу горіння ($K_н$ = 3).

$V_{св}$ — визначимо за формулою:

$$V_{св} = 0{,}8V_{прим},$$

$C_{СТ}$ визначимо по формулі:

$$C_{СТ} = 100/(1+4{,}84\beta),$$

де $\beta = n_C + \frac{n_H - n_X}{4} - \frac{n_O}{2}$ — стехіометричний коефіцієнт кисню в реакції горіння; n_C, n_H, n_O, n_X — число атомів С, Н, О і галоїдів у молекулі пального.

Розрахуємо ΔP за вищевказаною методикою, прийнявши до відомості, що:

- $V_{прим}$ = 65,8 м^3;
- на ділянці монтажу щодня витрачається 0,3 л спирту;
- розрахунок зробимо для самого несприятливого випадку — весь вміст надходить у приміщення (для 0,3 л ЛЗР площа розливу відповідає 0,3 м^2).

Масу парів рідини m визначимо по формулі:

$$m = W \cdot S \cdot T,$$

де W — інтенсивність випару, кг/(с·м^2); S — площа випару, м^2; T — тривалість випару (T = 3600 с).

$$W = 10^{-6}\eta\sqrt{M}P_n,$$

де η — коефіцієнт, обираний з табл.П2 у залежності від швидкості і температури над поверхнею рідини, при $\nu_{пов}$=0,2 м/с та $t_{пов}$=20 °С – η = 3.5; M — молекулярна маса (M = 46 г/моль); P_n — тиск насиченої пари; для C_2H_5OH, P_n = 5.85 кПа.

У результаті проведеного розрахунку можна зробити висновок, що дане приміщення відноситься по пожежонебезпеці до категорії В згідно вимог НАПБ Б.03.002-2007 та НАПБ Б.07.005-86. Оскільки в приміщенні для монтажу друкованих плат вибухонебезпечні суміші горючих парів і газів з повітрям не утворяться, а утворяться вони тільки в результаті аварії чи

несправності, то робочу зону приміщення можна віднести до класу П-IIа по пожежній небезпечності згідно НПАОП 40.1-1.32-01.

Основними причинами виникнення пожеж є:

- порушення встановлених правил пожежної безпеки НАПБ.А.01.001–14 і необережне поводження з вогнем;
- несправність і перевантаження електричних пристроїв (коротке замикання);
- несправність вентиляційної системи, що викликає осідання, самозаймання і вибухи пилу;
- халатне і необережне поводження з вогнем;
- самозапалювання бавовняної тканини, просоченої олією, бензином чи спиртом;
- статична електрика, що утвориться від тертя пилу чи газів у вентиляційних установках;
- грозові розряди при відсутності чи несправності блискавковідводів.

У приміщеннях, де провадиться монтаж друкованих плат, передбачаємо згідно вимог ДБН В.2.5–13–98 електричну пожежну сигналізацію (п'ять приладів для сповіщення типу СПД–1 та автоматичний пульт пожежної сигналізації), що служить для швидкого повідомлення служби пожежегасіння про виникнення пожежі.

Вхід у приміщення, проходи між робочими столами і коридори не дозволяється захаращувати різними предметами й устаткуванням, максимальна віддаленість робочих місць від евакуаційних виходів та ширина проходів відповідають вимогам СНиП 2.09.02–85. Мінімальна межа вогнестійкості будівлі відповідає вимогам СНиП 2.01.02–85. Для збереження всіх пожежонебезпечних речовин і матеріалів передбачаємо спеціальні шафи і ємності.

Як первинні засоби пожежогасіння в робочому приміщенні застосовуються вогнегасники, що розташовані безпосередньо в приміщенні, їх тип та кількість відповідають вимогам ISO3941–77.

4.4 Відповідність рівня освітленості робочої зони санітарним нормам

Загальне освітлення в робочому забезпечується за допомогою світильників з лампами денного світла типу ЛБ–40, потужністю 40 Вт, а місцеве за допомогою світильників з більш потужними лампами (60 Вт), і напругою 36 В.

Для розрахунку загального освітлення робочого приміщеня можна скористатись методом коефіцієнта використання світлового потоку, призначеного для розрахунку загального рівномірного освітлення горизонтальних поверхонь, при відсутності предметів, що затемнюють. При цьому в розрахунках враховується пряме та відбите світло. Необхідний світловий потік ламп у кожному світильнику визначається по формулі:

$$\Phi = \frac{E \cdot K \cdot S \cdot Z}{N \cdot n}.$$

Фактичне висвітлення робочих місць штучним освітленням визначається по формулі:

$$E_{\phi} = \frac{N \cdot n \cdot \Phi}{S \cdot K \cdot Z}\eta,$$

де N — кількість світильників (6шт); n — кількість ламп у світильнику(4шт); η — коефіцієнт використання світлового потоку; S — площа приміщення (48м^2); K — коефіцієнт запасу; Z — коефіцієнт нерівномірності висвітлення; Φ — світловий потік лампи (2800 лм).

Для визначення коефіцієнта використання світлового потоку визначаємо індекс приміщення і і коефіцієнт відбиття стелі $\rho_п$, стін $\rho_с$, робочої поверхні $\rho_р$.

$$i = \frac{l \cdot b}{h \ \ l + b},$$

де l — довжина приміщення, *м;* b — ширина приміщення, *м;* h — висота підвісу світильників, *м.*

$$i = \frac{9 \cdot 6}{2,6\ \ 9+6} = 1,4$$

Коефіцієнт відбиття побіленої стелі $\rho_п$ = 0,7, побілених стін при незавішених вікнах $\rho_с$ = 0,5, середніх робочих поверхонь $\rho_р$ = 0,3.

Для визначення коефіцієнта використання світлового потоку необхідно знати, що використаються лампи ЛБ-40 серії УСП5-4х40 (чотири лампи з розсіювачами). Тоді на підставі вищевикладеного знайдемо коефіцієнт, використовуючи табличні дані *(η=0,44)*.

В результаті отримаємо:

$$E_ф = \frac{6 \cdot 4 \cdot 2800 \cdot 0,44}{48 \cdot 1,5 \cdot 0,8} = 514 \text{лк}$$

Штучне освітлення в приміщеннях регламентується нормами ДБН В.2.5–28–2006. Для зорової роботи 4 розряду під розряд В при загальному освітленні це 500 лк. У нашому випадку фактичне освітлення більше припустимих норм.

При нормуванні природного освітлення промислових будинків використовується коефіцієнт природного освітлення (КПО). Значення КПО наведені в ДБН В.2.5–28–2006 для даного виду зорових робіт — 1,5%.

Фактичне значення природного освітлення при боковому освітленні визначається за формулою:

$$l_p^б = \ E_б q + E_{зд} R \ \ r_1 \frac{r_0}{K_з},$$

де $E_б$ — геометричний КПО в розрахунковій точці при боковому освітленні, що враховує пряме світло неба й визначається:

$$E_б = 0,01 \ \ n_1 n_2 \ \ ,$$

де n_1 — кількість променів, прохідних від неба через світлові прорізи в розрахункову крапку на поперечному перерізі приміщення; n_2 — кількість променів, що приходять із неба через світлові прорізи в розрахункову крапку

на плані приміщення; $E_{бд}$ — геометричний КПО в розрахунковій точці при боковому освітленні, що враховує світло, відбите від конфронтуючого будинку.

Причому $E_{бд} = 0$, тому що конфронтуючі будинки перебувають на відстані 100*м*, отже, світло, відбите від нього буде мізерно мале.

$$E_{б} = 0,01 \cdot \ 36 \cdot 32 \ = 11,52,$$

де q — коефіцієнт, що враховує нерівномірну яскравість хмарного неба, приймемо 0,52; r_1 — коефіцієнт, що враховує збільшення КПО при боковому освітленні завдяки світлу відбитому від поверхонь приміщення й підстильного шару, що прилягає до будинку; r_0 — загальний коефіцієнт світловипускання, визначається по формулі:

$$r_0 = r_1 r_2 r_3 r_4 r_5 ,$$

де r_1 — коефіцієнт світлопропускання матеріалу; r_2 — коефіцієнт, що враховує втрати світла в перетині світлоприймання; r_3 — коефіцієнт, що враховує втрати світла в несучих конструкціях; r_4 — коефіцієнт, що враховує втрати світла в сонцезахисних пристроях; r_5 — коефіцієнт, що враховує втрати світла в захисній сітці під ліхтарями, приймається 0,9.

$$r_0 = 0,8 \cdot 0,6 \cdot 0,8 \cdot 0,8 \cdot 0,9 = 0,276 .$$

$K_з$ — коефіцієнт запасу, приймається 1,3, при вертикальному розташуванні засклення.

Підставивши отримані значення у раніше наведену формулу, отримаємо

$$l_{р} = \frac{11,52 \cdot 0,52 \cdot 2,2 \cdot 0,276}{1,3} = 2,79 .$$

У даному випадку $l_{р} > l_{Н}$, таким чином природне освітлення у робочому приміщенні відповідає нормі.

4.5 Вимоги щодо безпечної експлуатації ПК

Оскільки прилад проектується як комплекс, який працюватиме з персональним комп’ютером то необхідно також дотримуватись умов безпеки роботи з ПК.

Відповідно до ДСанПіН3.3.2.007–98 та НПАОП 0.00-1.28-10 у приміщеннях для роботи з ПК не дозволяється розташовувати в підвалах будинків. Проходи дверей до цих приміщень не повинні мати порогів. Наявність порогів дозволяється при присутності переходів з кутом нахилу 30°. ПК установлюється відповідно до вимог заводу-виготовлювача. Монітор комп’ютера розташовується за 1 м від стін. Робочі місця з терміналами повинні розташовуватися на відстані не менш 1,5 м один від одного.

Регламентована площа приміщення на кожного працівника повинна складати не менш 6 м^2, а об’єм не менш 19.5 м^3.

Робота з ПК характеризується підвищеною інтенсивністю і монотонністю, тому необхідно виконувати наступне.

Згідно ДСанПіН3.3.2.007–98 працівник, що виконує роботи з програмною частиною системи відноситься до групи операторів, для яких потрібно призначати перерви для відпочинку тривалістю 15 хв через кожні дві години роботи. Тривалість робочої зміни не повинна перевищувати 8 годин. Присутність працівника за терміналом при 8-годинному робочому дні повинна складати не більш 4-х годин.

Робоче місце для виконання робіт у положенні сидячи повинно відповідати вимогам ГОСТ 12.2.032–78, ГОСТ 22269–76, ГОСТ 21829–76 і вимогам технічної естетики.

Для захисту від статичної електрики доцільно використовувати нейтралізатори та зволожувачі.

При експлуатації ПК у приміщенні існує як природна, так і технологічна іонізація повітря. Відповідно до вимог ГОСТ 12.2.006–87, як надлишок, так і нестача позитивних чи негативних іонів відносяться до

шкідливих виробничих факторів. Іонізуюче випромінювання негативно впливає на організм людини. Кількість іонів (як позитивних, так і негативних) у 1 $см^3$ повітря регламентується санітарно-гігієнічними нормами СН 4559–88, згідно яких мінімально необхідний рівень кількості позитивних іонів складає 400, а негативних — 600. В той час як оптимальний в межах 1000–1500 для позитивних іонів та 3000–5000 для негативних. Однак максимальний рівень для перших і для других не повинен перевищувати 50000 іонів.

Для нормалізації іонного складу повітря необхідно використовувати вентиляцію.

Для захисту людини від поразки електричним струмом конструкцією ПК передбачене електричне з'єднання з землею металевих частин корпуса ПК, що можуть бути під напругою. У ПК використовується спеціальна мережна вилка з трьома контактами.

ВИСНОВКИ

1. За результатами аналітичного огляду встановлено, що в даний час немає жодного комерційного зразка характерографа, який використовує тривимірну графіку для відображення результатів.

2. Розроблена система дозволяє проводити вимірювання вольт-амперних характеристик біполярних транзисторів. Вона включає апаратну частину, що проводить власне вимірювання та програмну, що відображає результати на екрані персонального комп'ютера. Особливістю даного рішення є використання тривимірної графіки.

3. Як показав аналіз можливих рішень, найбільш простий і дешевий спосіб організації передачі даних за допомогою шини USB — використання мікроконтролера PIC18F4550, з вбудованим USB модулем.

4. Розроблена система має такі основні параметри:

- Керована напруга від 0 до 50 В з кроком 0,05 В;
- Керований струм бази від 0 до 10 мА з кроком 40 мкА;
- Похибка вимірювання струму колектора не перевищує 250 мкА;
- Максимальна роздільна здатність — 262144 точок.

5. Наведені параметри було підтверджено в результаті тестування на моделях транзисторів з відомими характеристиками. Час вимірювання характеристики розміром 12x12 точок складає приблизно 20 секунд.

ПЕРЕЛІК ПОСИЛАНЬ

1. "Характериограф для транзисторов," Радио, Выпуск 12, 1990. — с. 78—79.
2. Цифровой запоминающий характериограф полупроводниковых приборов Л2-100 ТЕКО — Режим доступу: http://www.test-expert.ru/news/detail.php?ID=797 — Назва з екрана.
3. Type 576 Curve-Tracer — Режим доступу: http://circuitslab.case.edu/manuals/Tektronix_Type_576_Curve_Tracer.pdf — Назва з екрана.
4. ЦИФРОВОЙ ХАРАКТЕРИОГРАФ «ЭРБИЙ-7107» ДЛЯ ИСПЫТАНИЙ ПОЛУПРОВОДНИКОВЫХ ДВУХПОЛЮСНИКОВ — Режим доступу: http://erbysar.com/docs/buklet_erby_7107.pdf — Назва з екрана.
5. Характериограф Транзисторов Tr-4805 — Режим доступу: http://forum.cxem.net/index.php?showtopic=113266 — Назва з екрана.
6. Микроконтроллеры: краткий обзор — Режим доступу: http://www.myrobot.ru/stepbystep/mc_meet.php — Назва з екрана.
7. Хюльцебош Ю. USB в электронике / Ю. Хюльцебош — СПб. : БХВ-Петербург, 2009. — 224 с.: ил. — ISBN 978-5-9775-0324-2.
8. Poteus и два с половиной hello world для UART и USB на микроконтроллере — Режим доступу: http://habrahabr.ru/post/206034/ — Назва з екрана.
9. MAX16824 Высоковольтные 3-х канальные драйверы светодиодов высокой яркости — Режим доступу: http://catalog.gaw.ru/index.php?page=component_detail&id=3406 — Назва з екрана.
10. Повышающий DC-DC преобразователь. Принцип работы. — Режим доступу: http://easyelectronics.ru/povyshayushhij-dc-dc-preobrazovatel-princip-raboty.html — Назва з екрана.
11. Регуляторы напряжения для питания современных процессоров семейства Intel Pentium и процессоров AMD. — Режим доступу: http://mirpu.ru/motherboard/81-2011-02-12-20-10-05/153--intel-pentium-amd.html — Назва з екрана.
12. Управление MOSFET-ами №1 — Режим доступу: http://vasilisks.wordpress.com/2013/01/07/управление-mosfet-ами-1/ — Назва з екрана.
13. Как сделать простой цифро-аналоговый преобразователь (ЦАП, DAC)? — Режим доступу: http://kazus.ru/faq/3/46.html — Назва з екрана.

14. PIC18F2455/2550/4455/4550 Data Sheet / Microchip Technology Inc — Режим доступу: ww1.microchip.com/downloads/en/DeviceDoc/39632b.pdf — Назва з екрана.
15. Ямпурин Н. П. Основы надежности электронных средств : учеб. пособие для студ. высш. учеб. заведений / Н. П.Ямпурин, А. В. Баранова; под ред. Н. П. Ямпурина. — М. : Издательский центр «Академия», 2010. — 240 с.
16. Interrupt в PIC18 — Режим доступу: http://pro-diod.ru/programms/pic-micro/interrupt-v-pic18.html — Назва з екрана.
17. Микроконтроллеры PIC фирмы Microchip для начинающих — Режим доступу: http://subscribe.ru/archive/comp.soft.prog.pic/200703/13210529.html — Назва з екрана.
18. Широтно-Импульсная модуляция — Режим доступу: http://catcatcat.d-lan.dp.ua/skachat/primeryi-postroeniya-koda-programm-dlya-pic-kontrollerov/shirotno-impulsnaya-modulyatsiya/ — Назва з екрана.
19. PWM, DAC, LPF или как это по нашему ШИМ, ЦАП, ФНЧ — Режим доступу: http://picdevices.ru/eksperimentyi/pwm-dac-lpf-ili-kak-eto-po-nashemu-shim-tsap-fnch.html?doing_wp_cron=1464891333.4888598918914794921875 — Назва з екрана.
20. Реализация ШИМ на PIC-контроллерах — Режим доступу: http://portal.tpu.ru/SHARED/p/PEST/pwm.pdf — Назва з екрана.
21. АЦП в PIC18F45xx — Аналого-Цифровое преобразование — Режим доступу: http://pro-diod.ru/programms/pic-micro/acp-v-pic18f45xx-analogo-cifrovoe-preobrazovanie.html — Назва з екрана.
22. PIC18F2455/2550/4455/4550 Data Sheet / Microchip Technology Inc — Режим доступу: http://ww1.microchip.com/downloads/en/DeviceDoc/39632e.pdf — Назва з екрана.
23. Модульное программирование — Режим доступу: https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5 — Назва з екрана.
24. Создание и использование DLL — Режим доступу: http://www.xserver.ru/computer/sredaprogr/msvc/2/ — Назва з екрана.
25. DllMain entry point — Режим доступу: https://msdn.microsoft.com/ru-ru/library/windows/desktop/ms682583(v=vs.85).aspx — Назва з екрана.
26. Каналы передачи данных Pipe — Режим доступу: http://www.frolov-lib.ru/books/bsp/v27/ch2_3.htm — Назва з екрана.

27. Синхронизация процессов и потоков — Режим доступу: http://www.codenet.ru/progr/cpp/process-threads-sync.php — Назва з екрана.
28. How to Write Native Plugins for Unity — Режим доступу: http://www.alanzucconi.com/2015/10/11/how-to-write-native-plugins-for-unity/ — Назва з екрана.
29. Dynamically calling an unmanaged dll from.NET (C#) — Режим доступу: https://blogs.msdn.microsoft.com/jonathanswift/2006/10/03/dynamically-calling-an-unmanaged-dll-from-net-c/ — Назва з екрана.
30. IntPtr.Zero - поле — Режим доступу: https://msdn.microsoft.com/ru-ru/library/system.intptr.zero(v=vs.110).aspx — Назва з екрана.
31. Unload a plugin — Режим доступу: http://answers.unity3d.com/questions/10216/unload-a-plugin.html — Назва з екрана.
32. Easier way to handle unloading dlls? — Режим доступу: http://answers.unity3d.com/questions/293867/easier-way-to-handle-unloading-dlls.html — Назва з екрана.
33. Взаимодействие с неуправляемым кодом — Режим доступу: http://www.realcoding.net/articles/glava-17-vzaimodeistvie-s-neupravlyaemym-kodom.html — Назва з екрана.
34. NET и работа с неуправляемым кодом. Часть 1 — Режим доступу: https://habrahabr.ru/post/84076/ — Назва з екрана.
35. Программирование трехмерной графики. Часть 3. — Режим доступу: http://www.alexeyspace.ru/articles/3/ — Назва з екрана.
36. Процедурная генерация трёхмерных моделей — Режим доступу: https://habrahabr.ru/post/194620/ — Назва з екрана.
37. Create a mesh and color cubes — Режим доступу: http://answers.unity3d.com/questions/391561/create-a-mesh-and-color-cubes.html — Назва з екрана.
38. Proteus (система автоматизированного проектирования) — Режим доступу: https://ru.wikipedia.org/wiki/Proteus_(%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0% — Назва з екрана.
39. Proteus VSM USB Simulation — Режим доступу: http://www.labcenter.com/products/usb.cfm — Назва з екрана.

Додаток А Технічне завдання

Додаток Б Схема електрична принципова приладу