

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

КУРСОВИЙ ПРОЄКТ

Фаховий молодший бакалавр
(освітньо-професійний ступінь)

на тему: «Проектування систем керування та збору інформації на базі
мікроконтролерів сімейства PIC16»

Виконав студент IV курсу, групи *KI-42*
Спеціальності *123 Комп'ютерна інженерія*
Варіант № 33
Качмарик Дмитро Іванович
(прізвище, ім'я по батькові)

Керівник _____ *Роман СТОЛЯРЧУК*
(підпис) (ім'я прізвище)

Курсовий проєкт перевірений
і допущений до захисту:

«__» _____ 2025р.

Курсовий проєкт захищений «__» _____ 2025 р.
з оцінкою « _____ »

Львів 2025

**ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

ЗАТВЕРДЖЕНО
на засіданні циклової комісії
«Фундаментальної підготовки»
Протокол № 1 від 28 серпня 2025 р.
Голова комісії

_____ Ольга ЛАБАЗ

ЗАВДАННЯ

на курсовий проєкт варіант № 33

Качмарику Дмитру Івановичу

(прізвище, ім'я та по батькові)

з навчальної дисципліни: **МІКРОКОНТРОЛЕРИ ТА ЇХ ПРОГРАМУВАННЯ**

Студент групи: **KI-42**

1. Тема проєкту: Система моніторингу температури.

2. Дата видачі завдання: "23" вересня 2025 р

3. Термін здачі курсового проєкту: "16" грудня 2025 р.

4. Вихідні дані до проєкту:

4.1 Пристрої введення інформації : модуль ADC, кнопки

4.2 Функції пристрою введення інформації: . оцифрування даних, встановлення границь реле

4.4 Використовувані мікроконтролери: PIC16F876, PIC16F883 .4

4.5 Протокол обміну інформацією: UART.

4.6 Пристрій виведення інформації: CCI-3 р., (CA), реле

4.7 Функції пристрою виведення інформації: індикація вимірювань, 220

5 Перелік обов'язкових демонстраційних креслень:

5.1 Узагальнена структурна схема МКС

5.2 Блок-схема алгоритму роботи МКС

5.3 Скриншот роботи проєкту у програмі PROTEUS

6. Склад розрахунково – пояснювальної записки (перелік питань до розробки):

ВСТУП

- 1 Розробка структурної схеми МКС
 - 1.1 Аналіз завдання та синтез структурної схеми МКС
 - 1.2 Обґрунтування вибору мікроконтролерів
 - 1.3 Призначення, принцип роботи, особливості периферійних пристроїв
- 2 Розробка електричної принципової схеми МКС
 - 2.1 Розробка схеми електричної принципової мікроконтролера, периферійних пристроїв та їх інтерфейсу.
 - 2.2 Створення проекту з використанням програми PROTEUS
- 3 Розробка програмного забезпечення
 - 3.1 Розробка алгоритму функціонування МКС
 - 3.2 Написання керуючої програми для мікроконтролера :
 - робота з програмою MPLAB, створення проекту;
 - конфігурування МК з використанням директиви __CONFIG;
 - ініціалізація мікроконтролерів;
 - ініціалізація периферійних модулів;
 - пояснення до написання основної частини програми.
 - 3.3 Використання програми PROTEUS для аналізу роботи проекту.

ВИСНОВКИ

ПЕРЕЛІК ПОСИЛАНЬ

Календарний план

Назва етапів	Термін виконання	Примітка
Вступ		
1 Розробка структурної схеми		
2 Розробка електричної принципової схеми		
3 Розробка програмного забезпечення		
Висновки		
Перелік посилань		

Студентка

_____ (підпис)

Дмитро КАЧМАРИК

_____ (імя та прізвище)

Керівник проекту

_____ (підпис)

Роман СТОЛЯРЧУК

_____ (імя та прізвище)

РЕФЕРАТ

Текстова частина курсового проєкту: 38 сторінок, 19 рисунків, 1 таблиці та 6 посилань.

Об'єкт проєктування - "Система контролю мережі 220В".

Мета проєкту - довершити свої навички у розробці структурної схеми, електричної принципової схеми МКС, створення проєкту з використанням програми PROTEUS, розробці алгоритму функціонування МКС, а також написання керуючої програми для мікроконтролерів. Закріпити теоретичні знання з основних понять дисципліни "Мікроконтролери та їх П".

Результат роботи над курсовим проєктом показав, що завдяки використанню двох простих мікроконтролерів та модуля реального часу можна створити повноцінну систему моніторингу температури.

МК, ADC, UART, CCI, SPLAN, MPLAB IDE.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АЦП	Аналого-цифрові перетворювачі
ВІС	Великі інтегральні схеми
ГЗ	Глобальна задача
ГТІ	Генератор тактових імпульсів
ЗЗ	Загальна задача
ЗМП	Задачі малої розмірності
МК	Мікроконтролер
МКП	Мікроконтролерні пристрої
МКС	Мікроконтролерна схема
НЗ	Найпростіші задачі
ОМК	Однокристальні мікроконтролери
ПЗП	Постійний запам'ятовуючий пристрій
ЦАП	Цифроаналогові перетворювачі
ЦПП	Центральний процесорний пристрій
ШІМ	Широтно-імпульсна модуляція
I2C	Inter-Integrated Circuit (Міжінтегральна схема)
SPI	Serial Peripheral Interface (Послідовний периферійний інтерфейс)

ЗМІСТ

ВСТУП.....	7
1 РОЗРОБКА СТРУКТУРНОЇ СХЕМИ МКС.....	8
1.1 Аналіз завдання та синтез структурної схеми МКС.....	8
1.2 Вибір мікроконтролерів та аналіз їх параметрів.....	9
1.3 Опис периферійних пристроїв та робота з ними.....	11
1.3.1 Робота з модулем АЦП	11
1.3.2 Семисегментні індикатори	13
1.3.3 Робота з протоколом UART	15
1.3.4 Модуль TMR0.....	17
2 ПРОЕКТУВАННЯ ЕЛЕКТРИЧНОЇ ПРИНЦИПОВОЇ СХЕМИ.....	20
2.1 Розробка схеми електричної принципової мікроконтролера.....	20
2.2 Створення проекту в середовищі PROTEUS VSM.....	24
3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МКС.....	28
3.1 Розробка алгоритму функціонування МКС.....	28
3.2 Конфігурування пристрою з використанням директиви __CONFIG..	30
3.3 Написання керуючої програми для мікроконтролера.....	31
ВИСНОВКИ.....	37
ПЕРЕЛІК ПОСИЛАНЬ.....	38

ВСТУП

Типовими представниками RISC-процесорів є PIC-контролери (Peripheral Interface Controller – контролери периферійних інтерфейсів) виробництва фірми MicroChip.

PIC-контролери застосовуються у системах високошвидкісного керування автомобільними й електричними двигунами, приладах побутової електроніки, телефонних приставках з АВН, системах охорони із сповіщенням по телефонній лінії, міні-АТС. Окремі вимірювальні інформаційні системи відрізняються розрядністю ПЗП(постійно запам'ятовуючого пристрою): від 12 до 14 біт для серії PIC16Cxx, 16 біт – для серії PIC17Cxx.

Завдяки скороченій кількості команд (від 33 до 35) усі команди займають у пам'яті одне слово. Час виконання кожної команди, крім команд розгалуження, становить чотири такти – один цикл (200 нс на частоті 20 МГц). Оперативний запам'ятовувальний пристрій виконано за схемою з довільною вибіркою з можливістю безпосередньої адресації у коді команди до будь-якої комірки.

Стек реалізовано апаратно з глибиною 2, 8 або 16 комірок. Майже в усіх PIC-контролерах є система переривань, джерелом яких може бути таймер, а також зміна станів сигналів на деяких входах. У PIC-контролерах передбачений біт захисту ПЗП, що запобігає нелегальному копіюванню.

Великі інтегральні схеми PIC16Cxx мають вбудовані ПЗП ємністю від 0,5 до 4 кілобайт і ОЗП(оперативно-запам'ятовуючий пристрій) ємністю 32-256 байт. Основна частина контролерів має однократно програмований ПЗП, однак деякі контролери містять ПЗП з ультрафіолетовим стиранням, а PIC 16C84 містить пам'ять програм і пам'ять даних на базі ПЗП з електричним стиранням.

Крім того, контролери мають від одного до трьох таймерів, вбудовану систему скидання, watchdog таймер, внутрішній тактовий генератор, який може запускатися як від кварцового резонатора, так і від RC-ланцюга у широкому діапазоні частот – 0-25 МГц. Кількість розрядів портів – 12-33. Кожний розряд порту можна запрограмувати на введення або на виведення.

Контролер PIC16C64 додатково має вихід з ШІМ, за допомогою якого можна реалізувати ЦАП з розрядністю до 16 розрядів, а також послідовний двонаправлений синхронний порт з інтерфейсами SPI, I2C, SCI/UART. PIC 16C71 і PIC 16C74 мають внутрішній 8-розрядний АЦП із пристроєм вибирання/зберігання і вхідним аналоговим мультиплексором.

1 РОЗРОБКА СТРУКТУРНОЇ СХЕМИ МКС

1.1 Аналіз завдання та синтез структурної схеми МКС

При проектуванні мікроконтролерних пристроїв (МКП) або систем (МКС) можна використовувати блоково-ієрархічний підхід, при якому уявлення про МКП або МКС, що проектується, розчленовуються на ієрархічні рівні. На вищому рівні використовується найменш деталізоване уявлення, що відображає лише тільки загальні риси і особливості системи, що проектується. На наступних рівнях ступінь подробиць розгляду зростає, при цьому система розглядається не загалом, а окремими блоками. Такий підхід дозволяє на кожному рівні формувати і вирішувати задачі допустимої складності, що піддаються усвідомленню й розумінню людиною та рішенням за допомогою доступних засобів. Переваги такого підходу полягають в тому, що складна задача великої розмірності розбивається на групи задач малої розмірності, які послідовно вирішуються, причому всередині груп різні задачі можуть вирішуватися паралельно.

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

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

Об'єктом дослідження даного курсового проекту є Система контролю мережі 220В, що працює на основі 2 мікроконтролерів PIC16F876, PIC16F883.

Робота приладу забезпечується модулем ADC, граничні значення для якого будуть встановлюватись за допомогою кнопок.

Вивід інформації здійснюватиметься через 3-розрядний CCI(CA), що показуватиме індикацію. А також в нас є реле, яке буде контролювати роботу мережі.

Структурна схема МКС для Системи контролю температури визначає основні функціональні частини виробу та зазначена на рисунку 1.1.

Перший блок даної схеми відповідає за зчитування інформації. В подальшому, наші дані після зчитування пройдуть через два мікроконтролера по протоколу UART.

У другому блоці схеми Системи контролю мережі 220В зображено графічний вивід інформації, що буде мати вигляд індикації вимірювань, 220В.

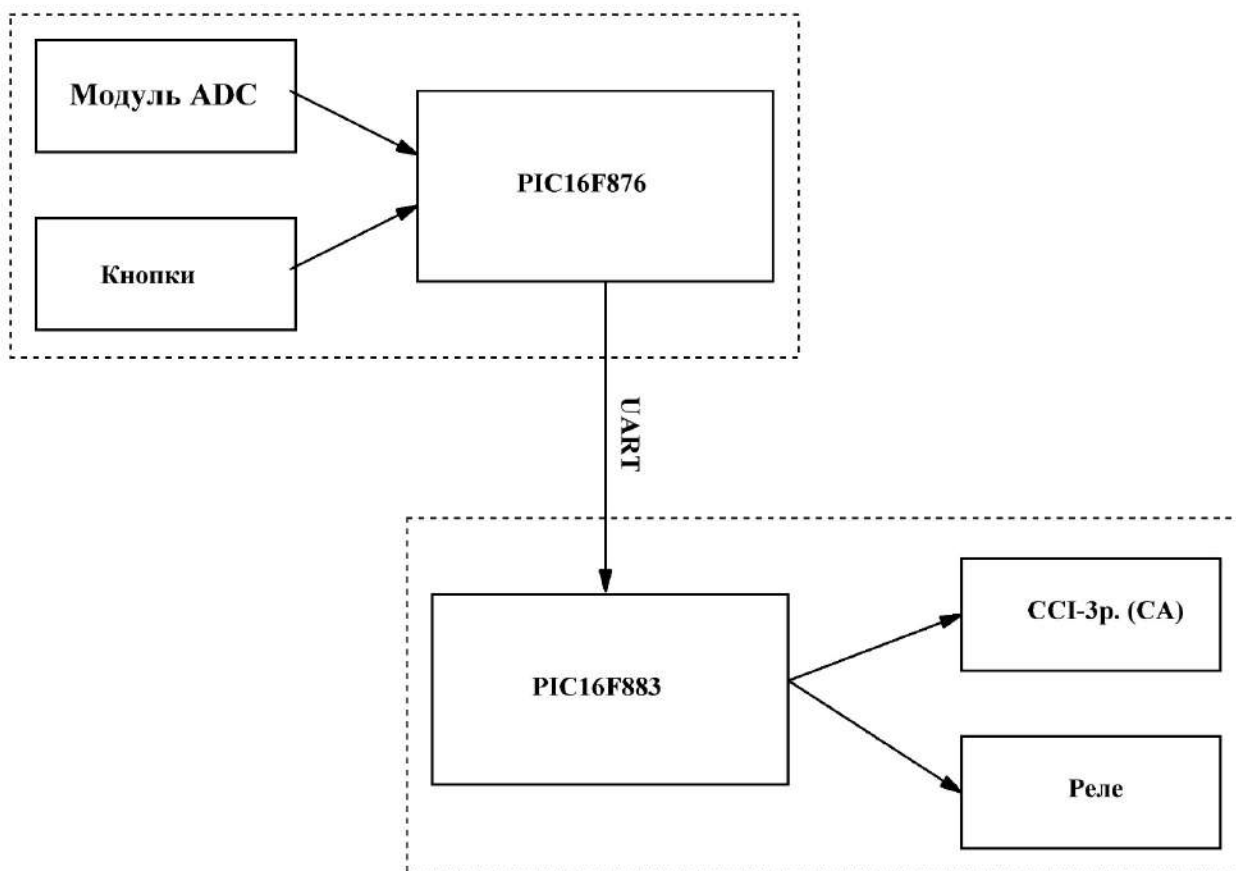


Рисунок 1.1 - Структурна схема Системи контролю мережі 220В

1.2 Вибір мікроконтролерів та аналіз їх параметрів

В останні роки при розробці систем керування все більше уваги приділяється мікроконтролерній техніці. Це пов'язано з її бурхливим розвитком і широким асортиментом пропонованої продукції. Використання мікроконтролерів дозволяє конструювати пристрої з невеликими габаритами, що є відносно дешевими, простими і надійними, сумісними з персональним комп'ютером через стандартні інтерфейси.

Основна мета при виборі мікроконтролера - обрати мікроконтролер з мінімальною ціною, але в той же час він повинен відповідати вимогам продуктивності, надійності, умовам застосування, тощо.

Наступний крок при виборі мікроконтролера – пошук моделей, які задовольняють системним вимогам. Він включає підбір літератури, технічних описів і технічних комерційних журналів, а також демонстраційні консультації. Остання стадія вибору складається з кількох етапів, мета яких - звужити список прийнятних мікроконтролерів до одного. Ці етапи включають в себе аналіз ціни, доступності, засобів розробки, підтримки виробника, стабільності та наявності

інших виробників. Проведення системного аналізу проекту дозволяє визначити вимоги до мікроконтролера:

- розрядність обчислювального ядра;
- набір вбудованих периферійних пристроїв (таймери, АЦП тощо...);
- апаратна організація обробки даних (структура машинного циклу, співвідношення тактів ГТІ і машинних циклів);
- можливість роботи по перериванню, за зовнішніми сигналами готовності або по командам людини;
- кількість керованих портів введення/виводу, характер передачі - байт або біт, програмне налаштування напрямку передачі;
- тип пристроїв введення/виводу, якими повинен керувати обирають мікроконтролер в проєктованій системі (термінали, вимикачі, реле, клавіші, давачі, цифрові пристрої візуальної індикації, аналого-цифрові й цифро-аналогові перетворювачі, модулятори тощо);
- підтримувані способи завантаження програм в мікроконтролер, - можливість внутрішньосистемного програмування (ISP), використання при цьому стандартизованих інтерфейсів (SPI, I2C);
- кількість і тип напруги живлення;
- малогабаритні та естетичні обмеження;
- умови навколишнього середовища, необхідні для експлуатації.

Мікроконтролери PIC16F876 та PIC16F883 мають режими енергозбереження і можливість захисту коду програми.

У мікроконтролери вбудований сторожовий таймер WDT, який може бути вимкнений тільки в бітах конфігурації мікроконтролера. Для підвищення надійності сторожовий таймер WDT має власний RC генератор.

Додаткових два таймера виконують затримку старту роботи мікроконтролера. Перший, таймер запуску генератора (OST), утримує мікроконтролер в стані скидання, поки не стабілізується частота тактового генератора. Другий, таймер включення живлення (PWRT), спрацьовує після включення живлення і утримує мікроконтролер в стані скидання протягом 72мс (типове значення), поки не стабілізується напруга живлення. У більшості додатків ці функції мікроконтролера дозволяють виключити зовнішні схеми скидання.

Режим SLEEP призначений для забезпечення наднизького енергоспоживання. Мікроконтролер може вийти з режиму SLEEP по сигналу зовнішнього скидання, по переповненню сторожового таймера або при виникненні переривань.

Вибір режиму роботи тактового генератора дозволяє використовувати мікроконтролери в різних додатках. Режим тактового генератора RC дозволяє зменшити вартість пристрою, а режим LP знизити енергоспоживання. Біти

конфігурації мікроконтролера використовуються для вказівки режиму його роботи.

1.3 Опис периферійних пристроїв та робота з ними

1.3.1 Робота з модулем АЦП

Аналого-цифровий перетворювач - пристрій, що перетворює вхідний аналоговий сигнал в дискретний код (цифровий сигнал). Розрядність АЦП характеризує кількість дискретних значень, які перетворювач може видати на виході. Вимірюється в бітах. Наприклад, АЦП, здатний видати 256 дискретних значень (0..255), має розрядність 8 бітів, оскільки

$$2^8 = 256.$$

Розрядність може бути також визначена в величинах вхідного сигналу і виражена, наприклад, в вольтях. Розрядність за напругою дорівнює напрузі, що відповідає максимальному вихідному коду, який ділиться на кількість вихідних дискретних значень. Наприклад:

Діапазон вхідних значень = від 0 до 10 вольт

Розрядність АЦП 12 бітів: $2^{12} = 4096$ рівнів квантування

Розрядність за напругою: $(10-0)/4096 = 0.00244$ вольт = 2.44 мВ

Аналоговий сигнал є безперервною функцією часу, в АЦП він перетворюється в послідовність цифрових значень. Оскільки реальні АЦП не можуть провести аналого цифрове перетворення миттєво, вхідне аналогове значення повинне утримуватися постійним принаймні від початку до кінця процесу перетворення (цей інтервал часу називають часом перетворення). Це завдання вирішується шляхом використання спеціальної схеми на вході АЦП - пристрою вибірки-зберігання (ПВЗ). ПВЗ зберігає вхідну напругу в конденсаторі, що сполучений з входом через аналоговий ключ: при замиканні ключа відбувається вибірка вхідного сигналу (конденсатор заряджається до величини вхідної напруги), при розмиканні - зберігання. Більшість АЦП виконані у вигляді інтегральних мікросхем і містять вбудовані ПВЗ.

Для побудови електронних АЦП використовують наступні типи АЦП:

- АЦП прямого перетворення або паралельний АЦП;
- АЦП послідовного наближення або АЦП з порозрядним врівноваженням;
- АЦП диференціального кодування;
- АЦП порівняння з зубчастим сигналом;
- АЦП з урівноваженням заряду;
- конвейєрні АЦП;
- Сигма-Дельта АЦП;

Коли необхідна роздільча здатність 12, 14 або 16 розрядів і не потрібна висока швидкість перетворення, а визначальними факторами є невисока ціна й низьке енергоспоживання, то застосовують АЦП послідовного наближення.

Цей тип АЦП найчастіше використовується в різноманітних вимірювальних приладах і в системах збору даних. У даний момент АЦП послідовного наближення дозволяє вимірювати напругу з точністю до 16 розрядів із частотою дискретизації від 100к до 1М відліків/сек.

На рисунку 1 показано спрощену блок-схему АЦП послідовного наближення. АЦП послідовного наближення містить компаратор, допоміжний ЦАП і регістр послідовного наближення. На початку циклу перетворення всі виходи цього регістра встановлюються в логічний 0, за винятком першого (старшого) розряду. Це формує на виході внутрішнього цифро-аналогового перетворювача (ЦАП) сигнал, значення якого дорівнює половині вхідного діапазону АЦП. А вихід компаратора перемикається в стан, що визначає різницю між сигналом на виході ЦАП і вимірюваною вхідною напругою.

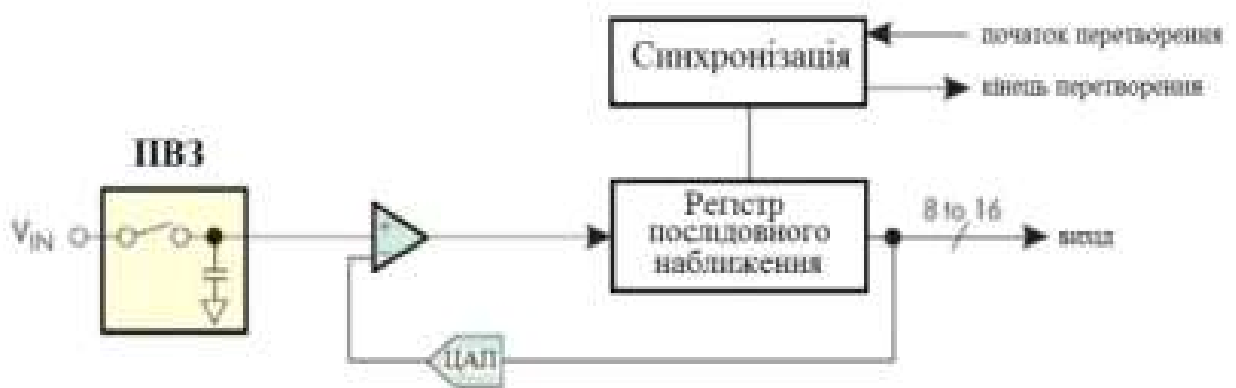


Рисунок 1.2 - АЦП послідовного наближення

АЦП перетворює аналоговий сигнал в цифровий за N кроків, де N — розрядність АЦП. На кожному кроці визначається по одному біту шуканого цифрового значення, починаючи від СЗР і закінчуючи МЗР. Послідовність дій за визначенням чергового біта полягає в наступному:

На допоміжному ЦАП виставляється аналогове значення, утворене з бітів, вже визначених на попередніх кроках; біт, який повинен бути визначений на цьому кроці, виставляється в 1, більш молодші біти встановлені в 0. Отримане на допоміжному ЦАП значення порівнюється з вхідним аналоговим значенням. Якщо значення вхідного сигналу більше значення на допоміжному ЦАП, то визначуваний біт отримує значення 1, інакше 0. Таким чином, визначення підсумкового цифрового значення нагадує двійковий пошук. АЦП цього типу характеризуються одночасно високою швидкістю і хорошою розрядністю. Таким чином АЦП послідовного наближення, потрібно один внутрішній такт перетворення для кожного розряду, або N тактів для N -розрядного перетворення.

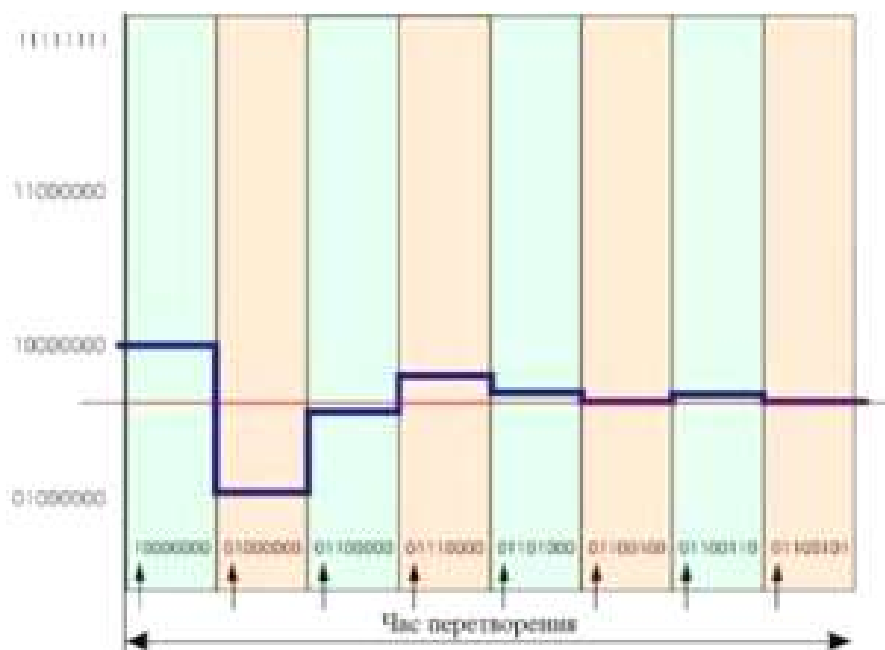


Рисунок 1.3 - Перетворення АЦП послідовного наближення

Проте, робота АЦП послідовного наближення має особливість, пов'язану з перехідними процесами у внутрішньому ЦАП. Теоретично, напруга на виході ЦАП для кожного з N внутрішніх тактів перетворення повинна встановлюватися за однаковий проміжок часу. Але насправді цей проміжок у перших тактах значно більше, ніж в останніх. Тому час перетворення 16-розрядного АЦП послідовного наближення більше ніж у два рази перевищує час перетворення 8-розрядного АЦП даного типу.

1.3.2 Семисегментні індикатори

Для відображення багатосимвольної інформації використовуються лінійні (однорядкові) дисплеї. Такі дисплеї є світлодіодними або рідкокристалічними.

Крім одиничних світлових індикаторів у мікропроцесорних системах часто застосовують знакосинтезуючі матриці, які ще називають цифровими індикаторами. Найпростішим прикладом цифрового індикатора може бути семисегментний індикатор (ССИ). Він є матрицю із семи світлодіодів, розмішених таким чином, що, засвічуючи їх у різних комбінаціях можна відобразити будь-яку десяткову цифру (від 0 до 9) та деякі латинські літери. Доповнюють його восьмим маленьким сегментом, що призначений для відображення десяткової коми (децимальна крапка). Розташувавши в ряд кілька таких індикаторів, можна відобразити будь-яке десяткове число із плаваючою комою.

Умовно кожен світлодіод в індикаторі має свій буквенний ідентифікатор (a, b, c, d, e, f, g, h), і одна з ніжок світлодіода підключена до відповідного зовнішнього виводу. Інакші ніжки з'єднані разом і підключені до загального виводу. Даний метод позначення виводів ССИ дозволяє визначити тип

індикатора – спільний анод (СА) чи спільний катод (СК). Якщо використовується ССІ типу СК то для засвічення сегментів необхідно на них високий рівень сигналу, а на загальний електрод – низький. При використанні ССІ типу СА сигнали інвертуються. Позначення ніжок ССІ та особливості ССІ типів СА і СК наведено на рисунку 1.4.

Підключення одного індикатора до МК: індикатор підключають як сім або вісім (якщо використовується децимальна крапка) незалежних світлодіодів.

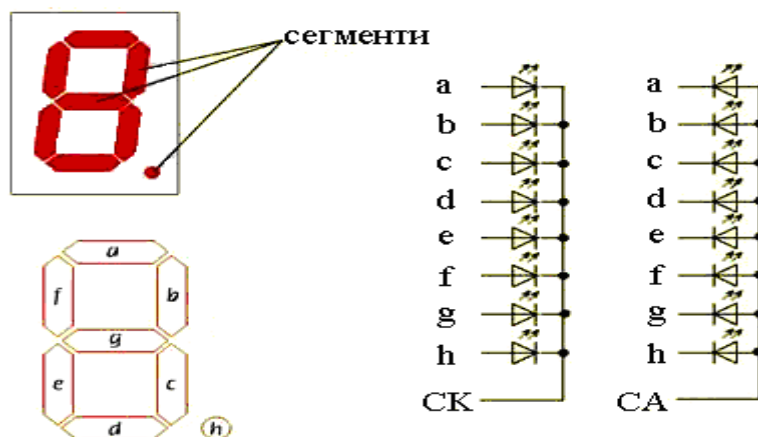


Рисунок 1.4 – Умовне графічне зображення ССІ

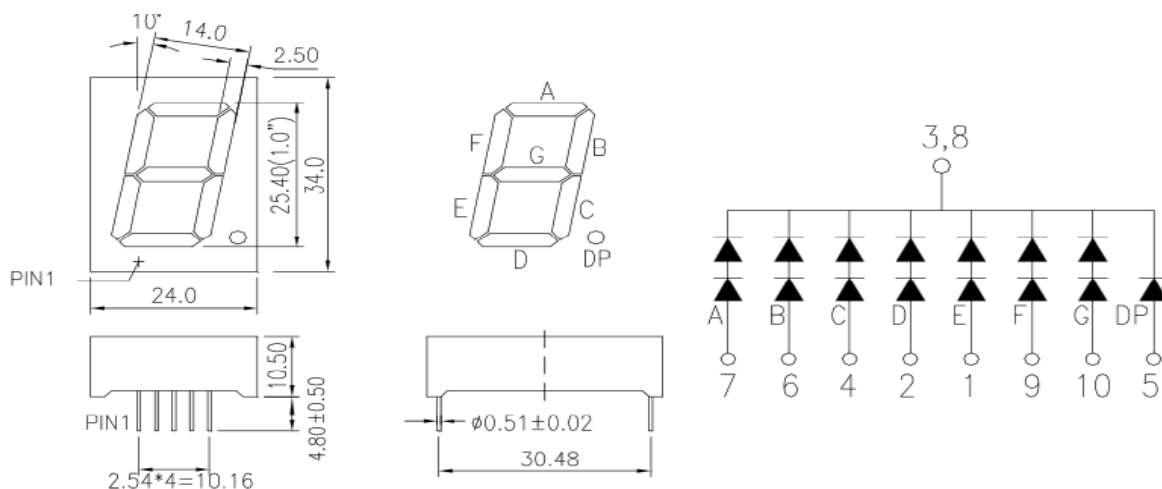


Рисунок 1.5 - Геометричні розміри та електрична схема однорозрядного ССІ зі спільним анодом.

Типовий спосіб підключення декількох індикаторів полягає в тому, щоб включити їх паралельно й потім керувати протіканням струму через загальні виводи окремих індикаторів. Через те, що величина цього струму перевищує припустиме значення вихідного струму МК, то для керування струмом включаються додаткові транзистори, які вибирають, котрий з індикаторів буде перебувати в активному стані. Семисегментним індикатором можна керувати статично або динамічно. При статичному керуванні розряди індикатора

підключені до мікроконтролера незалежно й інформація на них виводиться постійно. Суть статичної індикації полягає в постійному підсвічуванні індикатора від одного джерела. Тобто необхідно підключити індикатор до портів мікроконтролера, налаштувати ці порти на вихід і записати в ці порти дані так, щоб на індикаторах висвітилося щось зрозуміле, і відповідно міняти значення в портах при необхідності.

Щоб зменшити кількість необхідних виводів, застосовується динамічна індикація. Сутність динамічної індикації полягає в почерговому включенні індикаторів через загальний ланцюг перетворення коду. Підключення індикаторів необхідно робити із частотою $f=120\ldots140$ Гц, щоб уникнути мерехтіння індикаторів.

Схема з динамічною індикацією споживає менший струм, має менші габарити й меншу вартість. Умовну схему під'єднання ССІ представлено на рисунку 1.6.

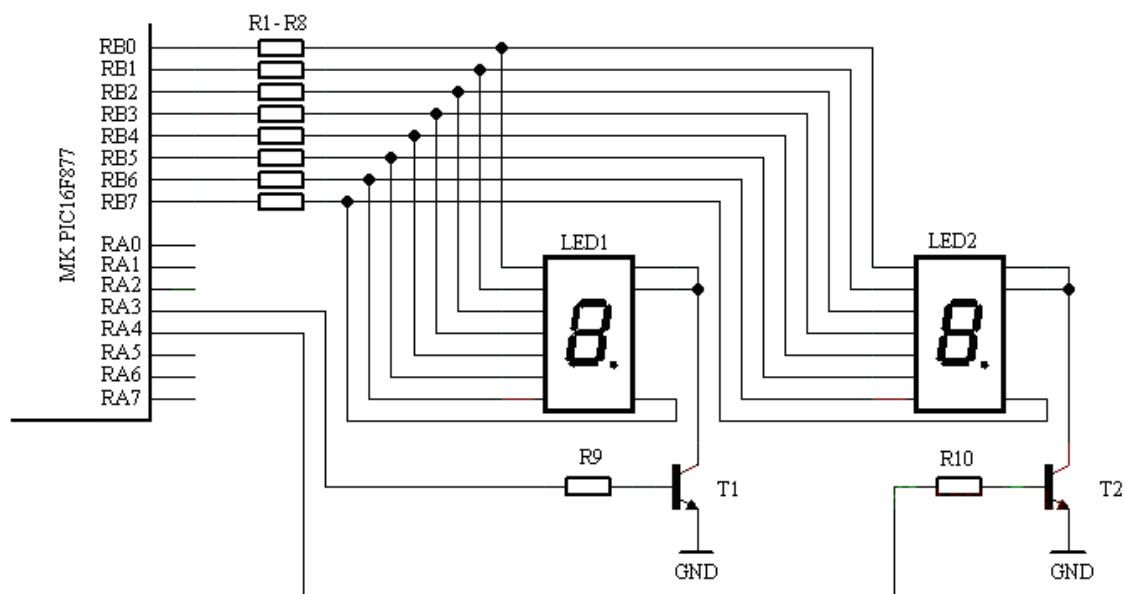


Рисунок 1.6 – Умовна схема під'єднання ССІ при динамічній індикації

Параметри 7-сегментного 1-розрядного світлодіодного індикатора наведені у відповідних технічних описах. Щоб засвітити на індикаторі якусь цифру потрібно налаштувати порти, до яких підключений індикатор, у режим виходу, "відкрити" транзистор (у цьому випадку подати на базу "одиницю") і встановити в порту мікроконтролера код бажаної цифри.

1.3.3 Робота з протоколом UART

UART (Universal Asynchronous Receiver Transmitter) представляє собою універсальний асинхронний прийомопередавач. Він призначений для послідовної передачі даних. Головною областю застосування асинхронної

передачі даних, як правило, є не обмін даними в складі схеми, а комунікація між блоками розділеними в просторі, які мають ознаки власного інтелекту. Прикладом може служити зв'язок між персональним комп'ютером і принтером, модемом, програмуючим пристроєм або регістратором даних.

При асинхронній передачі даних тактовий сигнал не передається на відміну від випадку синхронної послідовної передачі даних через SPI інтерфейс. Це висуває строгі вимоги до розподілу інтервалів часу в каналах передачі та прийому.

У найпростішому випадку для комунікації між двома пристроями досить мати просте з'єднання за допомогою трьох ліній: передаючі й приймаюльні лінії Tx0 й Rx0, а також лінії заземлення.

В стандарті RS-232 визначені високий і низький рівень напруги. Було прийнято, що в інтерфейсі RS-232 будуть діяти цифрові рівні +3 і -3В (рисунок 1.7). Напруги вище 3В приймають за сигнали логічного нуля, а будь-які напруги нижче -3В – логічною одиницею. Напруга в діапазоні від -3В до +3В вважається сигналами без змісту.

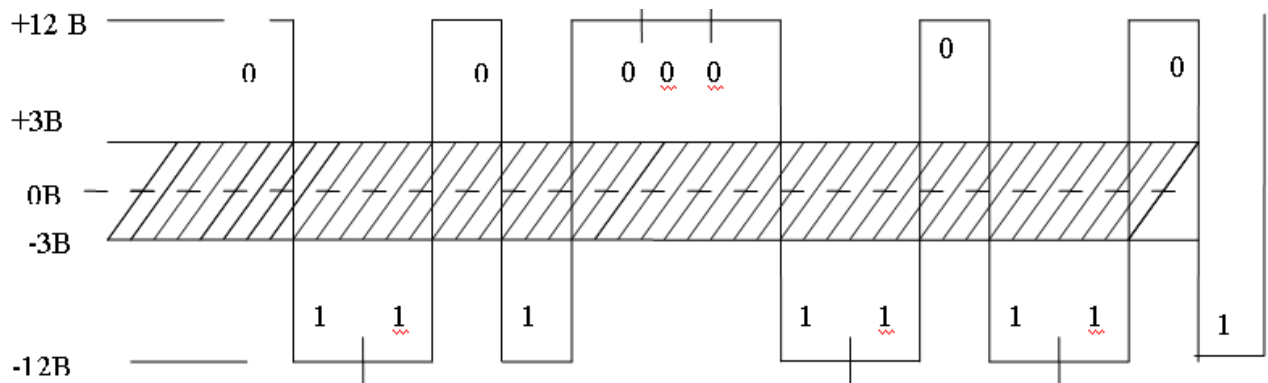


Рисунок 1.7 - Реалізація інтерфейсу RS-232 на фізичному рівні

По причині відсутності синхронізації, яка показує початок і кінець кожного байту, байт даних приходить починати стоповим і закінчувати стартовим бітами, щоб приймальні пристрої могли розпізнавати початок і кінець передачі байту даних. Стартовий біт знаходиться на початку кожного байту. В лінії передачі в режимі очікування діє високий рівень напруги. При виводі байта в лінії встановлюється низький рівень. Інтервал стартового сигналу такий як у біта даних.

Відповідно до визначення стандарту V.24, у лінії передачі даних у стані очікування встановлений рівень лог. 1. Передача може бути почата в будь-який момент часу. Для того щоб передати приймачу повідомлення про початок передачі, посилається стартовий біт з рівнем лог.0. Після цього передаються розряди даних (у більшості випадків 7 або 8, їхнє число зумовлюється в протоколі передачі даних), при цьому спочатку передається молодший значущий біт.

Для підвищення надійності передачі даних може бути доданий біт парності. Біт парності доповнює розряди даних таким чином, що досягається пряма парність (сума всіх бітів, включаючи біт парності, парний) або непряма парність (сума всіх бітів, включаючи біт парності, непарна). Необхідний вид парності (відсутність, пряма чи непряма) також повинен бути визначений у протоколі передачі даних.

Передача закінчується стоп-бітом з рівнем лог. 1. Після відправлення стоп-біта лінія даних знову переходить у вихідний стан очікування й готова до наступної передачі. Часова діаграма передачі байта E5h (11100101b) з непрямою парністю й стопбітом через інтерфейс типу V.24 показана на рисунку 1.8.

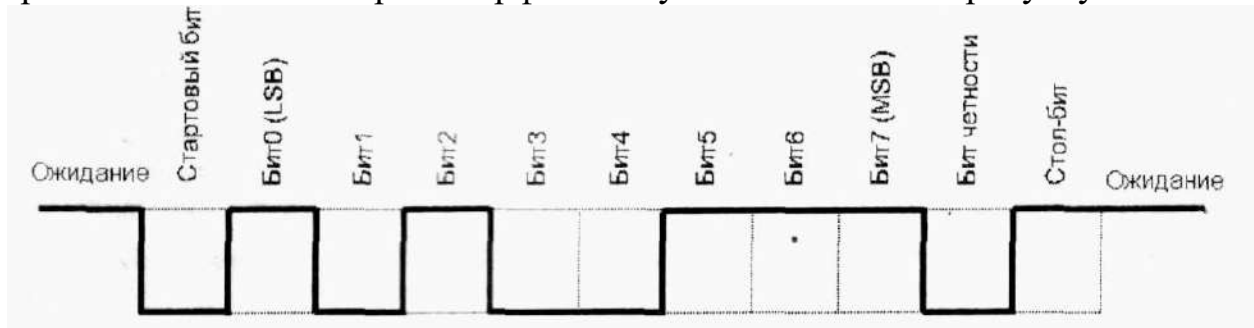


Рисунок 1.8 - Часова діаграма передачі байта E5h (11100101b).

У зв'язку з тим, що такт для синхронізації передачі не пересилається, початок розряду може бути розпізнано тільки по вимірюванні часу, що пройшов з моменту появи спадаючого фронту стартового біта. Саме тому необхідно визначати швидкості передачі даних, які є обов'язковими для передавальної й приймаючої сторони. У стандарті RS232 логічна одиниця називається "mark", при цьому рівень сигналу на лінії (-5..-15В). Логічний нуль у стандарті RS232 називається "space", йому відповідає рівень сигналу +5..+15В. У відсутності передачі на лінії завжди логічна одиниця. Про початок передачі даних сигналізує старт біт (логічний нуль). Кінець передачі даних позначається стоп бітом (логічна одиниця).

1.3.4 Модуль TMR0

Модуль TMR0 - це апаратно реалізований 8-розрядний таймер/лічильник з можливістю читання й запису поточного значення регістра-лічильника. Реалізований на основі програмно доступного 8-розрядного регістра TMR0, який знаходиться в 0-вому банку пам'яті даних за адресою 01h.

Модуль TMR0 конструктивно містить 8-розрядний програмований попередній дільник. Працює на основі сигналів внутрішнього або зовнішнього джерела тактового сигналу, при переповненні (перехід від FFh до 00h) формує сигнал переривання TOIF, який може бути використаний для зупинки основної програми та переходу на обробку підпрограми переривання.

Керування роботою модуля TMR0 здійснюється з використанням програмно доступного регістра OPTION, який знаходиться в 1-ому банку пам'яті даних за адресою 81h. Регістр OPTION доступний для читання й запису, містить біти керування:

біт 5: T0CS: Вибір тактового сигналу для TMR0

1= зовнішній тактовий сигнал з виводу RA4/T0CKI

0 = внутрішній тактовий сигнал CLKOUT

біт 4: T0SE: Вибір фронту збільшення TMR0 при зовнішньому тактовому сигналі

1= збільшення по задньому фронті сигнала на виводі RA4/T0CKI

0 = збільшення по передньому фронті сигнала (до високого рівня)

біт 3: PSA: Вибір включення попереднього дільника

1= попередній дільник включений перед WDT

0 = попередній дільник включений перед TMR0

біти 2-0: PS2: PS0: встановлення коефіцієнта ділення попереднього дільника

Біти PS0,PS1,PS2 визначають коефіцієнт попереднього дільника. (Попередній дільник - це послідовний ланцюжок з 8-ми тригерів, кожний з яких ділить на 2. Таким чином, максимальний коефіцієнт поділу = 256, і він, у межах дозволених таблицею значень, може задаватися значеннями бітів PS0,PS1,PS2. Попередній дільник може бути включений або перед таймером TMR0, або після сторожового таймера WDT. Це визначає 3-й біт регістра OPTION (PSA).

Через кожні 256 імпульсів (при переході зі стану FFh у стан 00h) відбувається переповнення таймера (перехід на нове кільце рахунку), кількість яких (при необхідності підрахунку кількості імпульсів більшого, ніж 256) підраховується. Наприклад: за 1сек. відбулося 10 переповнень, значить TMR0 порахував 2560 імпульсів. Якщо приплюсувати до цієї кількості вміст TMR0 на момент закінчення рахунку, то одержимо точну кількість імпульсів, що надійшли на вхід TMR0 за 1сек. Якщо перед TMR0 включений попередній дільник, то в підсумок підрахунку вносяться відповідні корективи, що зумовлені заданим коефіцієнтом поділу попереднього дільника й числом, що знаходиться у попередньому дільнику на момент закінчення рахунку.

На такому підрахунку й ґрунтується принцип роботи пристроїв, що роблять облік імпульсів за заданий інтервал часу.

4-тим бітом регістра OPTION (TOSE) встановлюється момент спрацьовування таймера TMR0 від зовнішньої імпульсної послідовності. При встановленні біта TOSE в 1, інкремент відбувається по спадах імпульсної

послідовності (перехід від 1 до 0), що є присутнім на виводі RA4/TOCKI, а при TOSE = 0 - по фронтах (перехід від 0 до 1).

Значення 5-го біта регістра OPTION (TOCS) визначає, який сигнал буде подаватися на вхід TMR0: або зовнішній сигнал з рахункового входу RA4/TOCKI (біт установлюється в 1), або внутрішній тактовий сигнал CLKOUT (біт установлюється в 0). Модуль TMR0 може працювати в одному з 2-х режимів: таймера або лічильника.

Переривання у модулі TMR0 відбувається тоді, коли формується переповнення регістра таймера/лічильника при переході від FFh до 00h. Тоді встановлюється біт запиту T0IF у регістрі INTCON<2>. Дане переривання можна замаскувати бітом T0IE у регістрі INTCON<5>. Біт запиту T0IF повинен бути скинутий програмно при обробці переривання.

Переривання по TMR0 не може вивести процесор з режиму SLEEP тому, що таймер у цьому режимі не функціонує.

При PSA=1 дільник буде приєднаний до сторожового таймера як попередній дільник (дільник на виході). При використанні попереднього дільника разом з TMR0, всі команди, що змінюють вміст TMR0, обнуляють попередній дільник. Якщо попередній дільник використовується разом з WDT, команда CLRWDT обнуляє вміст попереднього дільника разом з WDT.

Попередній дільник є 8-бітним лічильником, який використовується як дільник частоти перед TMR0 або після Watchdog таймера.

Якщо дільник з'єднаний з TMR0, то він не може бути підключений до Watchdog таймера і навпаки. Сторожовий таймер WDT це RC-очікуючий мултивібратор з перезавпуском, що формує імпульс тривалістю приблизно 18мс. Якщо робота WDT дозволена, то після старту програми, він запускається, і якщо його, в інтервалі часу до 18мс., не запустити знову, то він закінчить формування імпульсу, а по його задньому фронті, сформується сигнал RESET. У випадку "зависання" програми, WDT, скинувши програму на початок, може вивести її із цього стану. Для забезпечення цього сторожового режиму, у ході виконання програми, необхідно періодично скидати WDT (не допускати його спрацювання).

Якщо після WDT поставити попередній дільник, то період скидання WDT можна збільшити (це залежить від заданого Кділ попереднього дільника)

Попередній дільник може підключатись по-різному. Коли він приєднаний до TMR0, всі команди, що здійснюють запис в TMR0 (наприклад, CLRF 1, MOVWF 1, BSF 1, тощо...) будуть одночасно обнулювати дільник. Коли дільник підключений до WDT, команда CLRWDT також обнулить дільник разом з Watchdog таймером. Підключення дільника – програмно кероване.

2 ПРОЕКТУВАННЯ ЕЛЕКТРИЧНОЇ ПРИНЦИПОВОЇ СХЕМИ

2.1 Розробка схеми електричної принципової мікроконтролера

У процесі проектування будь-якого пристрою, формується набір технічної документації по цьому пристрою. В технічній документації описується параметри, робота пристрою, необхідні для виготовлення компоненти та методи і засоби налагодження функціональних можливостей проектованого пристрою.

Одним із найважливіших розділів технічної документації являється подання електричної принципової схеми проектованого пристрою.

Електрична принципова схема – це графічне зображення будови пристрою на фізичному рівні. Дана схема відображає кількість та тип використовуваних у пристрої компонентів (радіотехнічних елементів, мікросхем, інтерфейсних роз'ємів, тощо...) та їх взаємодію один з одним. Підключення компонентів схеми один до одного відображається лініями. Згідно цієї схеми відбувається розведення доріжок на платі у відповідності до ліній підключення, зображених на схемі, та монтаж і пайка елементів проектованого пристрою. Усі елементи, зображені на електричній принциповій схемі, підписуються згідно вимог ДСТУ.

Жоден електричний пристрій не може працювати без джерела напруги живлення. Пристрої, що побудовані на основі мікроконтролера, це цифрові пристрої. Діапазон напруги живлення для таких пристроїв складає від 1.5 до 5.5В, а струм - постійний. Для підключення цифрових пристроїв до мережі 220В використовуються трансформаторні, імпульсні, конденсаторні блоки живлення тощо... До складу найпростішої схеми входять: понижуючий трансформатор, діодний міст, електролітичні конденсатори та мікросхема стабілізації (див. рисунок 2.1).

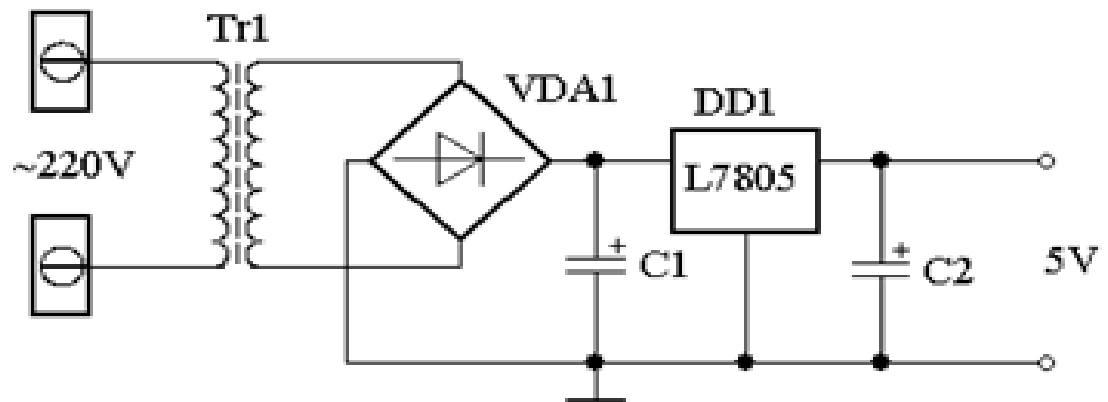


Рисунок 2.1- Схема стабілізації живлення для цифрових пристроїв на основі мікроконтролерів

Ключовим елементом в даній схемі є стабілізатор L7805. Існує два види стабілізатора 7805: з струмом навантаження до 1А і малопотужний 78L05 з

струмом навантаження до 0,1А. Крім них є проміжний варіант 78M05 з струмом навантаження до 0,5А.

Ємність $C1$ на вході стабілізатора необхідна для згладження високочастотних завад при подачі вхідної напруги. Ємність $C2$ на виході стабілізатора задає стабільність напруги при різкій зміні струму навантаження, а також суттєво знижує амплітуду пульсацій. Для його нормальної роботи напруга на вході повинна бути у діапазоні 10-20 В.

Для формування напруги такого рівня використовується однофазний трансформатор та діодний міст. Для того, щоб сформувати необхідні умови для роботи МКС необхідно передбачити джерело імпульсів синхронізації.

В мікроконтролерах сімейства PIC16 передбачено декілька типів генераторів. Для PIC16 користувач може запрограмувати два конфігураційних біти (FOSC1 і FOSCO) для вибору одного із чотирьох режимів: RC, LP, XT, HS., де:

RC – генератор на RC-ланці;

XT – стандартний кварцовий генератор;

HS – високочастотний кварцовий генератор;

LP – низькочастотний малої потужності генератор.

В режимах XT, LP і HS до виводів OSC1/CLKIN і OSC2/CLKOUT підключається кварцовий або керамічний резонатор.

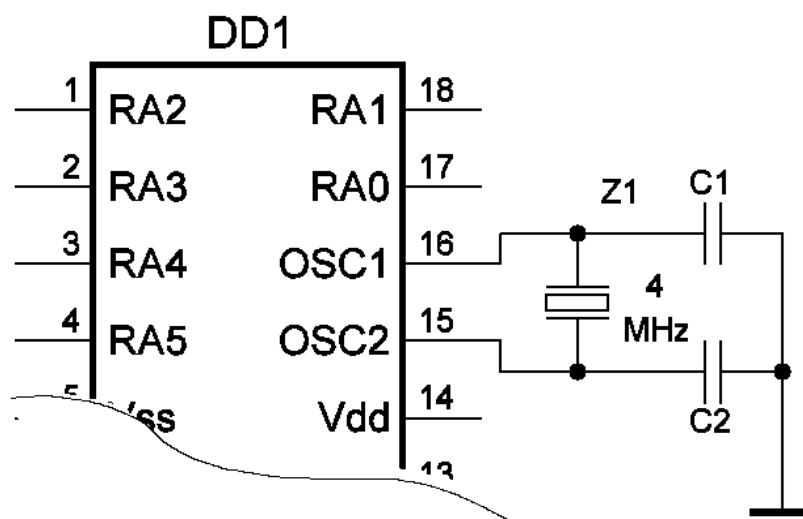


Рисунок 2.2 - Схема тактування мікроконтролера

Для підключення резонатора використовується типова схема виробника. Ємність конденсаторів повинна бути в інтервалі від 15 до 22 пФ, один вивід яких підключається до резонатора, а інший до землі.

У пристроїв, побудованих на основі мікроконтролерів передбачений механізм апаратного ресетування. У мікроконтролерів PIC16 вивід, що призначений для ручного перезапуску називається /MCLR. Активний сигнал

для скидання – це сигнал логічного 0, що подається на вхід /MCLR при натисканні кнопки S1.

Для зміни програмного забезпечення МК використовується роз'єм ICSP. Даний роз'єм, як і одноіменний протокол обміну, дозволяє змінювати робочі параметри мікроконтролера без вилучення його із пристрою. При прошиванні мікроконтролера на ніжку /MCLR подається сигнал рівнем 15В. Даний сигнал позначено на роз'ємі як Vpp. Задіюється механізму доступу до пам'яті мікроконтролера. Дані передаються по лінії Data на вивід PGD мікроконтролера, при цьому з лінії Clock роз'єму ICSP поступає сигнал тактування. Даний сигнал приймається виводом PGC мікроконтролера, та відповідає за коректний запис програми у мікроконтролер. Сигнали Vss і Vdd забезпечують стандартну напругу живлення для мікроконтролера рівнем +5В.

На електричній принциповій схемі показано як взаємодіють різні вузли між собою та які компоненти використовуються для побудови проектного пристрою. Згідно із цією схемою у подальшому формуватиметься принцип роботи усієї системи та алгоритм функціонування мікроконтролера, як основного вузла керування.

Пристрої введення/виведення – це різного роду периферійні модулі, які підключаються до функціонально завершеного пристрою для керування роботою цього пристрою та отримання інформації про виконуваним функції. Наприклад, для ПК такими пристроями є клавіатура, мишка (пристрої введення) та монітор і принтер (пристрої виведення).

Для мікроконтролерних систем номенклатура пристроїв введення чи виведення дещо інша. Пристроями вводу крім клавіатури (клавіатурної матриці) можуть буди різні датчики – тепла, вологості, загазованості, ІЧ-датчики, тощо... В свою чергу пристрої виводу також не обмежуються монітором. Для МКС це, – як правило, – знакосимвольні або графічні РКІ, ССІ, акустичні елементи (біпери, динаміки), навідь звичайні світлодіоди.

Найпростіші із цих пристроїв введення/виведення інтегруються в пристрій та підключаються безпосередньо до портів мікроконтролера, проте є і такі, для підключення яких використовуються інтерфейсні роз'єми.

Інтерфейс – у мікроконтролерній техніці – це набір провідників, які використовуються для підключення зовнішнього пристрою вводу/виводу та забезпечують реалізацію протоколу передачі даних. Як правило, такі групи провідників об'єднуються в межах одного роз'єму. Наприклад, для зовнішніх пристроїв, які працюють по USART/RS-232 протоколу, група провідників, яка забезпечує зв'язок із мікроконтролером основного пристрою по цьому протоколу об'єднується у роз'єм типу D-SUB9.

Наведена електрична принципова схема працює наступним чином. Після подачі живлення мікроконтролер покроково виконує програму, що закладена у його пам'ять програм. Першим етапом є ініціалізація портів мікроконтролера та

його вбудованих модулів. Активується інтерфейс зв'язку та виконується обмін даними.

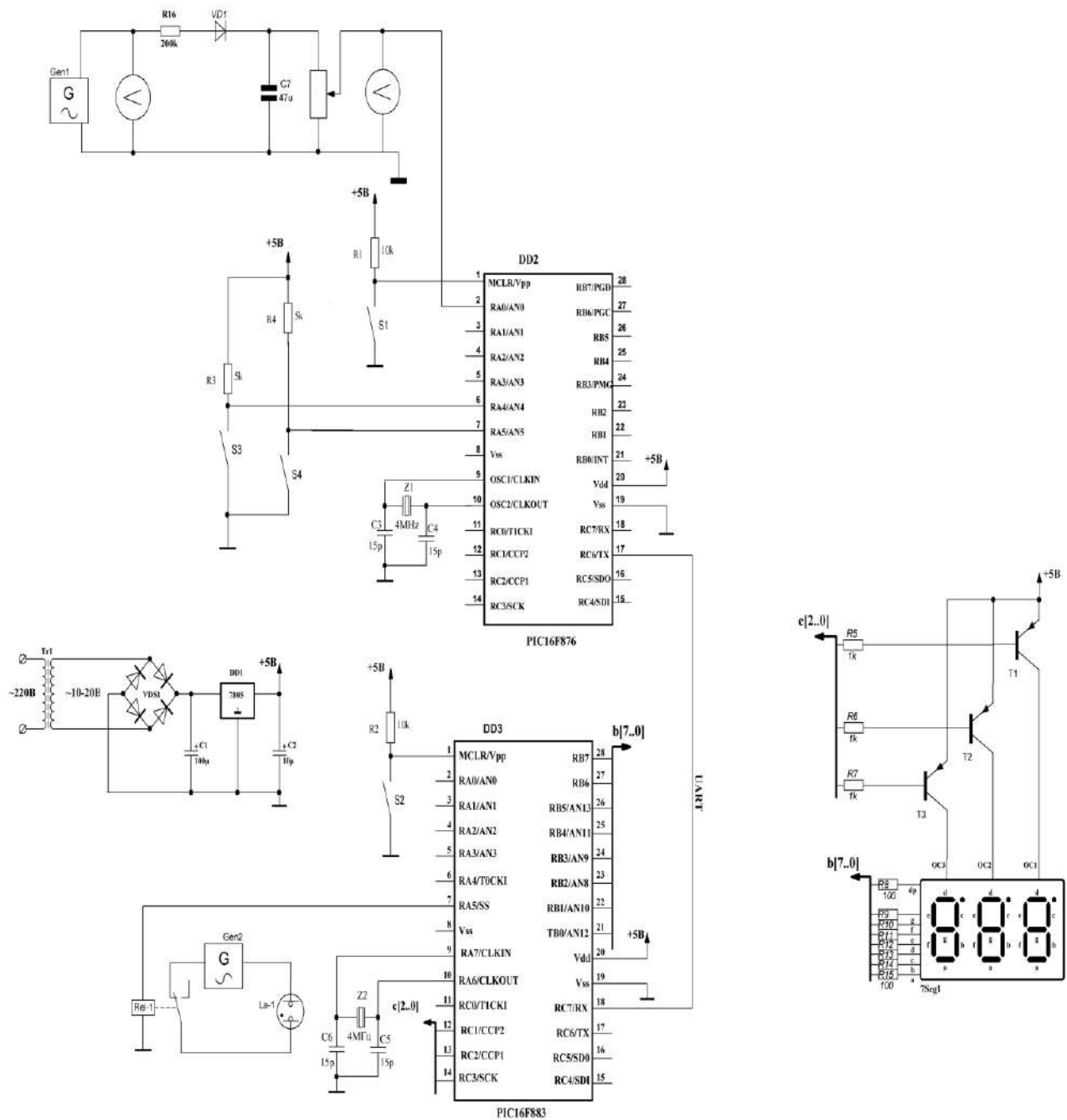


Рисунок 2.3 - Електрична принципова схема роботи Системи контролю мережі 220В

Таблиця 2.1 – Перелік елементів електричної принципової схеми

№ П/П	Назва елемента	Позначення	Номінал
1	Трансформатор	TR1	220В
2	Діодний міст	VDS1	D13A4
3	Регулятор напруги	DD1	7805
4	Кварцеві резонатори	Z1,Z2	4МГц
5	Конденсатор	C1	100мкФ
6	Конденсатор	C2	10мкФ
7	Конденсатори	C3, C4, C5, C6	10мкФ
8	Конденсатор	C7	47мкФ
9	Резистори	R1,R2	10кОм
10	Резистори	R3, R4	5кОм
11	Резистори	R5, R6, R7	1кОм
12	Резистори	R8-R15	100Ом
13	Резистор	R16	200кОм
14	Мікроконтролери	DD2,DD3	PIC16F876, PIC16F883
15	3-розрядний ССІ	7Seg1	MPX3-CA
16	Транзистори	T1, T2, T3	PNP
17	Кнопка-перемикач	S1,S2, S3, S4	SK1-53
18	Реле	Rel1	12В
19	Генератори сигналіви	Gen1, Gen2	1Гц
20	Лампа	La1	12В
21	Діод	VD1	Analog

2.2 Створення проекту в середовищі PROTEUS VSM

Більшість радіоаматорів стикалися із ситуацією, коли за браком досвіду чи присутності помилок у схемі або ж за іншими обставинами, псували радіоелектронні компоненти.

З'явилася величезна кількість програм симуляторів, що замінюють реальні радіодеталі й прилади, віртуальними моделями. Симулятори дозволяють, без збирання реального пристрою, налагодити роботу схеми, знайти помилки, отримані на стадії проектування, зняти необхідні характеристики й багато чого іншого. Однією з таких програм є PROTEUS VSM.

Proteus VSM складається із двох самостійних програм ISIS та ARES. ARES це трасувальник друкованих плат з можливістю створення своїх бібліотек. Запуск програми відбувається з меню Пуск – Програми - Proteus 7 Professional - ISIS 7 Professional. При запуску програми з'являється основне вікно програми ISIS 7 Professional.

Найбільший простір відведений під вікно редагування EDIT WINDOW. Саме в ньому відбуваються всі основні процеси створення, редагування й налагодження схеми пристрою. Ліворуч угорі маленьке вікно попереднього перегляду Overview Window з його допомогою можна переміщуватися по вікну редагування (клацаючи лівою кнопкою миші по вікну попереднього перегляду, можна переміщувати вікно редагування за схемою, якщо звичайно схема не вміщається у вікно). Переміщувати вікно редагування за схемою можна ще утримуючи натиснутої кнопку SHIFT та рухаючи курсором миші по вікну редагування. Наближувати й віддаляти схему у вікні можна кнопками F6 й F7 або ж колесом миші, F5 центрує схему у вікні, а натискання F8 підганяє розмір схеми під-вікно редагування.

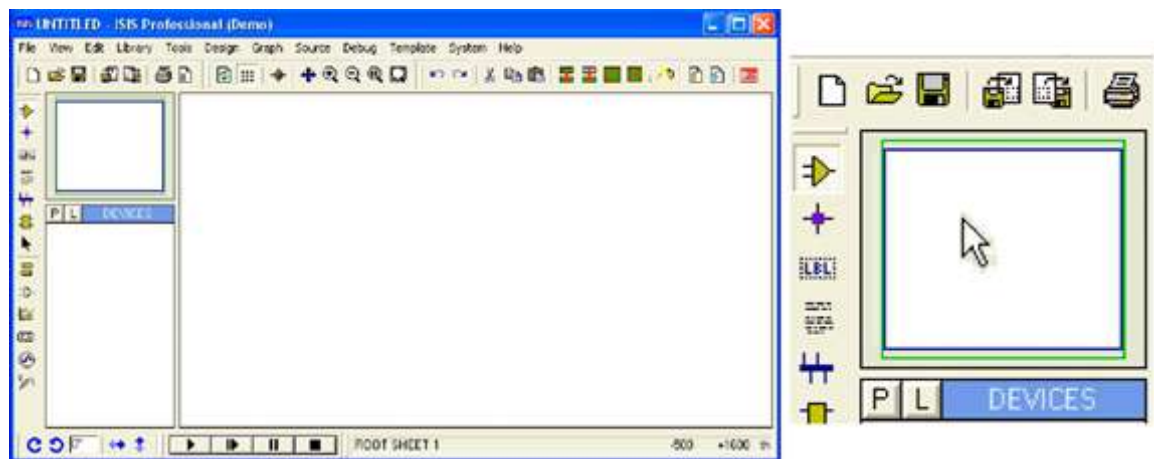


Рисунок 2.4 - Основне вікно програми ISIS 7 Professional і вікно попереднього перегляду.

Під вікном попереднього перегляду знаходиться Object Selector - список обраних у цей момент компонентів, символів й інших елементів. Виділений у списку об'єкт відображається у вікні попереднього перегляду.

Всі можливі функції та інструменти Proteus VSM доступні через головне меню, через піктограми які знаходяться під меню і у лівому куті основного вікна та через гарячі клавіші, які можуть перепризначуватися користувачем. Внизу основного вікна розташовані: кнопки обертання та розвороту об'єкта навколо своєї осі, панель керування інтерактивною симуляцією - виглядає як магнітофонна і функції такі ж: ПУСК, ПОКРОКОВИЙ РЕЖИМ, ПАУЗА,СТОП.

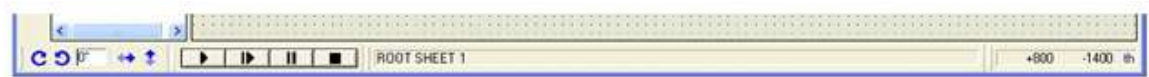


Рисунок 2.5 - Панель керування інтерактивною симуляцією.

Середовище PROTEUS має величезну бібліотеку електронних компонентів. Всі елементи перебувають у бібліотеці компонентів. Щоб до неї потрапити потрібно перейти в режим COMPONENT (компоненти), натиснувши відповідну піктограму P (Pick devices) або двічі клацнувши лівою кнопкою в полі вибору компонентів Object Selector.

Компоненти можна вибирати по категоріях, та у списку. Зображення компонента з'явиться у вікні попереднього перегляду. Щоб розмістити компонент на робочому полі потрібно навести курсор миші на потрібне місце і клацнути лівою кнопкою миші. Щоб з'єднати виводи компонентів провідником потрібно підвести курсор миші до виводу компонента, після появи на виводі компонента хрестика потрібно клацнути лівою кнопкою миші і підвести курсор миші до виводу іншого компонента.

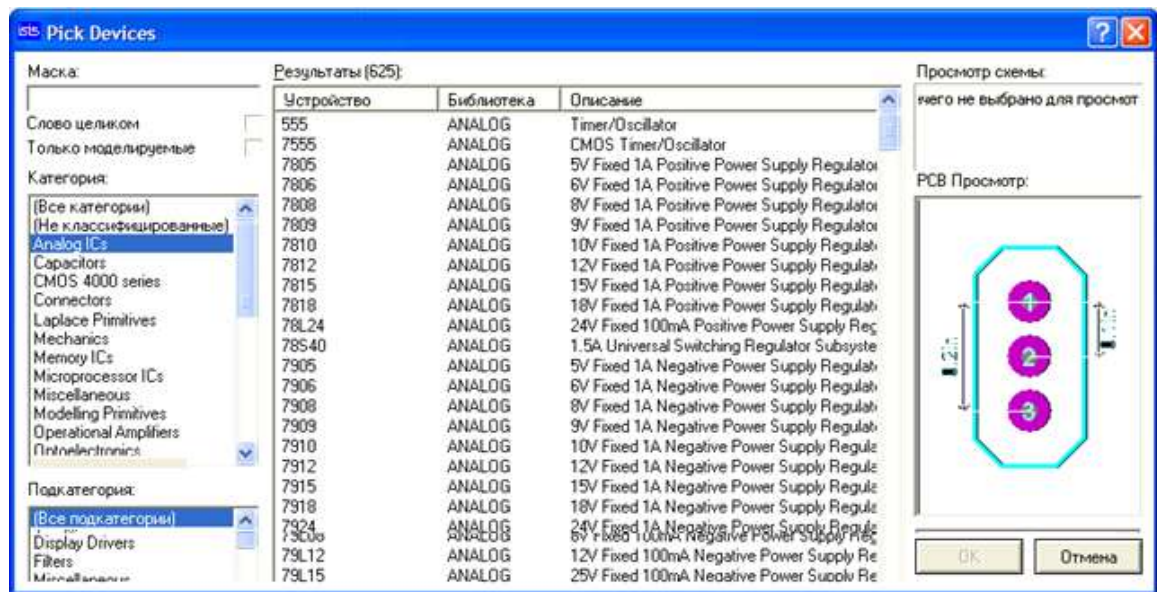


Рисунок 2.6 - Бібліотека електронних компонентів.

Для того, щоб переглянути роботу схеми, необхідний вихідний HEX-файл. Середовище PROTEUS підтримує багато засобів розробки, серед них і HI-TECH Cі компілятор і CROWHILL PIC BASIC і BASIC STAMP. Попередньо HEX-файл повинен бути створений з використанням середовищем програмування мікроконтролерів від фірми MICROCHIP, а саме MPLAB.

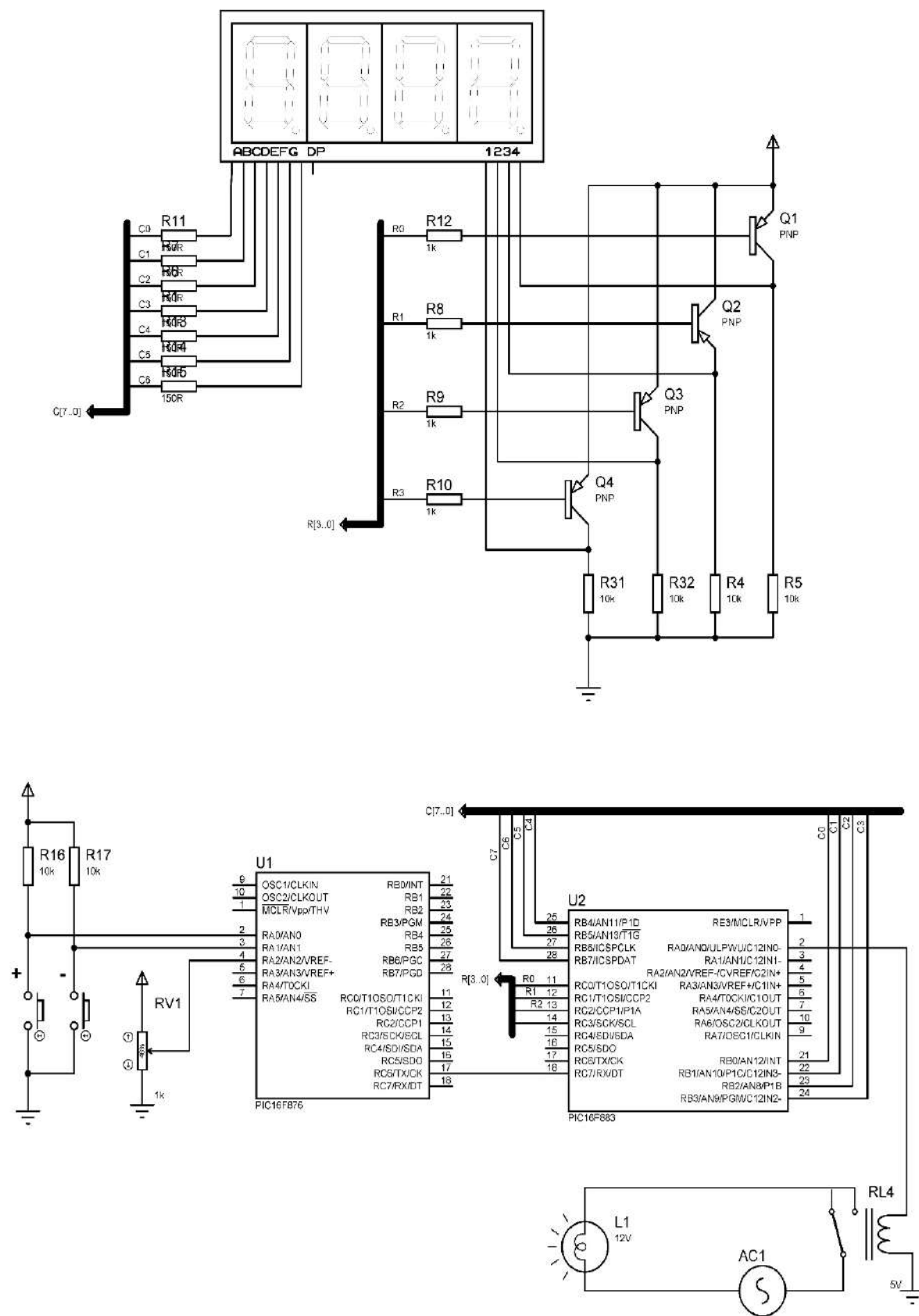


Рисунок 2.7 - Досліджувана схема

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МКС

3.1 Розробка алгоритму функціонування МКС

Алгоритм — це послідовність, система, набір систематизованих правил виконання обчислювального процесу, що обов'язково приводить до розв'язання певного класу задач після скінченного числа операцій. При написанні комп'ютерних програм алгоритм описує логічну послідовність операцій. Для візуального зображення алгоритмів використовують блок-схеми.

Блок-схема – схематичне зображення послідовності дій, спрямованих на досягнення спільної мети та їх особливостей. Розділ програми, у якому відбувається занесення значення змінних подається у вигляді паралелограма, в якому перераховані імена цих змінних. Розділ програми, у якому відбувається виконання певної операції подається у формі прямокутника, а розділ перевірки умови – у формі ромба. Крім того можуть використовуватись розділи, які повторюють певний набір команд для іншого аналогічного випадку. Такі розділи називають блоком ідентичних операцій та зображують у вигляді шестикутника. Ще один розділ, що є поширеним у програмах – це розділ виконання підпрограми. Зображується він у формі прямокутника із двома вертикальними смугами по краях фігури та записом у ньому назви підпрограми.

Кожен алгоритм є списком добре визначених інструкцій для розв'язання задачі. Починаючи з початкового стану, інструкції алгоритму описують процес обчислення, які відбуваються через послідовність станів, які, зрештою, завершуються кінцевим станом.

Принципи побудови програмного забезпечення точно такі ж, як і апаратної частини: створення кінцевого продукту з готових блоків - підпрограм. Якщо всю роботу програми розмістити у векторах переривань, то основний додаток ніяк не відчує виконання фрагментів сусідніх підпрограм. Головне визначити можливості апаратної частини.

Програмне забезпечення мікроконтролера необхідно будувати з максимальним використанням периферії. При цьому ядро не виконуватиме безлічі рутинних операцій.

Після включення живлення сервісна програма починає проводити ініціалізацію. На початку програми прикріплюється файл бібліотеки мікроконтролера, після чого оголошуються біти портів і регістри пам'яті, які використовуються в програмі. Мікроконтролер має вільні для використання комірки пам'яті і, щоб при кожному зверненні до певної комірки не описувати її адресу, та полегшити формування програми - присвоюється їй назва.

У поняття ініціалізації входить:

- присвоєння початкових значень всім змінним програми;
- налаштування всіх внутрішніх систем мікроконтролера;

- відновлення режимів роботи системи, встановлених у попередньому сеансі роботи шляхом зчитування відповідних параметрів з EEPROM-пам'яті;
- виконання підпрограм, призначених для налаштування всіх периферійних модулів у вихідний стан;
- виконання підпрограми запуску системних лічильників-таймерів і механізму переривань мікроконтролера.

Для використання у програмі констант і змінних необхідно їх попередньо описати та оголосити. Константа - це просто деяке число, що часто використовується в програмі, і має певний зміст.

Крім констант, які, відповідно до своєї назви, протягом усього часу роботи програми ніколи не міняються, у програмі існують змінні. Під змінною в програмі розуміють ім'я регістра, або адресу комірки пам'яті, де в процесі виконання програми буде тимчасово зберігатися заздалегідь невідома величина, або проміжний результат обчислень.

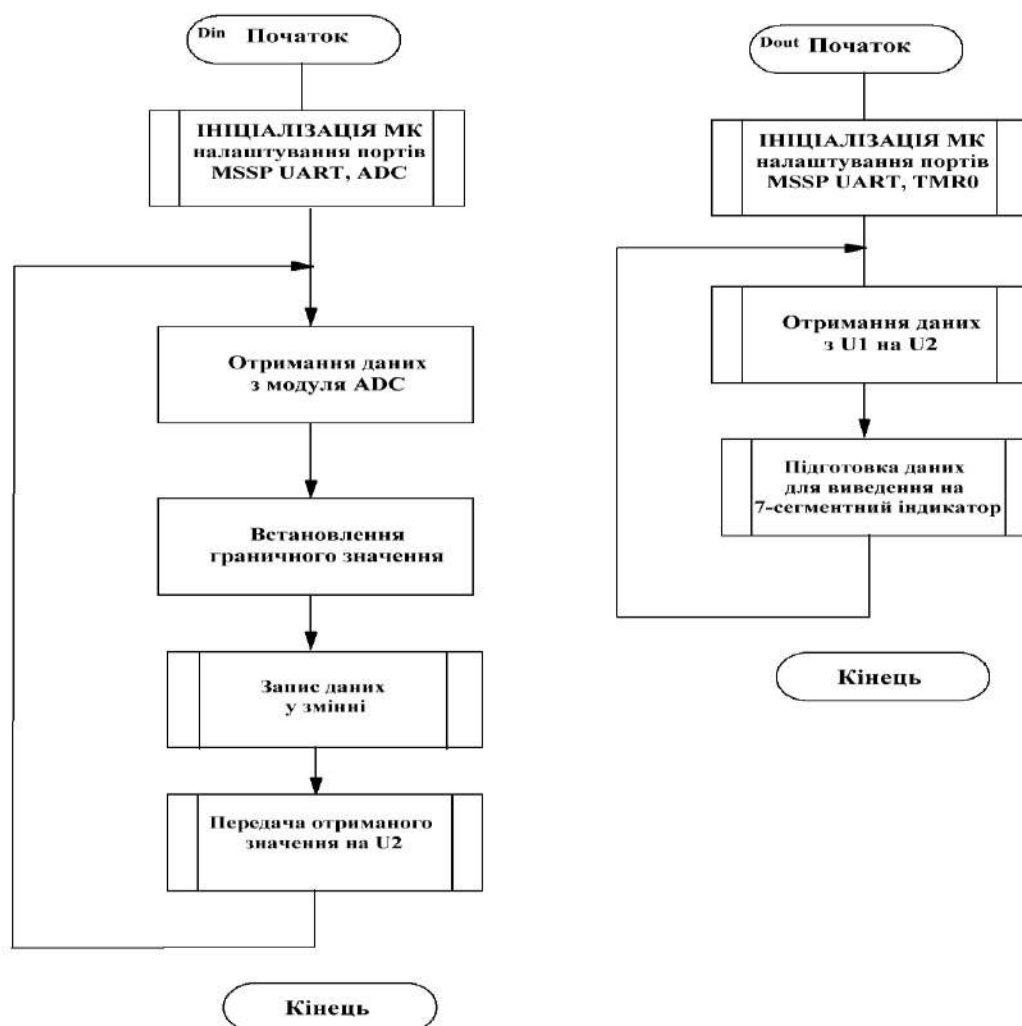


Рисунок 3.1 - Блок-схеми роботи Системи контролю мережі 220В

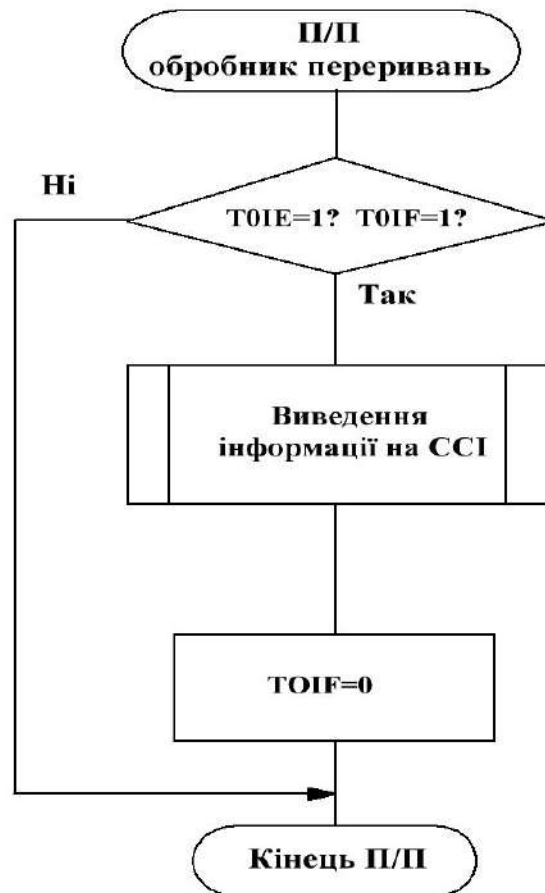


Рисунок 3.2 - Обробник переривання за TMR0

3.2 Конфігурування пристрою з використанням директиви __CONFIG

Сімейство мікроконтролерів PIC16 має набір спеціальних функцій, призначених для розширення можливостей системи, мінімізації вартості, виключення навісних компонентів, забезпечення мінімального енергоспоживання і захисту коду від зчитування. В PIC16 реалізовані наступні спеціальні функції:

- вибір типу генератора;
- скидання;
- схема скидання по включенню живлення ;
- таймер скидання (DRT);
- сторожовий таймер (WDT)
- режим пониженого енергоспоживання (SLEEP);
- захист коду від зчитування;
- біти ідентифікації.

В регістрі конфігурації описуються біти конфігурації, в які входять:

- біт захисту

- біт дозволу роботи таймера включення живлення
- біт дозволу роботи сторожового таймера
- вибір типу генератора

Біти конфігурації задаються за допомогою директиви `__CONFIG`.

Фрагмент заголовку при написання програми для мікроконтролера PIC16F876 з використанням мови C:

```
#include <pic.h>
__CONFIG(FOSC_HS&WDTE_OFF&PWRTE_OFF&CP_OFF&BOREN_OFF&L
VP_OFF&CPD_OFF&WRT_OFF&DEBUG_OFF);
```

Фрагмент заголовку при написання програми для мікроконтролера PIC16F883 з використанням мови C:

```
__CONFIG(FOSC_XT&WDTE_OFF&PWRTE_OFF&MCLRE_ON&CP_OFF&CP
D_OFF&BOREN_OFF&IESO_OFF&FCMEN_OFF&LVP_OFF&DEBUG_OFF);
__CONFIG(BOR4V_BOR40V&WRT_OFF);
```

3.3 Написання керуючої програми для мікроконтролера

Після включення живлення сервісна програма починає проводити ініціалізацію даних. На початку програми прикріплюється файл бібліотеки мікроконтролера, після чого оголошуються біти портів і регістри пам'яті, які використовуються в програмі. Даний тип мікроконтролера має вільні для використання комірки пам'яті і, щоб при кожному зверненні до певної комірки не описувати її адресу, та полегшити формування програми - присвоюється їй назва.

У поняття ініціалізації входить:

- присвоєння початкових значень всім змінним програми;
- налаштування всіх внутрішніх систем мікроконтролера;
- відновлення режимів роботи системи, встановлених у попередньому сеансі роботи шляхом зчитування відповідних параметрів із флеш-пам'яті;
- виконання підпрограм, призначених для установки всіх периферійних пристроїв схеми позиціонера у вихідний стан (очищення індикаторів, підтвердження команди зупинки двигуна тощо...);
- виконання підпрограми запуску системних лічильників-таймерів і механізму переривань мікроконтролера.

Крім констант, які, відповідно до своєї назви, протягом усього часу роботи програми ніколи не міняються, у програмі існують змінні. Під змінною в програмі розуміється ім'я регістра, або адреса комірки пам'яті, де в процесі виконання програми буде тимчасово зберігатися заздалегідь невідома величина, або проміжний результат обчислень.

Після виконання модуля ініціалізації програма переходить до основного циклу. У цьому циклі програма перебуває увесь час, поки включене живлення.

Першою частиною нашого проекту було написання тексту програми для першого мікроконтролера, що виконує функцію одержання даних з пристрою T74 та встановлення граничних значень, а також обробку та передачу даних на другий МК.

Наведений нижче модуль програми здійснює ініціалізацію портів мікроконтролера PIC16F876, модуля UART та ADC:

```
#include "МК_init.h"
//--- опис функцій -----
void МК_init(void)
{
  //--Налаштування портів--//
  TRISA=0b00000111;//перші 2 виводи - це кнопки, а 3 вивід - це моділь ADC
  TRISB=0;//не використовуванний порт
  TRISC7=0;//вивід передачі даних за протоколом UART
```

Наступний фрагмент коду здійснює ініціалізацію регістрів модуля UART:

```
//-----UART-----
TXREG=0;//регістр прийому
RCREG=0;//регістр прийому
//-----SPBRG-----
SPBRG=25;
//-----TXSTA-----
BRGH=1;
SYNC=0;//асинхронний режим
TX9=0;// робота 9-ти бітної посилки
TXEN=1;//дозвіл включення передавача
//-----RCSTA-----
RX9=0;//дозвіл роботи 9-ти бітної посилки
CREN=1;//дозвіл включення передавача
SPEN=1;//включення модуля UART
```

//Робота з аналогово-цифровим перетворювачем

```
//-----ADCON0-----
ADCS2=0; //швидкість перетворення ADC fosc/2
ADCS1=0;
ADCS0=0;
CHS2=0;
CHS1=1;//вибір каналу RA2
CHS0=0;
ADON=0; // АЦП не працює і не споживає струму;
```

```
//-----ADCON1-----
ADFM=1; //праве вирівнювання
```

// Біти налаштування каналів АЦП:

```
PCFG3=1;
PCFG2=0;
PCFG1=1;
PCFG0=0; //RA2 -> analog /digital
```



```

//-----INTCON -----
// Налаштування переривання від модуля АЦП:
ADIF=1;
ADIE=1;
GIE=1;
}

```

Після ініціалізації ми виконуємо написання керуючої програми у файлі main.c :

```

//----- ОСНОВНА ПРОГРАМА -----
void main(void)
{
    МК_init();//функція ініціалізації МК

//---- РОБОЧИЙ ЦИКЛ -----
    while(1)
    {
//----- ADC вимірювання -----
        ADON=1;
        __delay_us(50);
        GO_DONE=1;
        while(GO_DONE==1){}

        ADC_t=ADRESH;
        ADC_t=ADC_t<<8;
        ADC_t|=ADRESL;

        j=ADC_t*(5.0/1024); //--Обчислення вимірюваного значення напруги
        uart_trans(j);//та передаємо це значення на другий МК

//----- Робота з кнопка: -----
//визначення граничного значення до якого наше реле буде залишатись виключеним:
        if(Plys==0)//якщо нажата кнопка збільшення граничного значення, то:
        {
            while(Plys==0){} //очікуємо відтискання кнопки
            grn++; //збільшення граничного значення на 1
            uart_trans(grn);//передача значення на другий МК
        }

        if(Minus==0) // якщо нажата кнопка зменшення граничного значення, то:
        {
            while(Minus==0){} //очікуємо відтискання кнопки
            tmr--; //зменшення граничного значення на 1
            uart_trans(grn);//передача значення на другий МК
        }
    }
}

```

Другою частиною нашого проекту було написання тексту програми для другого мікроконтролера, що виконує функцію одержання даних з першого МК та виводить отримані дані на 3-рядковий CCI.

Наведений нижче модуль програми здійснює ініціалізацію портів мікроконтролера PIC16F883, модуля UART та TMR0:

```
#include "MK_init.h"
//--- опис функцій -----
void MK_init(void)
{
    //--Налаштування портів--//
    TRISA0=0;//порт виведення інформації для роботи реле
    TRISB=0; //порт на виведення інформації для засвічування сегментів
    TRISC=0b10000000; /*перші 3 виводи - керування засвічуванням розрядів CCI, а
    останній вивід - одержання даних від першого МК*/
```

Наступний фрагмент коду здійснює ініціалізацію регістрів модуля UART:

```
//-----UART-----
    TXREG=0;//регістр прийому
    RCREG=0;//регістр прийому
    //-----SPBRG-----
    SPBRG=25;
    //-----TXSTA-----
    BRGH=1;
    SYNC=0;//асинхронний режим
    TX9=0;// робота 9-ти бітної послідовності
    TXEN=1;//дозвіл включення передавача
    //-----RCSTA-----
    RX9=0;//дозвіл роботи 9-ти бітної послідовності
    CREN=1;//дозвіл включення передавача
    SPEN=1;//включення модуля UART

//-----Модуль переривання TMR0 -----
    TMR0=0;//скидання TMR0
    INTCON=0; //виклик переривань і скидання T0IF
    T0CS=0; //тактування внутрішнього джерела
    PSA=0; //з попереднім дільником
    //встановлення коефіцієнта розподілу 1:64
    PS2=1;
    PS1=0;
    PS0=1;
    //налаштування INTCON
    INTCON=0;
    T0IF=0;
    T0IE=1;
    GIE=1;
}
```

Після ініціалізації ми здійснюємо написання керуючої програми у файлі main.c :

```
//----- ОСНОВНА ПРОГРАМА -----
```

```
void main(void) {
    МК_init(); //модуль ініціалізації МК
//-----//
// Цей фрагмент написаний для того, щоб не переводити символи для ССІ зі С А
for(char i=0;i<13;i++)
{
    tmp=MAS1[i];
    tmp=~tmp;
    MAS[i]=tmp;
}
```

```
//---- РОБОЧИЙ ЦИКЛ -----
```

```
while(1) {
//виконуємо зчитування даних з першого МК, а саме напруги з модуля ADC:
    j =getch();
//та граничного значення, утвореного кнопками:
    grn= getch();
```

Наступний фрагмент здійснює підготовку даних для виведення на РКІ:

Спершу розділимо число з ADC на окремі цифри:

```
j1=(j /10)%10;
j2= j %10;
```

Після чого перевіряємо значення напруги чи менше воно граничного, для роботи реле:

```
if(j <= grn)//якщо значення напруги менше граничного, то:
{
    RA0=0; //реле залишається виключеним
}
```

//Якщо ж зчитане значення напруги більше встановленого граничного числа, то в нас відбувається включення реле:

```
else {
    RA0=1;
}
}
```

Це функція, яка дозволяє нам організувати безперервну динамічну індикацію:

```
void interrupt DYN(void)
{
    if(T0IE&&T0IF)
    {
        switch(cnt)
        {
            case 0: R3=1;PORTB=MAS[0];R0=0;R1=R2=1;break;
            case 1: R0=1;PORTB=MAS[j2];R1=0;R2=R3=1;break;
            case 2: R1=1;PORTB=MAS[j1];R2=0;R0=R3=1;break;
            case 3: R2=1;PORTB=MAS[0];R3=0;R1=R0=1;break;
        }
        cnt++;
    }
```

```

if(cnt==4){cnt=0;}
TMR0=0;
T0IF=0;//очищення прапорця T0IF
}
}

```

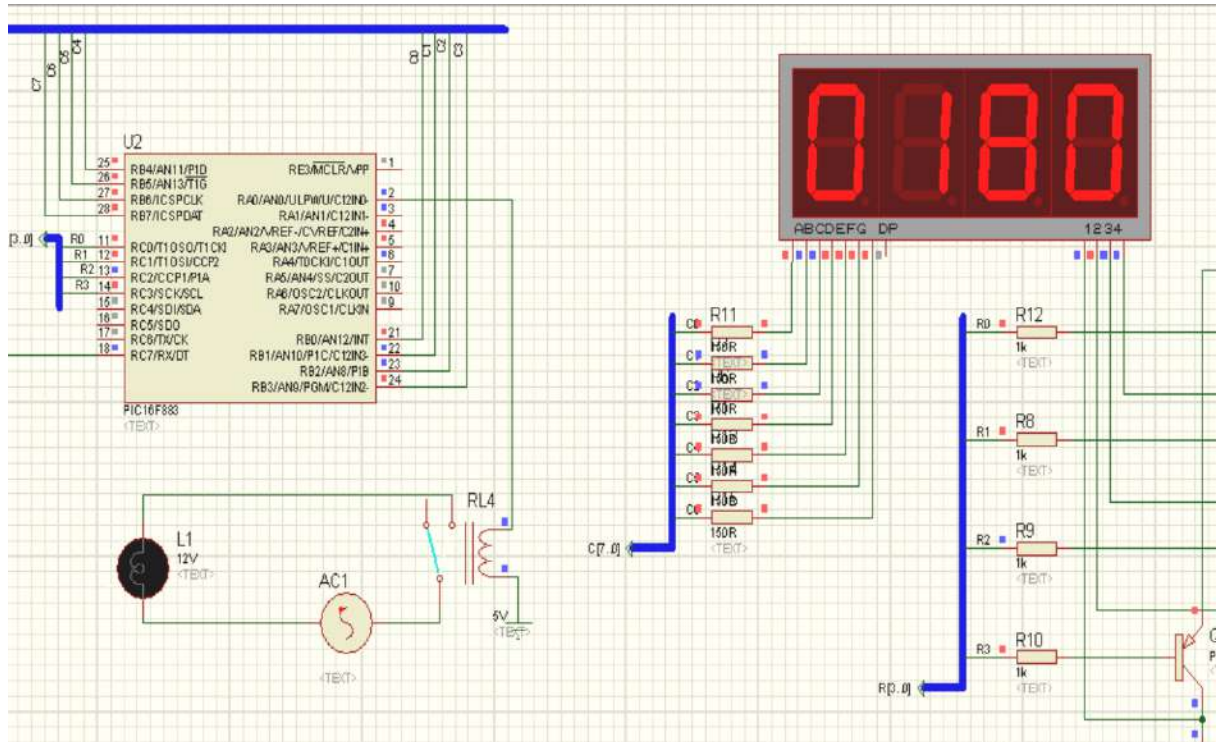


Рисунок 3.3 - Вигляд пристроїв виведення в режимі дотримання граничного значення

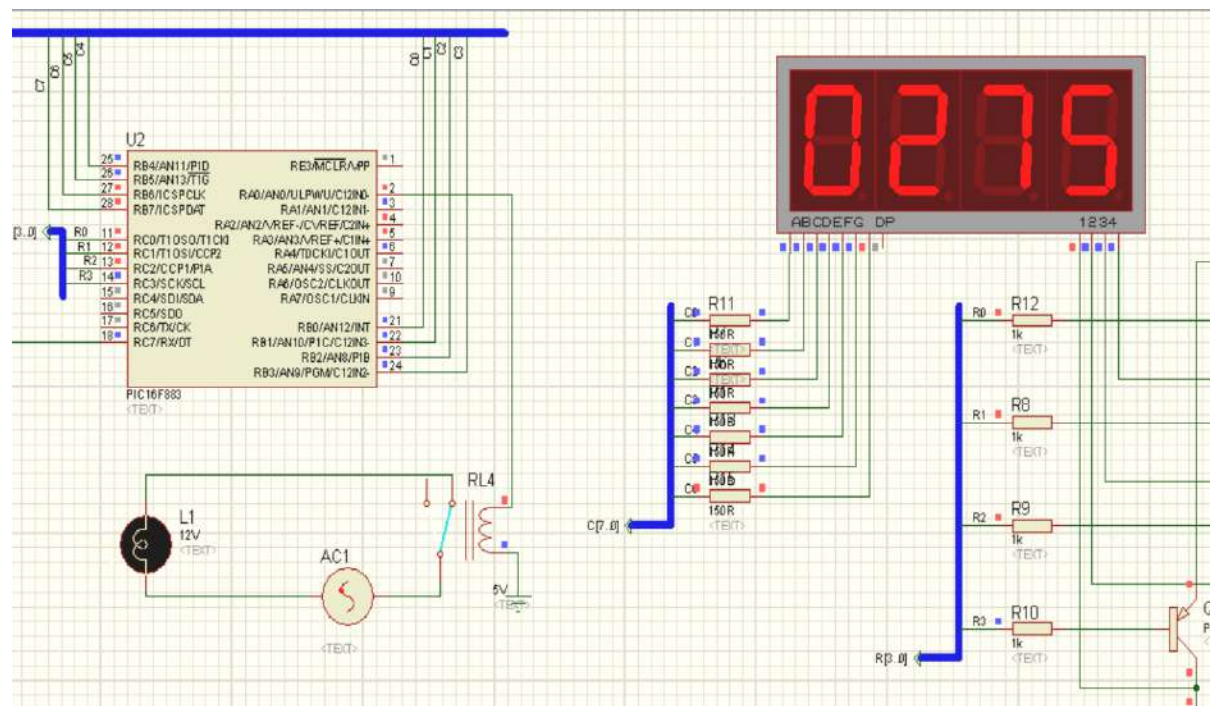


Рисунок 3.4 - Вигляд пристроїв виведення в режимі порушення граничного значення

ВИСНОВКИ

В процесі курсового проектування розглянуто принципи проектування пристроїв і систем, що побудовані з використанням однокристальних мікроконтролерів фірми Microchip. Під час виконання проектування здійснено закріплення теоретичних знань та практичних навичок по реалізації етапів проектування. Узагальнено технічний матеріал для формування основної ідеї проектування. Здійснено аналіз вхідних даних, обґрунтування і синтез структурної схеми. Детально розглянуто складові апаратної частини їх призначення і особливості функціонування. Проаналізовано документацію, що описує роботу складових вузлів МКС. На основі цих даних описано та спроектовано принципову схему та алгоритм роботи.

Для побудови блок-схем, алгоритмів та принципових схем закріплено навички роботи з програмою sPlan. Для написання і владодження програмного забезпечення використано середовище розробки MPLAB 8.92. У цьому середовищі створено проект, написано текст програми з використанням мови C, і здійснено компіляцію у hex-файл. Середовище розробки може використовуватись разом з програматорами для запису готового вихідного коду прошивки у пам'ять програми вибраного мікроконтролера.

При написанні тексту програми сформовано значення регістра конфігурації за даними документації виробника, описано регістри спеціальних функцій та загального призначення.

Проведено ініціалізацію мікроконтролера, його вузлів та модулів.

Для забезпечення роботи периферійних пристроїв здійснено керування, прийом та передача даних з пристроїв введення та до пристроїв виведення згідно варіанту.

У робочій частині забезпечено функціонування МКС згідно варіанту.

Курсовий проект може бути основою розробки керуючої системи збору даних чи системи керування та контролю за виконанням завдань.

ПЕРЕЛІК ПОСИЛАНЬ

1. Войтович І. В., Сидоренко В. М. Мікропроцесорні системи та мікроконтролери. — Київ: КНТ, 2020.
2. Мазур О. М. Мікроконтролери AVR: програмування мовою C. — Львів: Львівська політехніка, 2018.
3. Погорілий Ю. С. Вбудовані системи на базі мікроконтролерів. — Київ: КПІ ім. І. Сікорського, 2021.
4. Barrett S. F., Pack D. J. Atmel AVR Microcontroller Primer: Programming and Interfacing. — Morgan & Claypool Publishers, 2019.
5. Han-Way Huang. The Atmel AVR Microcontroller: MEGA and XMEGA in Assembly and C. — Cengage Learning, 2017.
6. Jack Ganssle. The Art of Designing Embedded Systems. — Newnes, 2020.