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: «**Проєктування систем керування та збору інформації на базі мікроконтролерів сімейства PIC16»**

Виконав студент III курсу, групи ***КІ-340***
Спеціальності *123 Комп'ютерна інженерія*
Варіант № 04
Дацюк Віталій Ігорович
(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

Дацюку Віталію Ігоровичу
(прізвище, ім'я та по батькові)

з навчальної дисципліни: МІКРОКОНТРОЛЕРИ ТА ЇХ ПРОГРАМУВАННЯ

Студент групи: **КІ-340**

1. Тема проекту: Система моніторингу температури.

2. Дата видачі завдання: ”26” вересня 2025 р

3. Термін здачі курсового проєкту: ”19” грудня 2025 р.

4. Вихідні дані до проєкту:

4.1 Пристрої введення інформації : кнопки, DS18B20

4.2 Функції пристрою введення інформації: налаштування границь, давач температури

4.4 Використовувані мікроконтролери: PIC16F877, PIC16F689.

4.5 Протокол обміну інформацією: UART.

4.6 Пристрій виведення інформації: ССІ-6 р., (СК), реле

4.7 Функції пристрою виведення інформації: індикація стану реле on/off, t- ххС

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 Обгрунтування 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		

Студент _____ **Віталій ДАЦЮК**
(підпис) (імя та прізвище)

Керівник проекту _____ **Роман СТОЛЯРЧУК**
(підпис) (імя та прізвище)

РЕФЕРАТ

Текстова частина курсового проєкту: 39 сторінок, 18 рисунків, 1 таблиці та 8 посилань.

Об’єкт проектування - “Система моніторингу температури”.

Мета проекту - довершити свої навики у розробці структурної схеми, електричної принципової схеми МКС, створення проекту 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, DS18B20, UART, CCI-6 р., (СК), 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

Типовими представниками RISC-процесорів є PIC-контролери (Periferial 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 кілослів 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 двонапрямлений 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 проектуються, 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 формулювати і вирішувати задачі допустимої складності, що піддаються усвідомленню й розумінню людиною та рішенню 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 температури, що працює на онові 2 мікроконтролерах PIC16F877, PIC16F689.

Робота приладу забезпечується давачем температури DS18B20 та кнопками керування, що встановлюють граничні значення.

Вивід інформації здійснюватиметься через 6-розрядний CCI (СК), що показуватиме індикацію, та реле.

Усі прилади даного проекту з’єднуються за протоколом UART.

Структурна схема МКС для Системи моніторингу 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 Система контролю температури зображено графічний вивід інформації, що буде мати наступний вигляд: реле on/off, t- xxC.

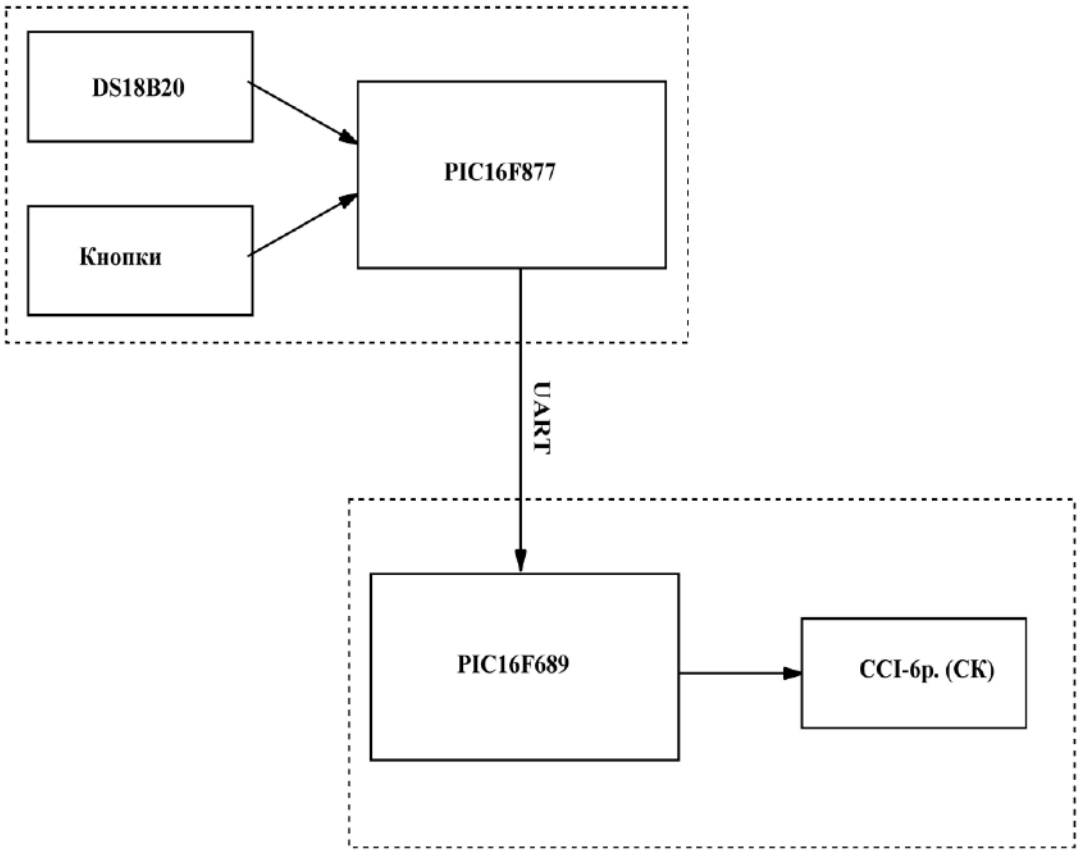

Рисунок 1.1 - Структурна схема системи контролю температури

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 - звузити 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 мікро-контролер в проектованій 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 PIC16F877 та PIC16F689 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 Давач температури DS18B20

Цифровий термодавач фірми DALLAS-MAXIMDS18B20 - це цифровий вимірювач температури, з перетворенням 9 - 12 розрядів і функцією тривожного сигналу контролю за температурою. Параметри контролю можуть бути задані користувачем і збережені в енергонезалежній пам'яті давача. DS18B20 обмінюється даними з мікроконтролером однопровідною лінією зв'язку, використовуючи протокол інтерфейсу 1-Wire. Діапазон вимірювання температури становить від -55 до +125 ° С.Для діапазону від -10 до +85 ° С похибка не перевищує 0,5 ° С. У кожної мікросхеми DS18B20 є унікальний серійний код довжиною 64 розряди, який дозволяє декільком давачам підключатися на одну загальну лінію зв'язку. Тобто через один порт мікроконтролера можна обмінюватися даними з декількома давачами, розділеними на значній відстані. Зовнішній вигляд і призначення виводів наведені у описі виробника.

На рисунку 1.2 наведена блок-схема датчика DS18B20. 64-бітний ПЗП (ROM) зберігає унікальний серійний код пристрою.

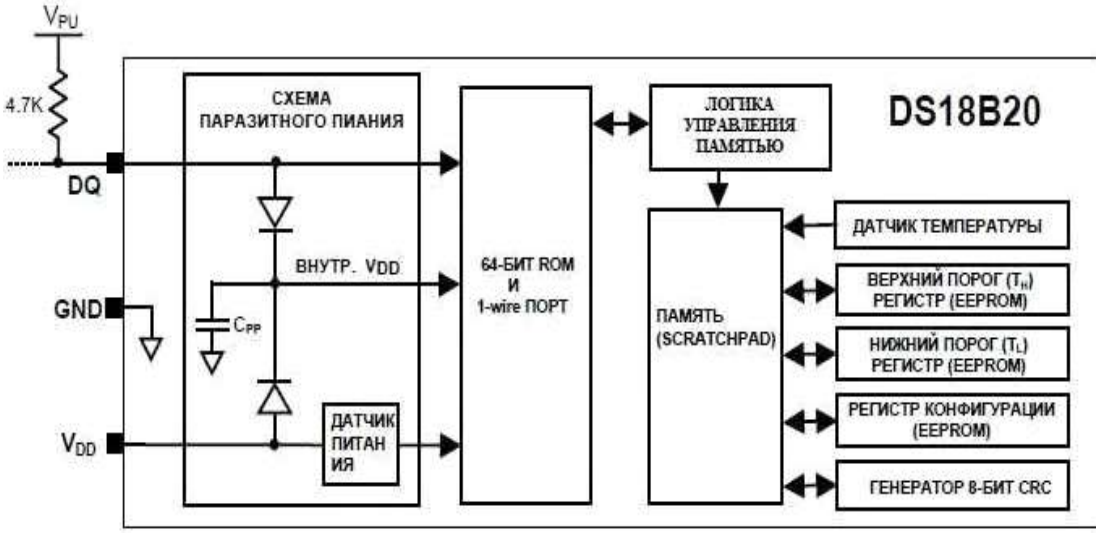

Рисунок 1.2- Блок-схема DS18B20

Основна функція DS18B20 - перетворення температури давача в цифровий код. При включенні живлення, стан регістра конфігурації встановлюється на 12 біт. Щоб ініціювати вимірювання температури мікроконтролер повинен подати команду перетворення температури [44h]. Після завершення перетворення, результат вимірювання температури буде знаходитися в 2 байтах регістра температури, і давач знову перейде в стан спокою. DS18B20 вимірює температуру в градусах за шкалою Цельсія. Результат вимірювання представляється як 16-розрядне, знакове число в

додатковому коді. Біт знака (S) дорівнює «0» для додатніх чисел і «1» для від'ємних.

Оперативна пам'ять мітить:

- значення виміряної температури (2 байта);
- верхній і нижній порогові значення спрацювання сигналу тривоги (Th, Tl);
- регістр конфігурації (1 байт);
- генератор 8-бітної CRC.

Організація пам'яті DS18B20 показана на рисунку 1.3. Вона включає в себе оперативну (SRAM) і енергонезалежну (EEPROM) пам'ять. У EEPROM зберігаються регістри Th, Tl і регістр конфігурації. Якщо функція тривожного сигналу не використовується, то регістри Th і Tl можуть використовуватися як регістри загального призначення.

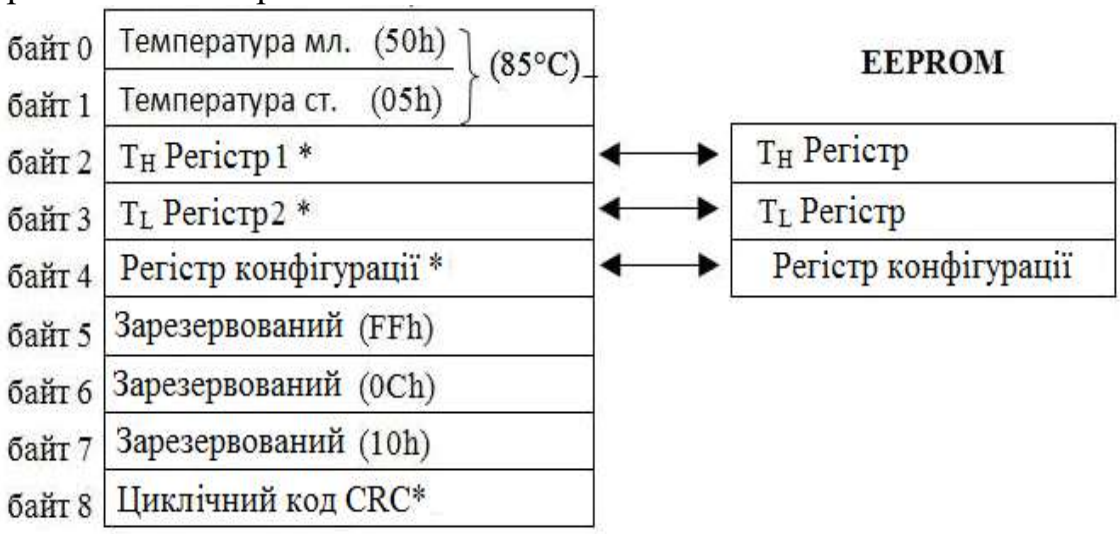

Рисунок 1.3 - Карта пам'яті DS18B20

Байт 4 пам'яті це регістр конфігурації. Бітами R0, R1 можна встановити розширення для перетворення температури. При включенні живлення стан бітів R0, R1 = 11, що відповідає розширеню в 12 біт. Треба пам'ятати, що існує пряма залежність часу перетворення від роздільної здатності. Біти 7 і 0 – 4 зарезервовані, вони не можуть використовуватися і при читанні повертають 1.

Таблиця 1.1 - Коди конфігурування розширення перетворення температури

R1	R0	Розширення	Час перетворення
0	0	9-біт	93.75мс (t_{conv}/8)
0	1	10-біт	187. 5мс (t_{conv}/4)
1	0	11-біт	375мс (t_{conv}/2)
1	1	12-біт	750мс (t_{conv})

У таблиці 1.1 показані приклади відповідності цифрових кодів значенням температури.

Для обчислення температури необхідно:

- при додатньому значенні (S = 0) код перевести в десятковий і помножити на 0,0625 ° C;
- при від'ємному значенні (S = 1) - перевести обернений код в прямий. Для цього треба інвертувати кожен розряд двійкового коду і додати 1. А потім перевести в десятковий і помножити на 0,0625 ° C.

Пам'ять включає в себе оперативну (SRAM) і енергонезалежну (EEPROM) пам'ять. У EEPROM зберігаються регістри TH, TL і регістр конфігурації. Якщо функція тривожного сигналу не використовується, то регістри TH і TL можуть використовуватися як регістри загального призначення.

В байтах з адресами 0 і 1 зберігаються молодший і старший байти регістра виміряної температури. Ці байти доступні тільки для читання. 2й і 3й байти - TH і TL регістри. 4 Байт - регістр конфігурації. Байти 5, 6, 7 зарезервовані, не можуть бути записані і, при читанні, завжди повертають 1. 8 Байт доступні тільки для читання. Він містить циклічний код (CRC) для перших восьми байтів.

Запис даних в байтах 2, 3 і 4 відбувається командою запису пам’яті [4Eh]. Дані повинні передаватися, починаючи з молодшого біта байта 2. Для перевірки запису даних можна прочитати пам'ять командою читання пам’яті [код BEh]. При читанні дані передаються по шині, в послідовності починаючи з молодшого біта байта 0. Запис Даних TH, TL і регістра конфігурації в EEPROM відбувається командою [48h]. При включенні живлення дані з незалежної пам'яті EEPROM перевантажуються в оперативній пам'яті (SRAM). Ведучий пристрій повинен контролювати стан шини, щоб визначити завершення перезавантаження. Слот читання низького рівня означає, що перезавантаження ще не закінчилось. По завершенню перезавантаження DS18B20 передає слот читання.

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 (ССІ). 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 буквений 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 ССІ 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.

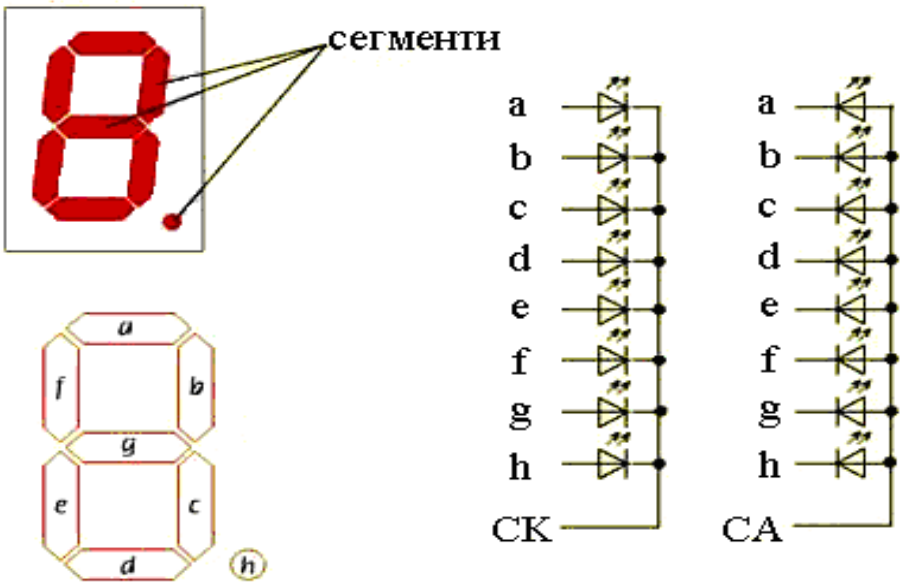

Рисунок 1.4 – Умовне графічне зображення ССІ

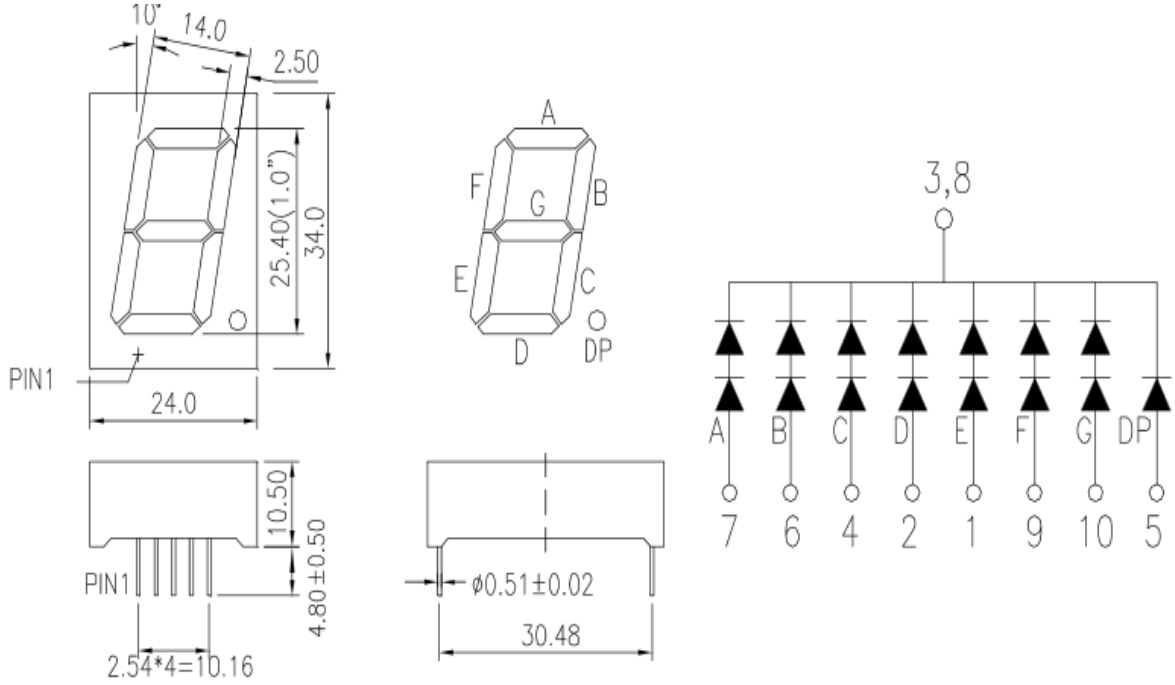

Рисунок 1.5 - Геометричні розміри та електрична схема однорозрядного ССІ зі спільним анодом.

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...140 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.

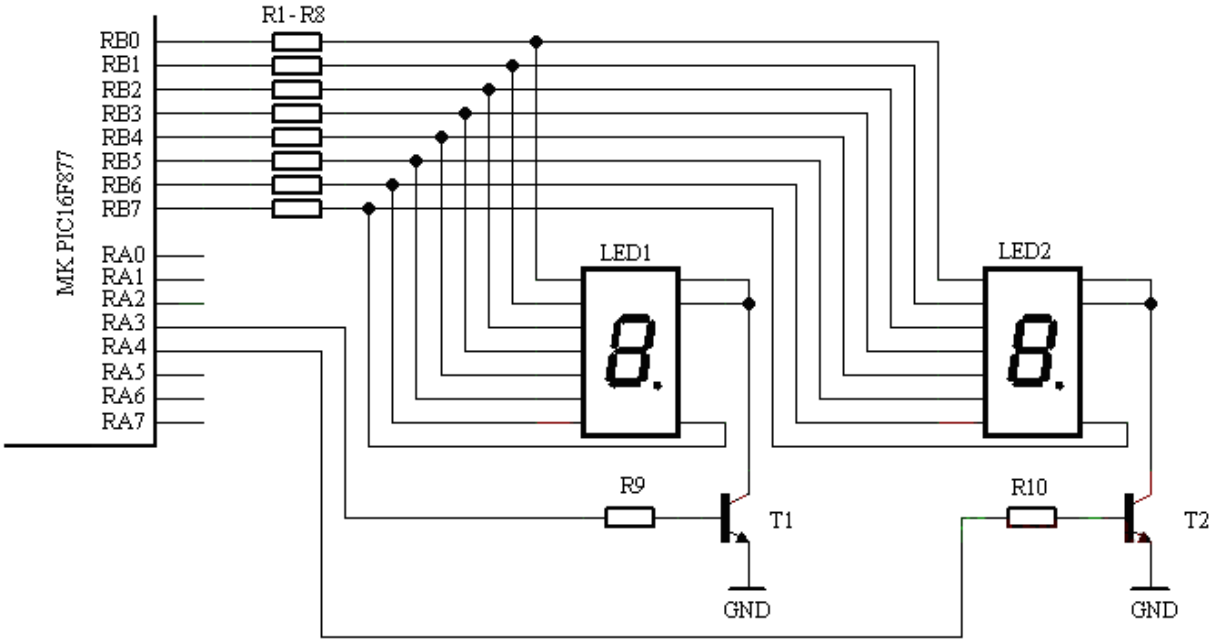

Рисунок 1.6 – Умовна схема під‘єднання ССІ при динамічній індикації

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.

Для спрощення написання коду можна використати 16-кову систему.

	h	g	f	e	d	c	b	a		
для цифри 1:	0	0	0	0	0	1	1	0	=>	0x06
для цифри 2:	0	1	0	1	1	0	1	1	=>	0x5B
для цифри 3:	0	1	0	0	1	1	1	1	=>	0x4F
і так далі...										

Використовуючи десяткові цифри від 0 до 9, як індекс масиву, легко виводити в порт потрібні коди. Цього можна досягнути, використовуючи у програмі так звану таблицю символів

У сегмент таблиці в програмі включені символи цифр. Якщо потрібні інші символи, можна розширити цю таблицю.

Принцип читання з таблиці полягає в наступному. При виклику таблиці у підпрограмі відбувається додавання молодшого байта лічильника команд PCL і

значення, яке містить робочий регістр W з використанням інструкції «*addwf PCL, 1*». У результаті додавання лічильник збільшується на величину, яка утримується в W і відбувається перехід на відповідну команду в таблиці - в робочий регістр W записується значення номера «комірки» таблиці – у такий спосіб у пам'яті програм відбувається перехід до інструкції з адресою PCL+W.

Діапазон адрес, який може бути отриманий в результаті, містить «комірки» таблиці у вигляді інструкцій «retlw» (наприклад «retlw b'00000110'»), які викликають повернення з підпрограми із занесенням свого аргументу в робочий регістр W. Зазначене число являє собою комбінацію бітів, яке потім записується в порт і приводить до відображення осмисленого символа. Таким чином, у результаті повернення з таблиці, W містить необхідне значення. Необхідно контролювати вміст регістра W при виклику таблиць, щоб не відбувся перехід за межі підпрограми.

Робота з масивами у С дуже проста і зручна. Масив - це структура даних, що представлена у вигляді групи значень одного типу, які об'єднані під єдиним ім'ям. Елементами масиву можуть бути дані будь-якого типу.

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 1.9). Напруги вище 3В приймають за сигнали логічного нуля , а будь-які напруги нижче -3В – логічною одиницею. Напруга в діапазоні від -3В до +3 вважається сигналами без змісту.

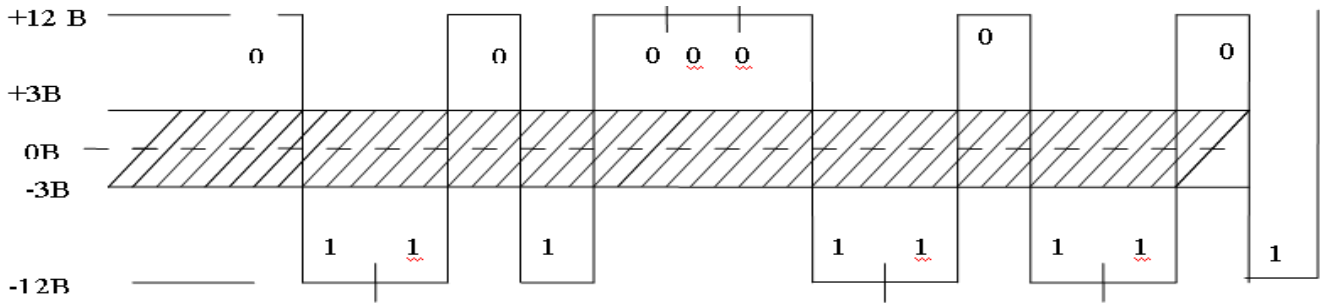

Рисунок 1.7 - Реалізація інтерфейсу RS-232 на фізичному рівні

По причині відсутності синхронізації, яка показує початок і кінець кожного байту, байт даних приходиться починати стоповим і закінчувати стартовим бітами (рисунок 1.10), 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 1.10.

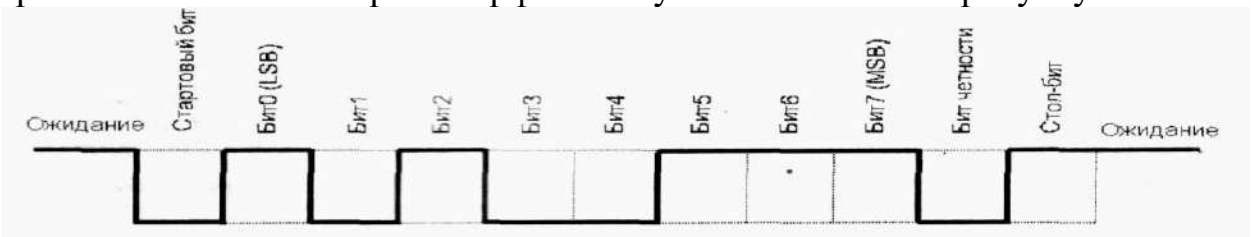

Рисунок 1.8 - Часова діаграма передачі байта E5h (11100101b).

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 Особливості протоколу 1-WIRE

1-Wire - протокол передачі даних по одній лінії. Даний протокол розроблений корпорацією Dallas Semiconductor (зараз Maxim Integrated). З використанням 1-Wire працює більшість домофонних чіпів (DS1990A), карток доступу, а також через 1-Wire спілкуються популярні давачі температури (DS18S20 і DS18B20), транзисторні ключі (DS2405, DS2406), програмовані порти введення-виведення (DS2408), АЦП і ЦАП, годинники реального часу (DS2417) тощо…. Режим зв'язку в цьому протоколі - асинхронний і напівдуплексний. При цьому завжди є ведучий - один пристрій на шині, яке посилає команди, і ведені - пристрої, які ці команди приймають і відповідають на них, якщо необхідно; кожен з ведених пристроїв підключається безпосередньо до загальної шини. Протокол 1-Wire не складний в реалізації однак досить чутливий до часу і до перешкод. Також 1-Wire не призначений для передачі великих об’ємів інформації та для швидкісного обміну даними - максимальна швидкість 9600бод/с.

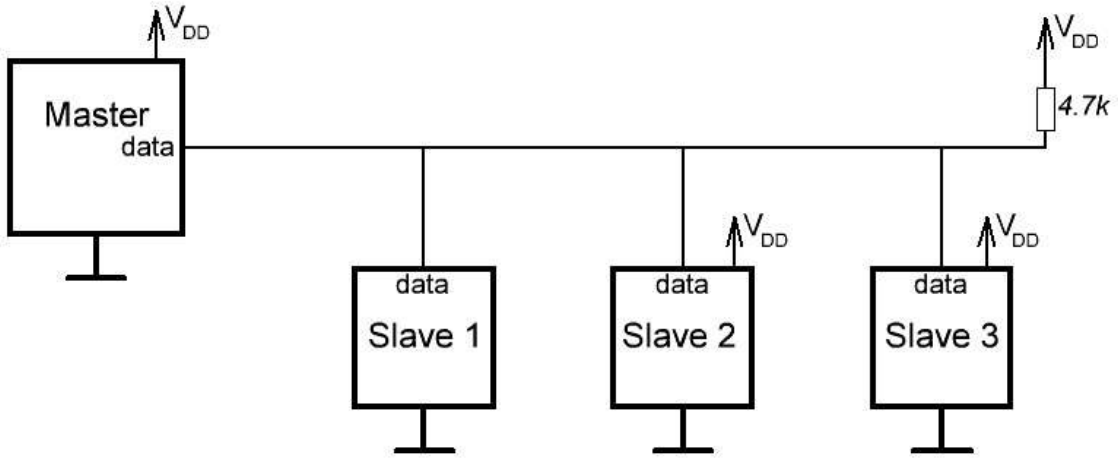

Рисунок 1.9 -Приклад підключення ведучого і ведених приладів за протоколом 1-Wire

Протокол 1-Wire описує фізичний, канальний, мережевий і транспортний рівні взаємодії. На фізичному рівні даються описи методів зв'язку, вимоги до шини даних і живленню. Канальний рівень описує способи читання і передачі бітів по протоколу. Мережевий рівень визначає методи адресації до різних пристроїв на лінії. Нарешті, транспортний рівень описує функціональні команди, використовувані пристроями 1-Wire.

Обмін інформацією ведеться тайм-слотами (60 мкс): один тайм-слот служить для обміну одним бітом інформації. Дані передаються біт за бітом, починаючи з молодшого біта молодшого байта.

При обміні інформацією ведучий ініціює зв'язок на бітовому рівні. Це означає, що передача кожного біта, незалежно від напрямку (передача або прийом), повинна бути ініційована ведучим.

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 електричні 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).

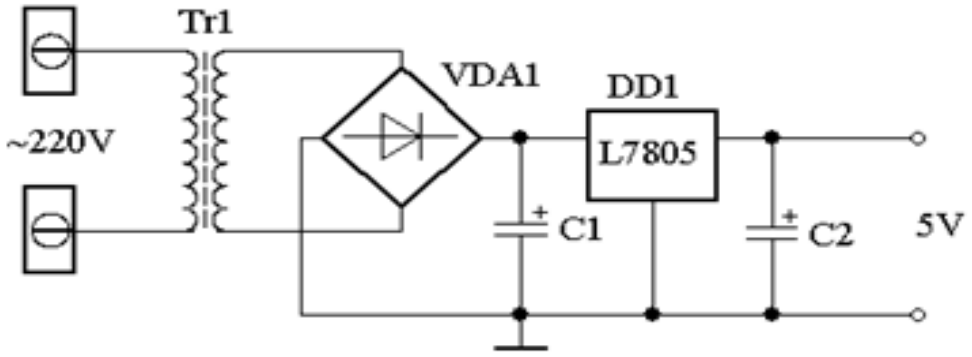

Рисунок 2.1- Схема стабілізації живлення для цифрових пристроїв на основі мікроконтролерів

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 78М05 з струмом навантаження до 0,5А.

Ємність С1 на вході стабілізатора необхідна для згладження високочастотних завад при подачі вхідної напруги. Ємність С2 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.

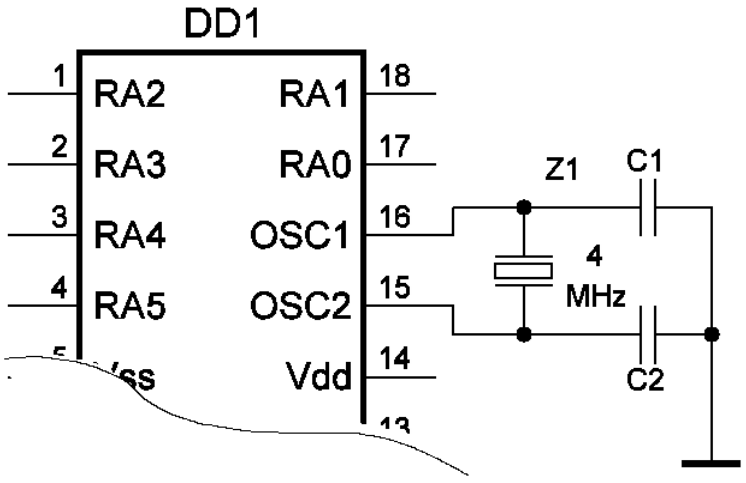

Рисунок 2.2 - Схема тактування мікроконтролера

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 ISCP 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 проектованого 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 виконувані ним 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‘єднюються 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.

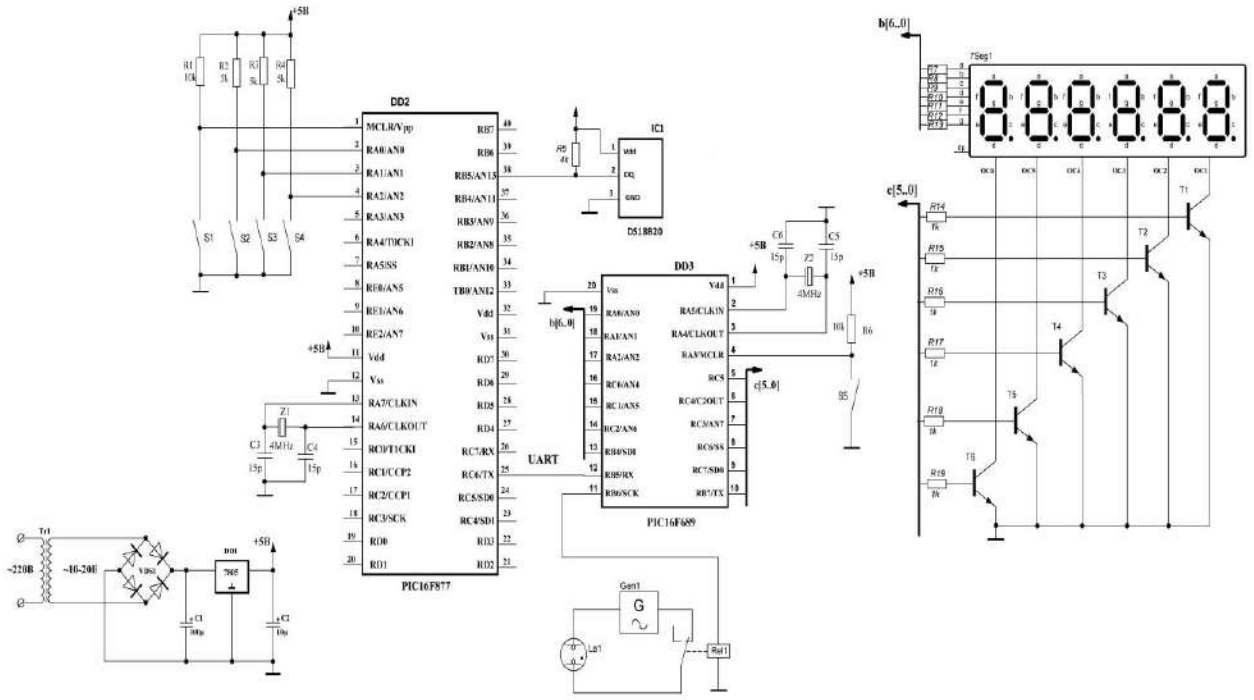

Рисунок 2.3 - Електрична принципова схема роботи Системи контролю температури

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	15пФ
8	Резистори	R1,R6	10кОм
9	Резистори	R2, R3, R4	5кОм
10	Резистор	R5	4кОм
11	Резистори	R14-R19	1кОм
12	Резистори	R7-R13	100Ом
13	Мікроконтролери	DD2,DD3	PIC16F877, PIC16F689
14	Реле	Rel1	12В
15	Генератор сигналів	Gen1	1Гц
16	Лампа	La1	12В
17	Цифровий термодавач	IC1	DS18B20
18	8-розрядний ССІ	7Seg1	MPX8-CK

19	Транзистори	T1-T6	NPN
20	Кнопка-перемикач	S1-S5	SK1-53

2.2 Створення проекту в середовищі PROTEUS VSM

Більшість радіоаматорів стикалися із ситуацією, коли за браком досвіду чи присутність 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 трасирувальник 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.

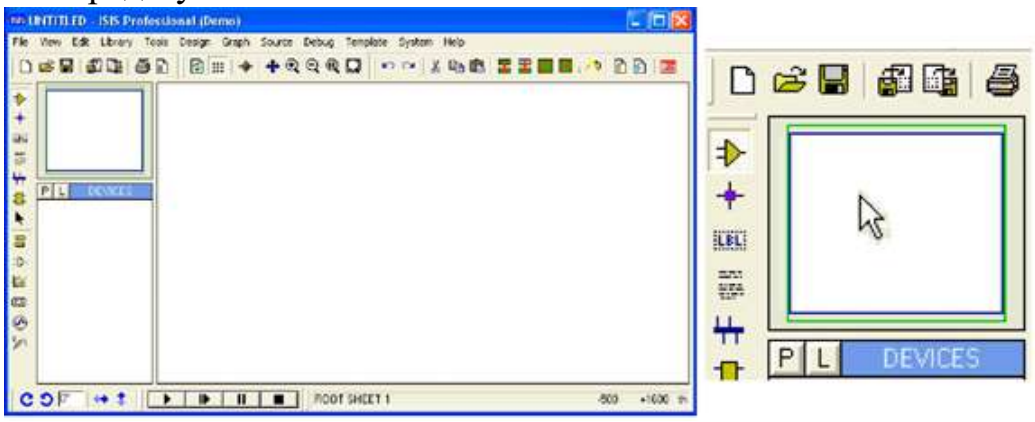

Рисунок 2.4 - Основне вікно програми ISIS 7 Professional і вікно попереднього перегляду.

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'явитися 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.

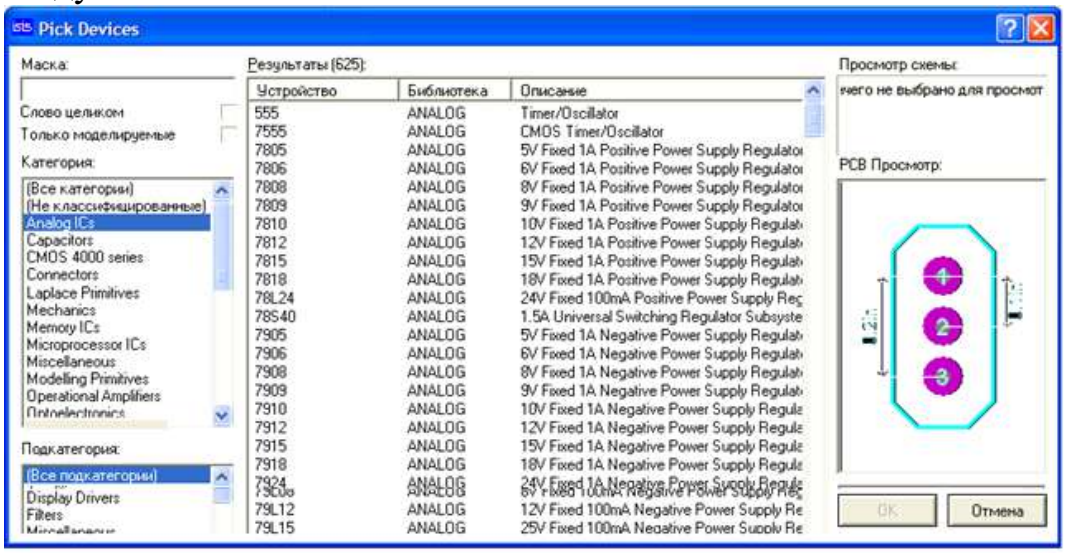

Рисунок 2.6 - Бібліотека електронних компонентів.

Для того, щоб переглянути роботу схеми, необхідний вихідний HEX-файл. Середовище PROTEUS підтримує багато засобів розробки, серед них і HI-TECH Сі 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.

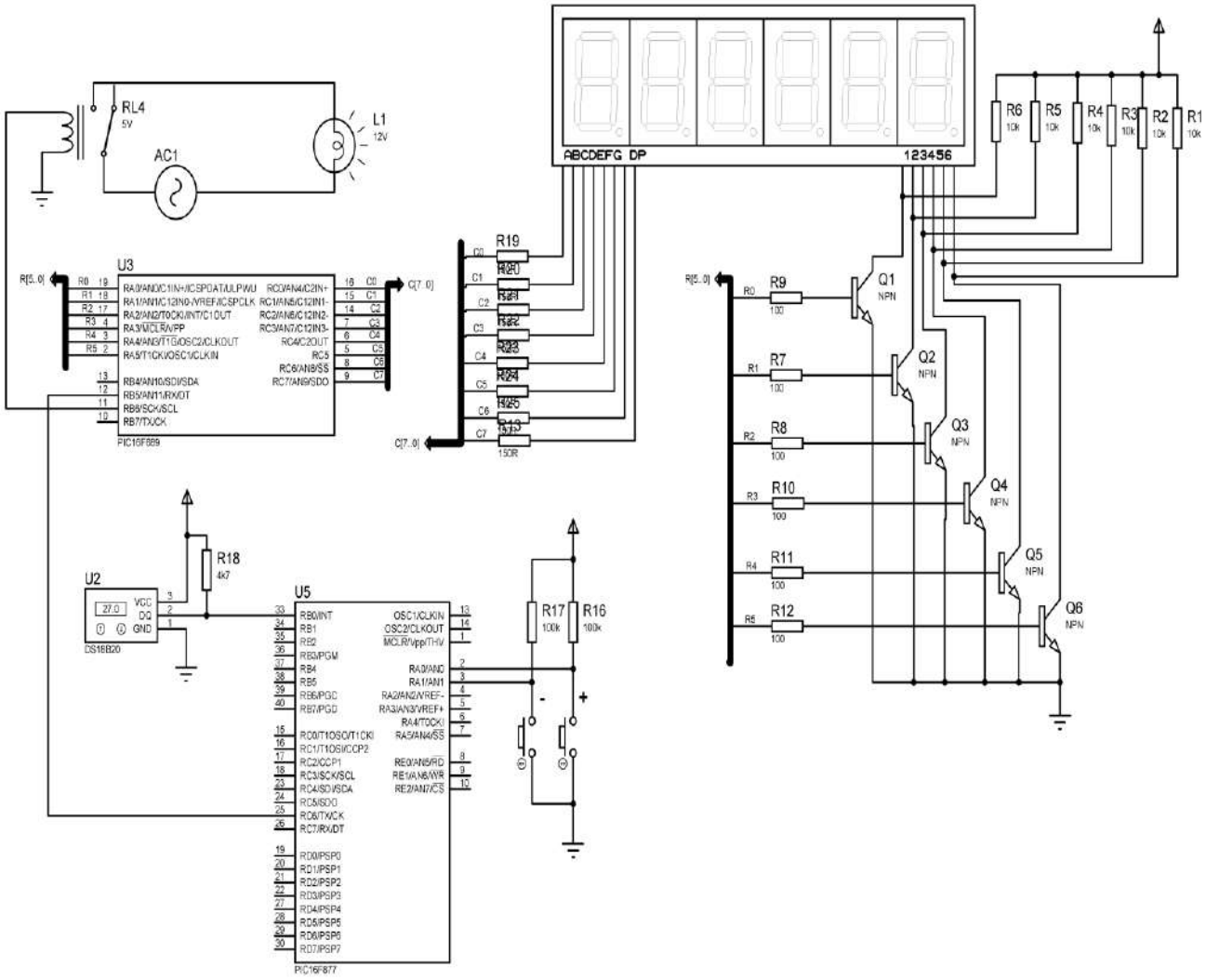

Рисунок 2.7 - Досліджувана схема

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.

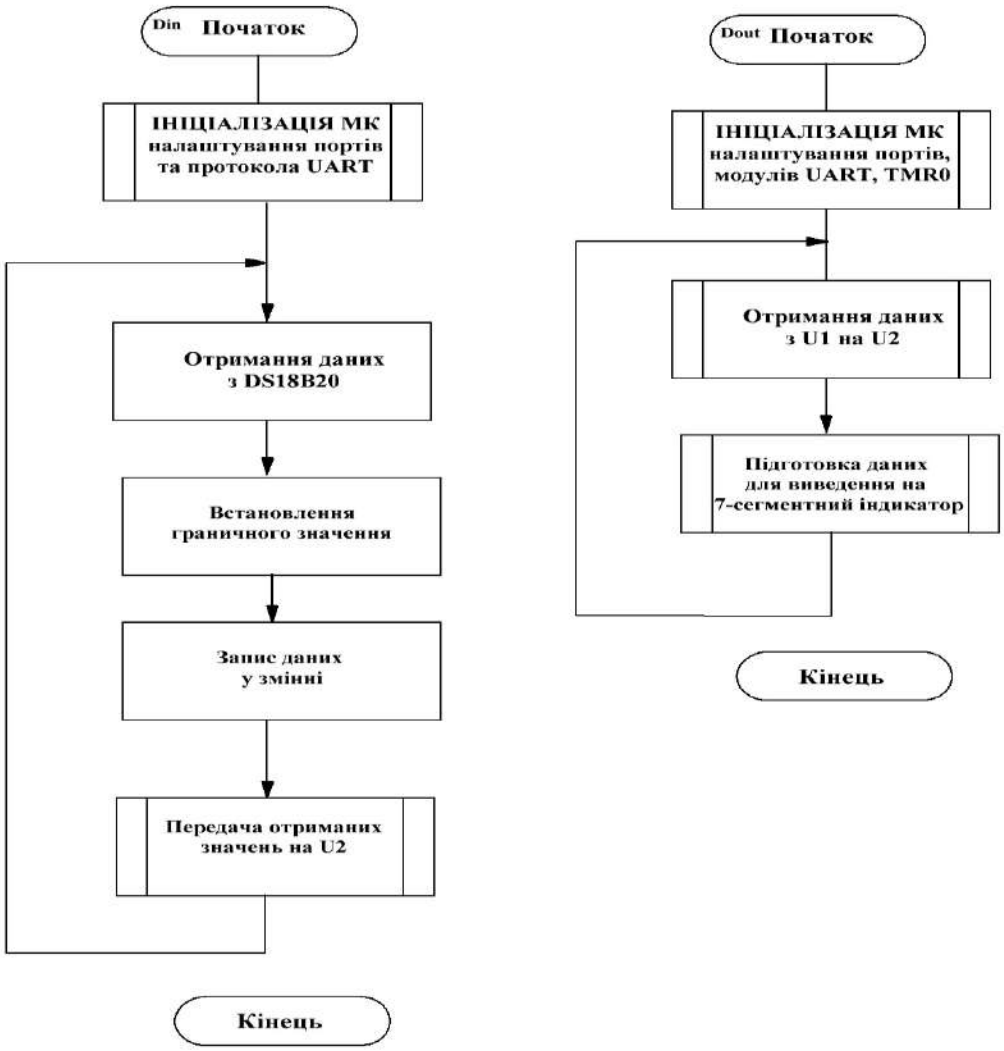

Рис. 3.1 - Блок-схеми роботи Система контролю температури

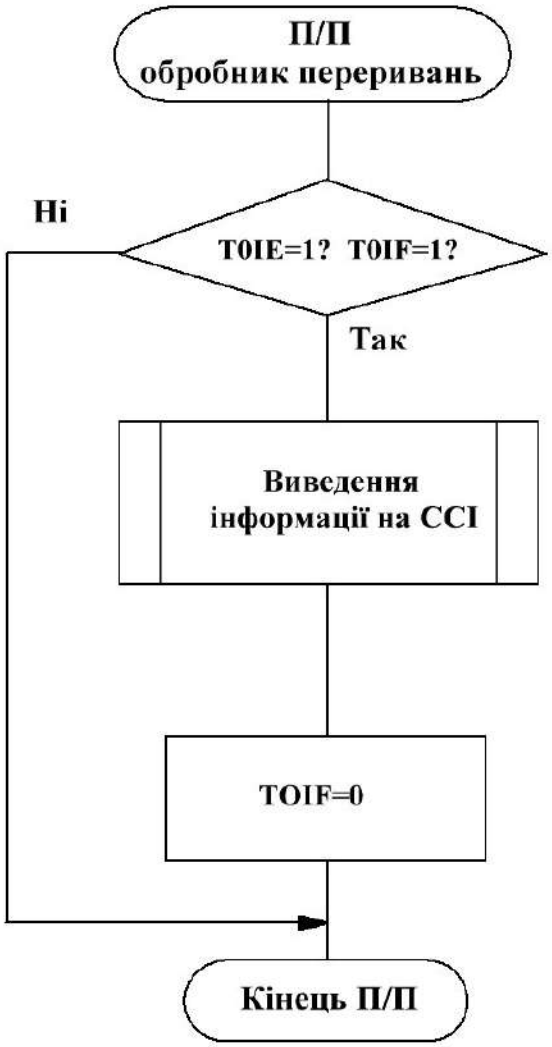

Рис. 3.2 - Обробник переривання за TMR0

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 PIC16F877 з використанням мови С:

__CONFIG(FOSC_XT&WDTE_OFF&PWRTE_OFF&CP_OFF&BOREN_OFF&LVP_OFF& CPD_OFF & WRT_OFF &DEBUG_OFF);

Фрагмент заголовку при написання програми для мікроконтролера PIC16F689 з використанням мови С:

```
#include <pic.h>
__CONFIG(FOSC_HS&WDTE_OFF&PWRTE_OFF&CP_OFF&BOREN_OFF&LVP_OFF&CPD_OFF&WRT_OFF&DEBUG_OFF &MCLRE_ 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 DS18B20 та встановленя граничного значення за допомогою керуючих кнопок, а також обробку та передачу даних на другий МК.

Наведений нижче модуль програми здійснює ініціалізацію портів мікроконтролера PIC16F877 та модуля UART:

```
#include "MK_init.h"
//---- опис функцій ----------------------
void MK_init(void)
{
//--Налаштування портів--//
        TRISA=1;//порт читання команд з кнопок
        TRISB0=1; //вивід по якому ми отримуємо дані з DS18B20
        TRISC6=0; //вивід по якому ми передаємо дані на другий 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;//асинхроний режим
TX9=0;// робота 9-ти бітної посилки
TXEN=1;//дозвіл включення передавача
//------RCSTA---------------
RX9=0;//дозвіл роботи 9-ти бітрної посилки
CREN=1;//дозвіл включення передавача
SPEN=1;//включення модуля UART
}
```

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

```
//---- ОСНОВНА ПРОГРАМА ------------------
void main(void)
{
MK_init();
```

//Наступний фрагмент здійснює читання даних температури з пристрою DS18B20:

```
ds18b20_init();//підключаємо функцію ініціалізації DS18B20
ds18b20_trans(Skip_ROM);//звернення до єдиного пристрою на шині
ds18b20_trans(Adc_strt);// запускаємо конвертацію температури
 __delay_ms(750); //затримка для коректної роботи
ds18b20_init();
ds18b20_trans(Skip_ROM);
ds18b20_trans(Read_RAM); //зчитуємо адресу пристрою, щоб визначити адреси //єдиного пристрою на шині
DS_t=ds18b20_receive();//запис даних у змінну
        while(1) {
```

Далі здійснюється обробка даних для передачі на другий мікроконтролер:

```
ds18b20_init();
ds18b20_trans(0xCC);// пропускаємо ідентифікацію
ds18b20_trans(0x44);// запускаємо конвертацію температури
__delay_ms(750);
   ds18b20_init();
ds18b20_trans(0xCC); //знову пропускаємо ідентифікацію
ds18b20_trans(0xBE);//та здійснюємо читання регістра
```

Після чого ми записуємо конвертовану температуру у змінну T та передаємо на другий МК:

```
T=receive();
T=T>>4;
Tl=T<<12;
uart_trans(T);//передача даних

//---------- 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 6-розрядний ССІ (СК).

Наведений нижче модуль програми здійснює ініціалізацію портів мікроконтролера PIC16F689 та модуля UART:

```
#include "MK_init.h"
//---- опис функцій ----------------------
void MK_init(void)
{
//--Налаштування портів--//
TRISC=0;//порт виведення інформації для засвічування сегментів ССІ
TRISA=0; /порт виведення інформації для активації розрядів ССІ
TRISB5=1; //вивід по якому ми отримуємо дані від першого МК
TRISB6=1;//вивід керування роботою реле
```

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;//асинхроний режим
TX9=0;// робота 9-ти бітної посилки
TXEN=1;//дозвіл включення передавача
//------RCSTA---------------
RX9=0;//дозвіл роботи 9-ти бітрної посилки
CREN=1;//дозвіл включення передавача
SPEN=1;//включення модуля UART
}
```

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

```
void main(void)  {
        MK_init(); //модуль ініціалізації МК

while(1) {

T= getch(); //читання отриманого значення температуру з першого МК
grn=getch();//читання граничного значення, утвореного кнопками на першому МК
```

Наступний фрагмент здійснює підготовку даних для виведення на РКІ:
Спершу розділимо число з DS18B20 на окремі цифри:

```
        T1=(T /10)%10;
        T2= T %10;
```

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

```
        if(T <= grn)//якщо значення напруги менше граничного, то:
        {
        RB6=0; //реле залишається виключеним
        pol1=0b0010101;//змінна для виведення букви “N”
        pol2=0b0111111;//змінна для виведення букви “O”

        }
```

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

```
        else {
        RB6=1;
        pol1=0b1000111;//змінна для виведення букви “F”
        pol2=0b0111111;//змінна для виведення букви “O”
                }

        }
}
```

Це функція, яка дозволяє нам організувати безперервну динамічну індикацію:

```
void interrupt DYN(void)
{
if(T0IE&&T0IF)
{
```

```
switch(cnt)
{
case 0: R5=1;PORTC=MAS[];R0=0;R1=R2=R3=R4=1;break;
case 1: R0=1;PORTC=pol2; R1=0;R2=R3=R4=R5=1;break;
case 2: R1=1;PORTC=pol1; R2=0;R0=R3=R4=R5=1;break;
case 3: R2=1;PORTC=MAS[];R3=0;R1=R0=R4=R5=1;break;
case 4: R3=1;PORTC=MAS[T2];R4=0;R1=R0=R2=R5=1;break;
case 5: R4=1;PORTC=MAS[T1];R5=0;R1=R0=R2=R3=1;break;
}
cnt++;
if(cnt==6){cnt=0;}
TMR0=0;
T0IF=0;//очищення прапорця T0IF
}
}
```

Рисунок 3.3 зображує роботу схеми коли значення на DS18B20 не перевищує граничне значення, яке ми встановили за допомогою керуючих кнопок. Це означає, що в даному режимі роботи наше реле знаходиться у виключеному режимі.

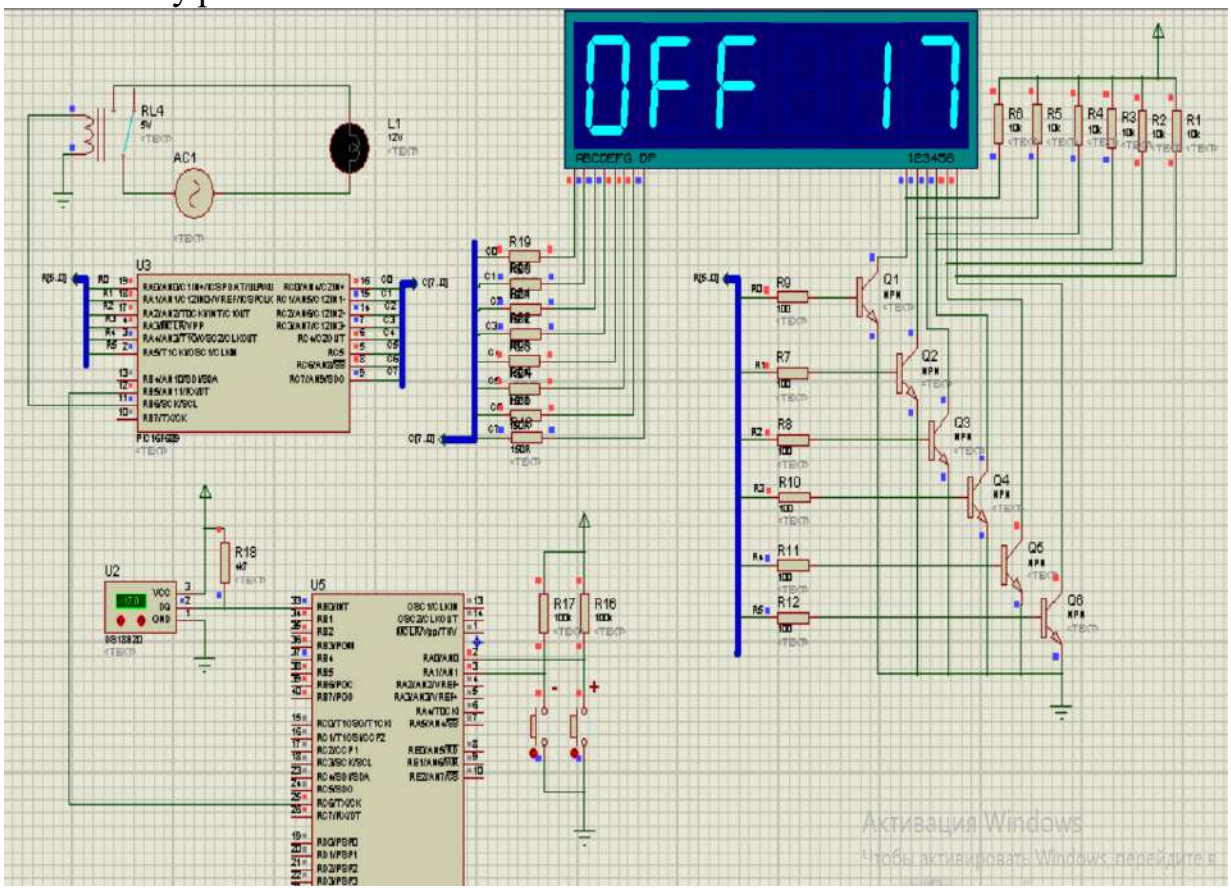

Рисунок 3.3 - Робоча схема з виключеним реле

На наступному рисунку (рис. 3.4) зображено схему, яка працює із даними температури DS18B20, які є вищі від встановленого граничного значення. В даному випадку у нас включається реле та виводиться відповідна інформація на ССІ.

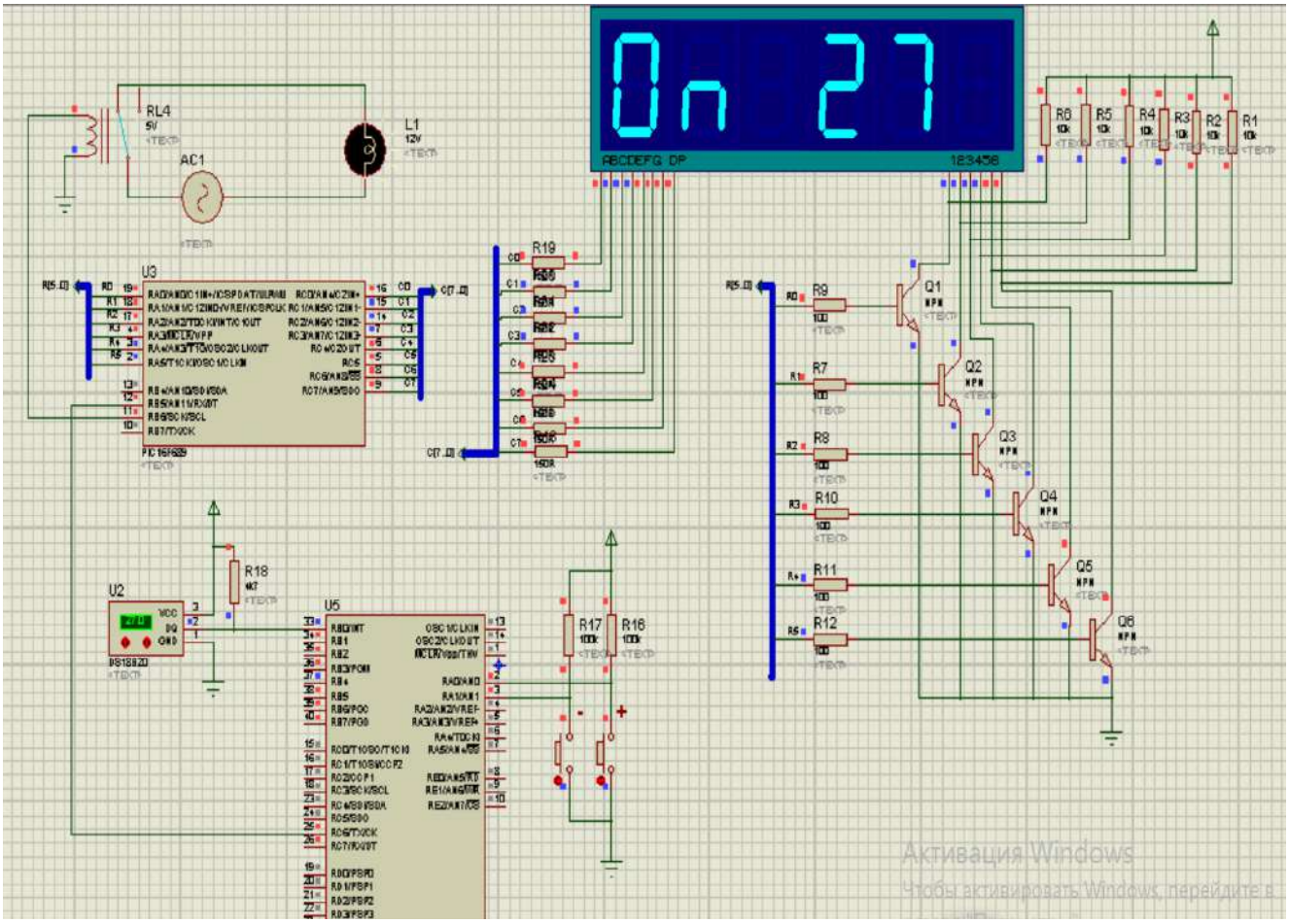

Рисунок 3.4 - Робоча схема з включеним реле

ВИСНОВКИ

В процесі курсового проєктування розглянуто принципи проєктування пристроїв і систем, що побудовані з використанням однокристальних мікроконтролерів фірми Microchip. Під час виконання проєктування 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 влагодження програмного забезпечення використано середовище розробки MPLAB 8.92. У цьому середовищі створено проект, написано текст програми з використанням мови С, 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 проєкт може бути основою розробки керуючої системи збору даних чи системи керування та контролю за виконанням завдань.

ПЕРЕЛІК ПОСИЛАНЬ

1. PIC-контролери та MPLAB: програмування на асемблері: практ посіб./ С.В.Войтко – Київ:КПІ ім. Ігоря Сікорського, видавництво «Політехніка», 2017 – 160с.
2. Войтович І. В., Сидоренко В. М. Мікропроцесорні системи та мікроконтролери. — Київ: КНТ, 2020.
3. Мазур О. М. Мікроконтролери AVR: програмування мовою С. — Львів: Львівська політехніка, 2018.
4. Погорілий Ю. С. Вбудовані системи на базі мікроконтролерів. — Київ: КПІ ім. І. Сікорського, 2021.
5. Barrett S. F., Pack D. J. Atmel AVR Microcontroller Primer: Programming and Interfacing. — Morgan & Claypool Publishers, 2019.
6. Han-Way Huang. The Atmel AVR Microcontroller: MEGA and XMEGA in Assembly and C. — Cengage Learning, 2017.
7. Jack Ganssle. The Art of Designing Embedded Systems. — Newnes, 2020.
8. HI-TECH C® FOR PIC10/12/16 USER'S GUIDE© 2010 Microchip Technology Inc., -364c