Міністерство освіти і науки України

Прикарпатський національний університет

імені Василя Стефаника

Фізико-технічний факультет

Кафедра комп'ютерної інженерії та електроніки

Т.Г. Бенько.

МІКРОКОНТРОЛЕРИ. НАВЧАЛЬНО-МЕТОДИЧНИЙ ПОСІБНИК

м. Івано-Франківськ

– 2023 –

УДК 621.382:004(076.5)
Б-46

Бенько Т.Г.Мікроконтролери :Навчально – методичний посібник. – Івано-Франківськ, Прикарпатський національнийуніверситет ім. Василя Стефаника, 2023. – 151 с.

Навчальний посібник поєднує питання, що стосуються побудови і моделювання електричних схем на основі Мікроконтролерних пристроїв та вивчення їх основних параметрів. Навчальний посібник розроблено відповідно до навчальних програм дисципліни «Мікроконтролери» для студентів спеціальності «Електроніка» та інших спеціальностей, в навчальному плані яких є аналогічна дисципліна.

Рецензенти:

Рачій І.Б. – доктор фізико-математичних наук, професор кафедри матеріалознавства і новітніх технологій Прикарпатського національного університету імені Василя Стефаника

Никируй Л.І. – кандидат фізико-математичних наук, професор кафедри фізики і хімії твердого тіла Прикарпатського національного університету імені Василя Стефаника

Затверджено на засіданні кафедри комп’ютерної інженерії та електроніки (протокол №7 від “28” лютого 2023 р.)

ЗМІСТ

ПЕРЕДМОВА 5
1 ОСНОВИ ТЕОРІЇ МІКРОПРОЦЕСОРНИХ ПРИСТРОЇВ 6
1.1 Класифікація мікропроцесорів. Варіанти архітектури 6
1.2 Базова структура мікропроцесорної системи 9
1.3 Основні характеристики МП 11
1.4 Типова структура мікропроцесора 12
1.5 Система команд МП і режими адресації 16
1.6 Організація передачі інформації в МПС. Інтерфейс 20
1.6.1 Способи передачі інформації 24
1.6.2 Методи обміну інформацією в мікропроцесорній системі 29
2 МІКРОКОНТРОЛЕРИ 42
2.1 Структура МК 45
2.2 Процесорне ядро МК 48
2.3 Резидентна пам'ять МК 50
2.4 Порти введення/виводу 54
2.5 Таймери і процесори подій 58
2.6 Аналогово-цифрові й цифро-аналогові перетворювачі 69
2.7 Мінімізація споживання енергії в МП системах 70
2.8 Моніторинг напруги живлення МК 75
2.9 Апаратні і програмні рішення щодо підвищення надійності роботи МК 78
4 КОНТРОЛЕРИ Intel MCS-51 80
3.1 Структурна організація Intel 8051 81
3.1.1 Арифметично-логічний пристрій 81
3.1.2 Резидентна пам'ять 83
3.1.3 Зовнішня пам'ять 85
3.1.4 Пристрій керування і синхронізації 85
3.2 Програмна модель ОМК 88
3.3 Система команд Intel 8051 91
3.4 Периферійні пристрої ОМК 97
3.4.1 Порти уведення/виводу 97
3.4.2 Лічильники/таймери 103
3.4.3 Послідовний порт 107
4 МІКРОКОНТРОЛЕР PIC16F877 115
4.1 Характеристика мікроконтролера 115
4.2 Структурна схема мікроконтролера PIC16F877 116
4.3 Організація пам'яті 118
4.3.1 Пам’ять програм 119
4.3.2 Організація пам'яті даних 120
4.4 Регістр стану STATUS 122
4.5 Регістр OPTION 124
4.6 Регістр INTCON 125
4.7 Лічильник команд 126
4.8 Стік 127
4.9 Порти введення/виводу 128
4.9.1 Регістри PORTA і TRISA 128
4.9.2 Регістри PORTB і TRISB 130
4.9.3 Регістри PORTC і TRISC 131
4.9.4 Регістри PORTD і TRISD 132
4.9.5 Регістри PORTE і TRISE 133
4.10 Таймери 133
4.10.1 Модуль таймера TMR0 133
4.10.2 Модуль таймера TMR1 135
4.10.3 Модуль таймера TMR2 138
4.11 Модуль 10-розрядного АЦП 139
4.12 Переривання 143
4.13 Сторожовий таймер WDT 144
146
147
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 150

ПЕРЕДМОВА

При цифровому способі обробки інформації кожній змінній величині в системі ставиться відповідно її цифровий код. Функціональні залежності в системі реалізуються шляхом безпосереднього розв'язання рівнянь системи тими або іншими чисельними методами за заздалегідь закладеною програмою. Пристрій, що реалізує це розв'язання, називається процесором.

Перший процесор, як програмно функціонуючий пристрій, здатний виконувати арифметичні й логічні операції, а так само здійснювати розгалуження алгоритму свого функціонування залежно від результату попередніх обчислень, був створений в 40-і роки попереднього сторіччя у США фахівцями фірми IBM. Він являв собою пристрій на електромеханічних реле, займав кілька поверхів будинку, мав украй низьку швидкодію й надійність і був придатний лише для дуже вузького класу специфічних обчислень. З прогресом електронної техніки вдосконалилася й елементна база для побудови процесорів. З'являлися процесори на електронних лампах, транзисторах, дискретних логічних мікросхемах малого ступеня інтеграції. У міру вдосконалювання процесори мали все менші габаритні розміри, споживали менше енергії, мали більшу продуктивність і надійність. Однак вони залишалися мало придатними для виконання операцій керування в реальному масштабі часу, а тому використалися в основному тільки для певного класу обчислювальних завдань.

Дійсна революція в обчислювальній техніці відбулася після появи першого так званого мікропроцесора, тобто процесора, виконаного у вигляді однієї мікросхеми великого ступеня інтеграції. Це був 4-розрядний мікропроцесор 4004 фірми INTEL (1971 р.). У 1973 р. фірма INTEL випускає 8-розрядний мікропроцесор 8080, а в 1978 р. – 16-розрядний мікропроцесор 8086, що мав 29 тисяч транзисторів на кристалі й початкову вартість 360$. Основними напрямами еволюції мікропроцесорів були (і є) збільшення розрядності одночасно зроблених обчислень і зменшення часу виконання обчислень.

1 ОСНОВИ ТЕОРІЇ МІКРОПРОЦЕСОРНИХ ПРИСТРОЇВ

1.1 Класифікація мікропроцесорів. Варіанти архітектури

З погляду користувача при виборі мікропроцесора доцільно мати у своєму розпорядженні деякі узагальнені комплексні характеристики можливостей мікропроцесора. Розроблювач має потребу в з'ясуванні й розумінні лише тих компонентів мікропроцесора, які явно відбиваються в програмах і повинні бути враховані при розробці схем і програм функціонування системи. Такі характеристики визначаються поняттям архітектури мікропроцесора.

Архітектура мікропроцесора – це його логічна організація, розглянута з погляду користувача; вона визначає можливості мікропроцесора за апаратною і програмною реалізаціями функцій, необхідних для побудови мікропроцесорної системи. Поняття архітектури мікропроцесора відображає:

- його структуру, тобто сукупність компонентів, що становлять мікропроцесор, і зв'язків між ними; для користувача досить обмежитися регістровою моделлю мікропроцесора;
- способи подання й формати даних;
- способи доступу до всіх програмно-доступних для користувача елементів структури (адресація до регістрів, комірок постійної й оперативної пам'яті, зовнішніх пристроїв);
- набір операцій, виконуваних мікропроцесором;
- характеристики керуючих слів і сигналів, вироблюваних мікропроцесором, і тих, які входять до нього ззовні;
- реакцію на зовнішні сигнали (система обробки переривань і т. п.).

За способом організації простору пам'яті мікропроцесорної системи розрізняють два основних типи архитектур.

Організація, при якій для зберігання програм і даних використовується один простір пам'яті, називається *фон-неймановскою архітектурою* (за ім’ям математика, який запропонував кодування програм у форматі, що відповідає формату даних) або *Прінстонською* (за ім’ям лабораторії Прінстонського університету, яка запропонувала її). Програми й дані зберігаються в єдиному просторі, і немає ніяких ознак, що вказують на тип інформації в комірці пам'яті. Перевагами такої архітектури є більше проста внутрішня структура мікропроцесора й менша кількість керуючих сигналів.

Організація, при якій пам'ять програм CSEG (Code Segment) і пам'ять даних DSEG (Data Segment) відокремлені й мають свої власні адресні простори й способи доступу до них, називається *Гарвардською архітектурою* (за ім’ям лабораторії Гарвардського університету, яка запропонувала її). Така архітектура є більше складною й вимагає додаткових керуючих сигналів. Однак вона дозволяє здійснювати більш

гнучкі маніпуляції інформації, реалізовувати компактно кодований набір машинних команд і, у ряді випадків, прискорювати роботу мікропроцесора.

У цей час випускаються мікропроцесори зі змішаною архітектурою, у яких CSEG і DSEG мають єдиний адресний простір, однак різні механізми доступу до них.

За кількістю команд мікропроцесори поділяються на мікропроцесори з повним набором команд CISC (Complete Instruction Set Computer) і обмеженим набором команд RISC (Reduce Instruction Set Computer). Останні працюють швидше через простоту схеми, але урізані команди доводиться заміняти декількома. Відносно недавно з'явилася VLIV (Very Large Instruction Word) архітектура. Її особливістю є використання дуже довгих команд (до 128 і більше бітів), окремі поля яких вміщують коди, які забезпечують виконання різних операций. Така команда викликає виконання декількох оперцій паралельно в різних операціонних пристроях.

Класифікація сучасних МП за функціональними відзнаками показана на рисунку 1.1.

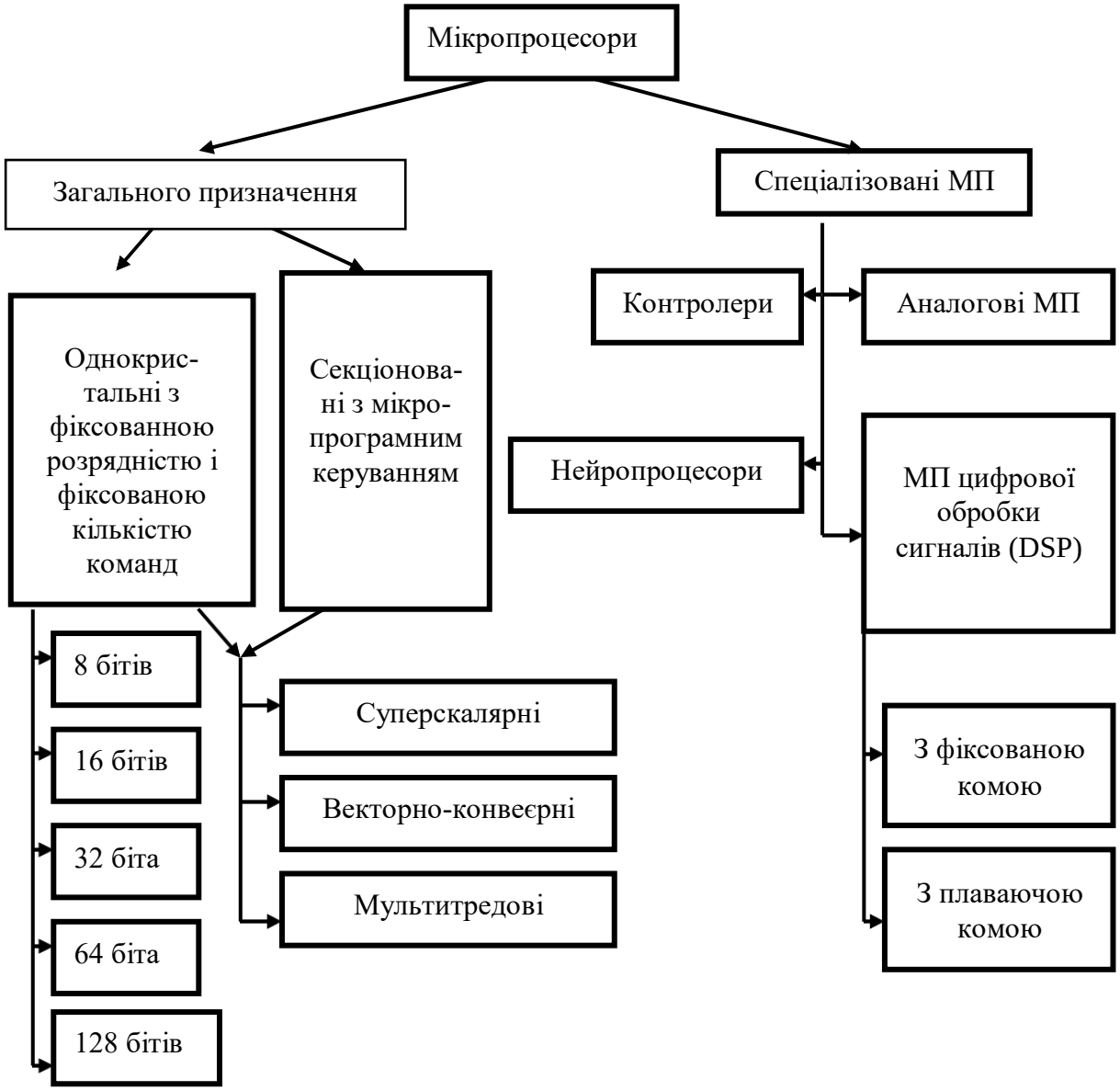


Рисунок 1.1 – Класифікація сучасних МП за функціональними відзнаками

За призначенням мікропроцесори поділяються на:

- *МП загального призначення (універсальні).* Вони призначені для розв'язання широкого кола задач. При цьому ефективна продуктивність процесора трохи залежить від специфіки завдання.

- *МП спеціалізовані.* Вони призначені для розв'язання якогось одного класу завдань. При цьому раціональна схемна побудова дозволяє вирішувати ці завдання найефективніше. Завдання інших класів або вирішуються набагато повільніше або не вирішуються взагалі. Спеціалізація мікропроцесора, тобто його проблемна орієнтація на прискорене виконання певних функцій, дозволяє різко збільшити ефективну продуктивність при рішенні тільки певних завдань.

Багатокристальні *секційні* мікропроцесори виходять у тому випадку, коли у вигляді БІС реалізуються частини (секції) логічної структури процесора. Микропрцессорная секція – це БІС, призначена для обробки декількох розрядів даних або виконання певних керуючих операцій. Секціонованість БІС МП визначає можливість «нарощування» розрядності оброблюваних даних або ускладнення пристроїв керування мікропроцесором при паралельному включенні великої кількості БІС. Багатокристальні *секційні* мікропроцесори мають розрядність від 2…4 до 8…16 біт і дозволяють створювати високопродуктивні процесори ЕОМ.

Суперскалярні, векторно-конвеєрні та *мультитредові* МП поєднують переваги однокристальних та секціонованих МП і застосовуються для побудови паралельних обчислювальних структур.

МП-контролер або *мікроконтролер* – обчислювально-керуючий пристрій, призначений для виконання функцій контролю й керування периферійним устаткуванням. Ухил убік керування накладає відбиток на особливість архітектури мікроконтролерів. Основною із цих особливостей є те, що поряд із процесорним ядром мікроконтролери мають у своєму складі підсистему введення/виводу й, можливо, підсистему пам'яті. В останньому випадку іноді прийнято говорити про однокристальні мікро-ЕОМ.

За видом оброблюваних вхідних сигналів розрізняють цифрові й аналогові мікропроцесори. Самі мікропроцесори – це цифрові пристрої, однак можуть мати убудовані аналого-цифрові й цифро-аналогові перетворювачі. Тому вхідні аналогові сигнали передаються в МП через перетворювач у цифрову форму, обробляються й після зворотного перетворення в аналогову форму надходять до виходу. З погляду архітектури такі МП являють собою аналогові функціональні перетворювачі сигналів і називаються *аналоговими мікропроцесорами*. Вони можуть виконувати функції будь-якої аналогової схеми.

Нейропроцесори призначені для сторення нейронних мереж керування та для реалізації алгоритмів «нечіткої» логіки.

Процесори цифрової обробки сигналів DSP (Digital Signal Procesor). Їхня архітектура має особливості, що дозволяють їм з найбільшою

продуктивністю здійснювати алгоритми рекурентної обробки даних, які використовуються в багатьох завданнях, що вимагає їхнього виконання в масштабі “реального часу”, таких як аудіо- і відеокодування, регулювання, цифрова фільтрація, цифровий зв'язок і т. п.

1.2 Базова структура мікропроцесорної системи

Основні вузли мікропроцесорної системи (МПС) показані на рисунку 1.2.

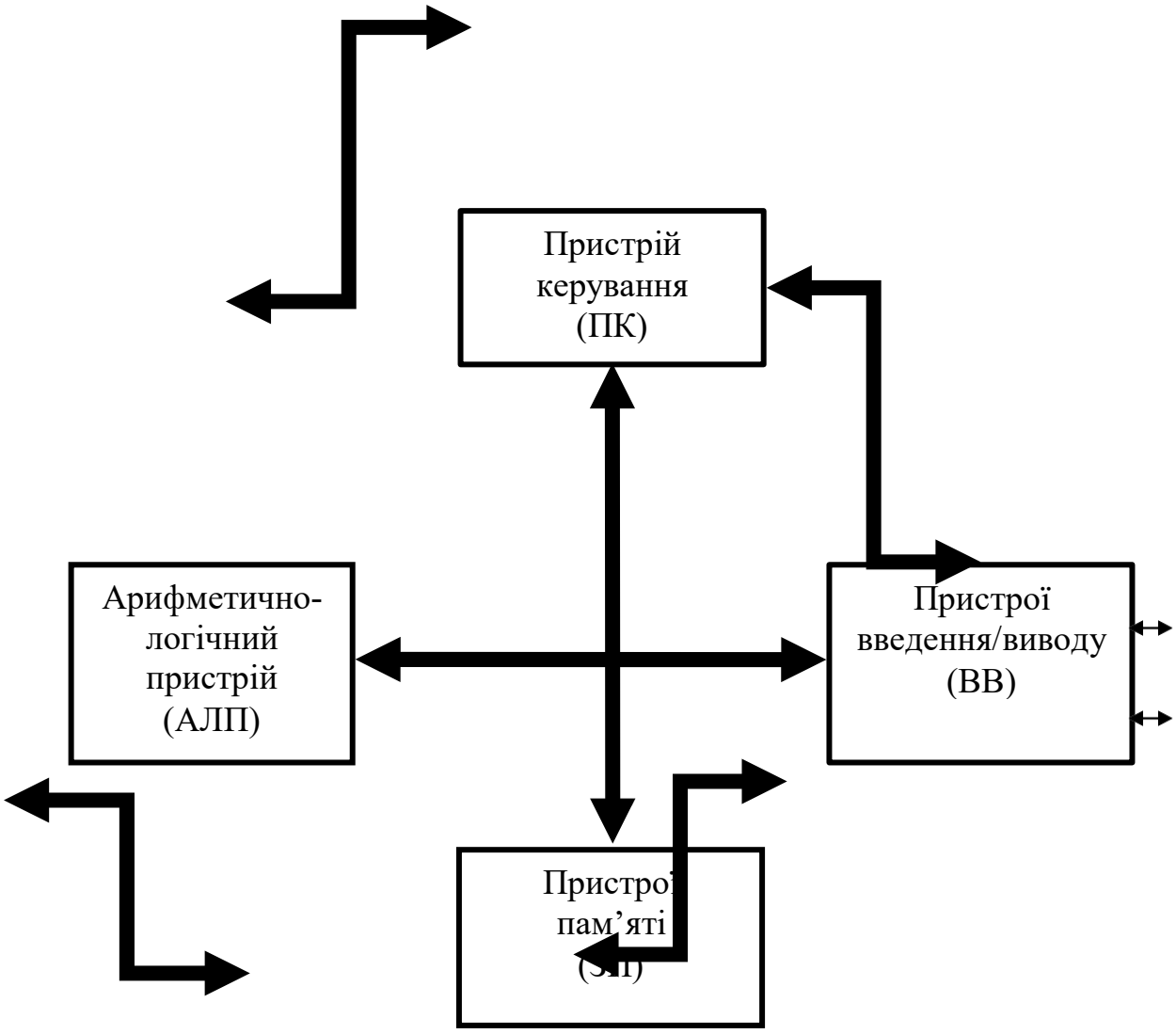


Рисунок 1.2 – Структура мікропроцесорної системи

На пристрій керування (ПК) покладається завдання виконання всіх програмних дій, необхідних відповідно до алгоритму роботи пристрою. Арифметично-логічний пристрій (АЛП) виконує арифметичні і логічні операції над кодами. У блоці пам'яті (ЗП) зберігаються команди програми функціонування процесора, а також значення констант і змінних величин,

що беруть участь в обчисленнях. Блок введення/виводу (ВВ) виконує функцію сполучення мікропроцесорної системи з об'єктом керування. ПК об’єднаний з АЛП називають центральним процесорним елементом (ЦПЕ), або просто центральним процесором (ЦП). ЦП зчитує з пам'яті команди, які складають програму й дешифрує їх. Відповідно до результату дешифрування команд він здійснює вибірку даних з пам'яті або пристроїв уведення, обробляє їх і пересилає назад до пам’яті або пристроїв виводу. Існує також можливість введення/виводу даних з пам'яті на зовнішні пристрої й назад, минаючи ЦП. Зв'язок між процесором й іншими пристроями МПС може здійснюватися за принципами радіальних зв'язків, загальної шини або комбінованим способом. На рисунку 1 показана структура з радіальними зв'язками. Усі блоки безпосередньо зв'язані один з одним. Перевагою цієї структури є максимальна швидкодія, тому що всі блоки можуть обмінюватися даними одночасно. До недоліків такої структури варто віднести велику кількість міжблокових зв'язків і складність розширення такої МПС. Для усунення недоліків була розроблена структура із загальними шинами (рис. 1.3), але недоліки усунуті за рахунок втрати швидкодії. У сучасних МПС структура із загальними шинами найбільш поширена.

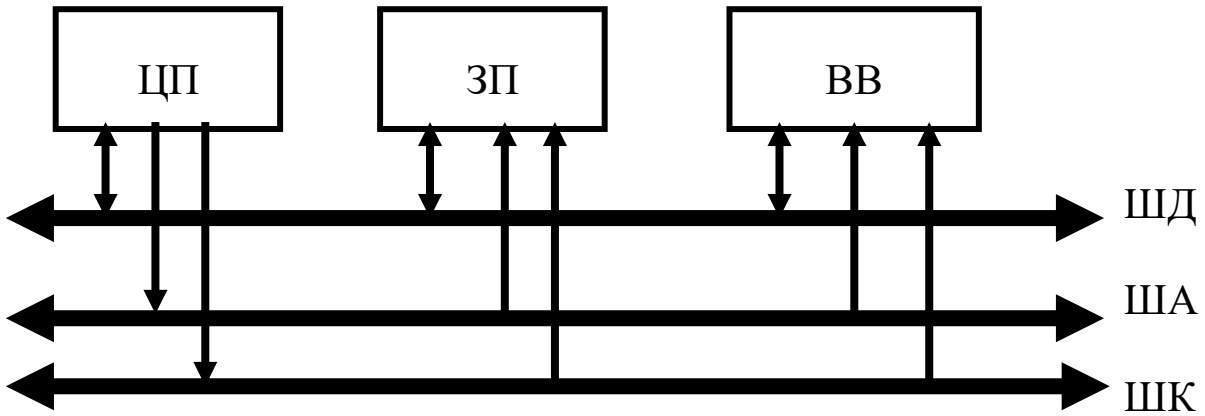


Рисунок 1.3 – Структура МПС із загальними шинами

Усі вузли мікропроцесорної системи з'єднані між собою за допомогою трьох основних шин:

ШД – шина даних (DATA bus),

ША – шина адреси (ADDR bus),

ШК – шина керування (CONTROL bus).

Усі разом вони утворять системну шину.

Шина даних. Ця шина призначена для передачі даних від мікропроцесора до периферійних пристроїв, а так само у зворотному напрямку. Розрядність шини даних визначається типом застосовуваного

процесора. У простих мікропроцесорах шина даних звичайно має 8 розрядів. Сучасні процесори можуть мати шину даних у 16, 32, 64 розрядів.

Шина адреси. Як і шина даних, шина адреси являє собою набір провідників, якими проходить передача двійкових чисел в електронній формі. Однак, на відміну від шини даних, двійкові числа, передані шиною адреси, мають інший зміст і призначення. Вони являють собою адресу комірки пам'яті або порту уведення виводу, до якого в цей момент звертається процесор. Кількість розрядів адресної шини відрізняється більшою розмаїтістю. Від кількості розрядів шини адреси залежить те, яку кількість комірок пам'яті може адресувати процесор. Процесор, що має шістнадцятирозрядну шину адреси, може звертатися до 2^{16} (тобто до 65536) комірок пам'яті. Це число називається обсягом пам'яті, що адресується. Для адресації портів уведення/виводу так само потрібна шина адреси. В обох випадках у цій ролі виступає та сама шина. Але ні одній мікропроцесорній системі ніколи не може знадобитися така сама велика кількість портів уведення/виводу, скільки звичайно буває комірок пам’яті. Тому у процесі обміну з портами уведення/виводу процесор використовує тільки вісім (рідше 16) молодших розрядів шини адреси.

Шина керування. Ця шина не має такої ж чіткої структури, як шина даних або шина адреси. У шину керування умовно поєднують набір ліній, що передають різні керуючі сигнали від процесора на всі периферійні пристрої й назад. У будь-якій шині керування обов'язково присутні лінії, що передають наступні сигнали:

RD (Read) – сигнал читання,
WR (Write) – сигнал запису,
MREQ – сигнал, ініціалізації пристроїв пам'яті (ОЗП або ПЗП),
IORQ – сигнал ініціалізації портів уведення/виводу.
Крім того до сигналів шини керування відносяться:
READY – сигнал готовності,
RESET – сигнал скидання,
І ще кілька спеціальних сигналів, про які йтиметься пізніше.

1.3 Основні характеристики МП

1 Тип мікроелектронної технології:
- побудований на біполярній технології;
- побудований на МОП- технології;
- спільної технології.

2 Довжина слова: 4-, 8-, 16-, 32-, 64-розрядний.
3 Швидкодія (тактова частота), час виконання команд.
4 Ємність адресованої пам'яті.

5 Типи пристрою керування: схемний, мікропрограмний.

6 Ефективність системи команд:

- кількість команд;
- способи адресації;
- виконання операцій;
- наявність команд роботи зі стекпам'яттю;
- наявність команд, оперуючих з бітами, десятковими числами, числами із плаваючою комою.

7 Кількість рівнів переривання.

8 Можливість прямого доступу до пам'яті.

9 Пропускна здатність каналу введення/виводу.

10 Кількість і рівні живильної напруги.

11 Споживана потужність.

12 Номінальні параметри випробуваних сигналів.

13 Наявність і доступність для користувача апаратно-програмувальних засобів підтримки проектування й відлагодження МП-пристроїв і систем.

14 Розміри кристала, кількість елементів на кристалі.

15 Тип корпусу кристала.

1.4 Типова структура мікропроцесора

Типова структура мікропроцесора наведена на рисунку 1.4. Мікропроцесор складається із трьох основних блоків: арифметично-логічного пристрою (АЛП), резидентного запам’ятовуючого пристрію (РЗП) і пристрою керування.

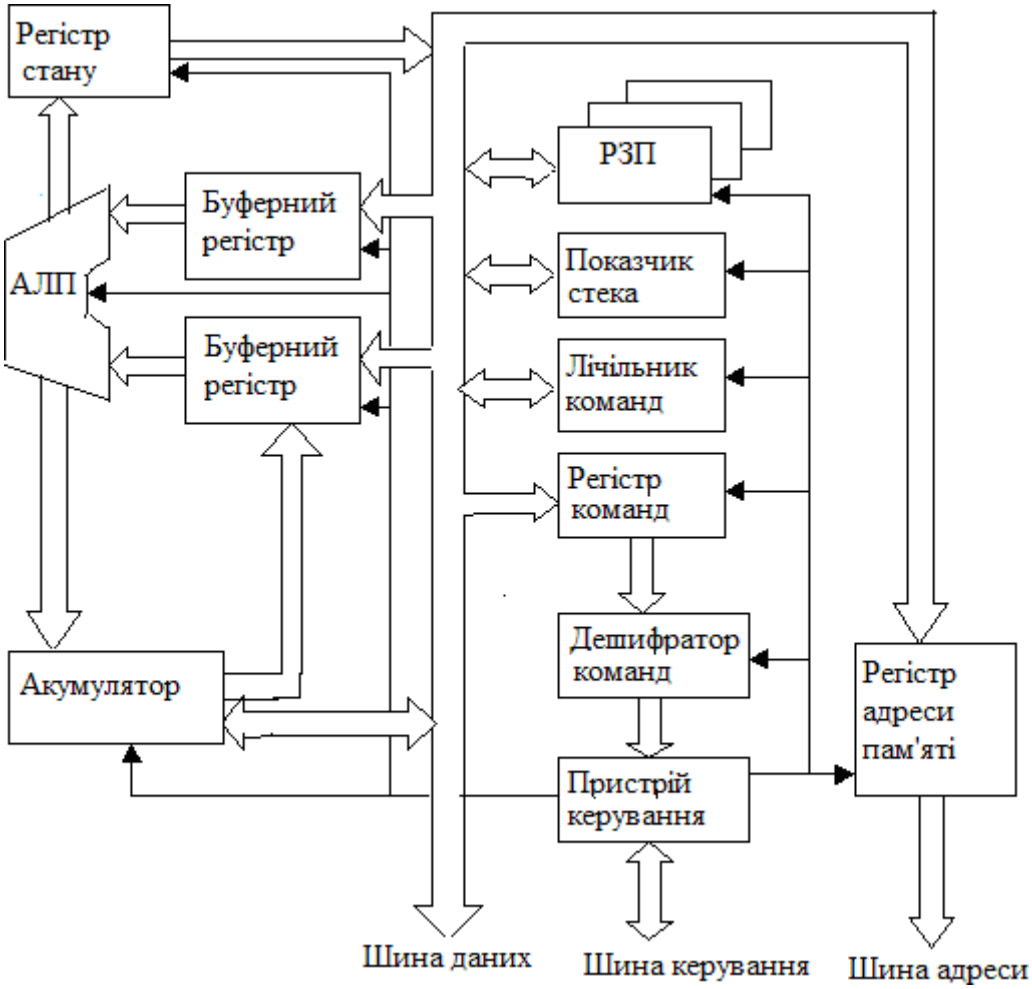


Рисунок 1.4 – Типова структурна схема мікропроцесора

Для передачі даних між цими блоками використовується внутрішня шина даних. АЛП виконує одну з головних функцій мікропроцесора – обробку даних. Перелік функцій АЛП залежить від типу мікропроцесора. Операції, виконувані АЛП більшості мікропроцесорів, наступні: додавання, вирахування, логічне додавання, логічне множення, виключне АБО, інверсія, зрушення вправо, зрушення вліво, збільшення на 1, зменшення на 1.

Кожний регістр мікропроцесора можна використати для тимчасового зберігання одного слова даних. Деякі регістри мають спеціальне призначення, інші – багатоцільове. Останні називаються регістрами загального призначення (РЗП) і можуть використатися програмістом за його розсудом. Кількість і призначення регістрів залежить від типу МП. Розглянемо призначення основних регістрів, наявних майже у всіх мікропроцесорах.

Акумулятор – це головний регістр при різних маніпуляціях з даними. Більшість арифметичних і логічних операцій здійснюється шляхом використання АЛП й акумулятора. Кожна з таких операцій над двома словами даних (операндами) припускає розміщення одного з них в акумуляторі, а іншого – в пам'яті або якому-небудь регістрі. Результат

виконання операції АЛП теж звичайно розміщується в акумуляторі, вміст якого при цьому втрачається. Операцією іншого типу, яку використовує акумулятор, є передача даних з однієї частини мікропроцесора в іншу. Наприклад, пересилання даних між портом введення/виводу й пам'яттю, між двома областями пам'яті й т. д.

Виконання такої операції здійснюється двома етапами: спочатку виконуються пересилання даних із джерела в акумулятор, потім – з акумулятора в пункт призначення. Мікропроцесор може виконувати деякі дії над даними безпосередньо в акумуляторі. Наприклад, акумулятор можна очистити (скинути) шляхом запису в усі його розряди двійкових нулів, установити в одиничний стан шляхом запису в усі його розряди двійкових одиниць. Уміст акумулятора можна зрушувати вліво або вправо, інвертувати, тобто значення нулів замінити на значення одиниць у всіх розрядах і навпаки.

Лічильник команд – це один з найбільш важливих регістрів мікропроцесора. Як відомо, програма – це послідовність команд (інструкцій), збережених у пам'яті й призначених для того, щоб інструктувати процесор, як вирішувати поставлене завдання. Для правильного виконання програми команди повинні надходити в певному порядку. Лічильник команд забезпечує формування адреси чергової команди, записаної в пам'яті. Перед виконанням програми лічильник команд необхідно завантажити адресою, що вказує на першу команду програми. Звичайно, це нульова адреса. Адреса першої команди програми посилається через регістр адреси пам'яті адресною шиною до схем керування пам'яттю, у результаті чого з пам'яті зчитується вміст першої команди. Далі ця команда передається в спеціальний регістр мікропроцесора, називаний *регістром команд.* Після добування команди з пам'яті мікропроцесор автоматично дає збільшення вмісту лічильника команд. У такий спосіб виконується послідовність команд, розташованих у пам'яті одна за одною. Лічильник команд можна завантажити іншим умістом при виконанні особливої групи команд. Може виникнути необхідність виконати частину програми, що «випадає» з послідовності команд основної (головної) програми. Наприклад, таку частину програми, що повторюється в процесі виконання всієї програми. Замість того, щоб писати цю частину програми щоразу, коли в ній виникає необхідність, її записують один раз і вертаються до її повторного виконання, відступаючи від зазначеної послідовності. Частина програми, виконувана шляхом відступу від послідовності команд головної програми, називається *підпрограмою*. У цьому випадку в лічильник команд безпосередньо записується необхідна адреса.

Лічильник команд має звичайно більше розрядів, чим довжина слова даних мікропроцесора. Так, у більшості 8-розрядних мікропроцесорів, кількість розрядів лічильника команд дорівнює 16. Розрядність лічильника команд повинна забезпечити можливість вибірки команди з будь-якої області пам'яті програм (з будь-якої комірки пам'яті). Якщо кількість

розрядів лічильника команд дорівнює 16, то за допомогою цих розрядів з пам'яті можна вибрати 2^{16} комірок пам'яті програм, тобто 65536 комірок.

Регістр команд містить команду в процесі її дешифрування й виконання. Код команди надходить через шину даних з пам'яті.

Регістр адреси пам'яті. При кожному звертанні до пам'яті процесор указує адресу комірки пам'яті, що підлягає використанню. У комірці пам'яті може перебувати або код команди, або дані, над якими виконується дія.

Буферний регістр призначений для тимчасового зберігання (буферування даних).

Регістр стану (регістр ознак або регістр прапорців) призначений для зберігання результатів деяких перевірок, здійснюваних у процесі виконання програми. Розряди регістра стану набувають того або іншого значення при виконанні операцій, що використають АЛП й деякі регістри. Запам'ятовування результатів згаданих перевірок дозволяє використати програми, що містять переходи (порушення природної послідовности виконання команд). При наявності в програмі переходу за заданою ознакою виконання команди починається з деякої нової області пам'яті, тобто лічильник команд завантажується новим числом. У випадку умовного переходу така дія має місце, якщо результати певних перевірок збігаються з очікуваними значеннями. Зазначені результати перебувають у регістрі стану. Регістр стану надає програмістові можливість організувати роботу мікропроцесора так, щоб за певних умов мінявся порядок виконання команд. Розглянемо найчастіше використовувані розряди регістра стану.

1 *Перенос-позика.* Даний розряд указує, що остання виконувана операція супроводжувалася переносом або заємом (негативним переносом). Значення цього розряду встановлюється як 1, якщо в результаті додавання двох чисел має місце перенос зі старшого розряду АЛП. Приклад: 10001110+11000011=01010001. Позика фіксується в регістрі стану при вирахуванні більшого числа з меншого.

2 *Нульовий результат.* Цей розряд набуває одиничного значення, якщо після закінчення операції у всіх розрядах регістра результату виявлені двійкові нулі.

3 *Знаковий.* Набуває одиничного значення, коли старший біт регістра результату дорівнює 1. При виконанні арифметичних операцій з числами в додатковому коді одиничне значення старшого біта показує, що в регістрі перебуває негативне число.

Регістри загального призначення (РЗП). Більшість МП мають у своєму складі набір регістрів, використовуваних у якості надоперативних запам'ятовувальних пристроїв. Тому що АЛП може робити операції із умістом РЗП без виходу на зовнішню магістраль адрес і даних, то вони відбуваються багато швидше, ніж операції із зовнішньою пам'яттю. Кількість РЗП і можливість програмного доступу до них у різних мікропроцесорів не однакові.

Покажчик стека. Стік – набір регістрів мікропроцесора або комірок опретивної пам'яті, звідки дані або адреси вибираються “зверху” за принципом: першим вибирається елемент, що надійшов у стек останнім. При записі в стек чергового слова усі раніше записані слова зміщаються на один регістр униз. При вибірці слова зі стека слова, що залишилися, переміщаються на один регістр нагору. Стік звичайно використається в мікропроцесорах для зберігання адрес повернення при звертанні до підпрограм, а також для запам'ятовування стану внутрішніх регістрів при обробці переривань. Зупинимося докладніше на використанні стека при організації підпрограм. Якщо в програмі є підпрограма, то на виконання підпрограми можна перейти з любого місця програми за командою CALL <мітка>, де CALL – мнемонічне позначення команди, а <мітка> – будь-яке символьне позначення рядка програми, де розташована перша команда підпрограми. Остання команда підпрограми має позначення RET, що означає «вихід з підпрограми». Команда виходу з підпрограми не має явної вказівки на адресу команди, яку потрібно виконати після цього, тобто не зазначена адреса повернення з підпрограми. Отож, при виконанні команди CALL адреса повернення з підпрограми, а це адреса команди CALL, збільшена на 1, зберігається в стеці. При виконанні команди RET адреса витягається зі стека й надходить у лічильник команд, тобто далі виконується команда, що записана в програмі за командою CALL.

У такий спосіб процес функціонування стека нагадує роботу з пачкою документів, коли кожний новий документ кладеться зверху пачки. При такій організації стека необхідний спеціальний регістр – *покажчик стека* для зберігання адреси останнього за часом надходження елемента стека.

Пристрій керування. Роль схеми керування полягає в підтримці необхідної послідовності функціонування всіх інших ланок МП. За сигналами схеми керування чергова команда витягається з регістра команд. При цьому визначається, що необхідно робити з даними, а потім забезпечується послідовність дій для виконання поставленого завдання.

Одна з головних функцій схеми керування – декодування команди, що перебуває в регістрі команд, за допомогою дешифратора команд, що у результаті видає сигнали, необхідні для її виконання.

Крім зазначених вище дій схеми керування виконують деякі спеціальні функції: керування послідовністю включення живлення й процесами переривань. Переривання – це свого роду запит, що надходить на схеми керування від інших пристроїв (пам'яті, пристроїв вводу-виводу). Переривання пов'язане з використанням внутрішньої шини даних мікропроцесора. Схеми керування приймають рішення, коли й у якій послідовності інші пристрої можуть користуватися внутрішньою шиною даних.

Система шин. На характеристики мікропроцесора великий вплив робить спосіб організації його зв'язку із зовнішнім середовищем – пристроями введення/виводу (ПВВ) і запам'ятовувальними пристроями

(ЗП). За способом організації зв'язків із зовнішнім середовищем розрізняють мікропроцесори з мультиплексованою шиною адреси й даних і з роздільними шинами адреси й даних.

У мікропроцесорах з тимчасовим мультиплексуванням шини адреси й даних при звертанні до зовнішнього пристрою на загальній шині якийсь проміжок часу виставляється адреса, а потім шина надається даним. Такі шини вимагають включення додаткового регістра адреси (Рга), у який записується адреса за сигналом “адреса – дані”, поки адреса перебуває на загальній шині.

1.5 Система команд МП і режими адресації

МП працює в складі МПС, обмінюючись інформацією з пам'яттю й зовнішніми пристроями. В основі роботи МП лежить *командний цикл* – дії за вибором з пам'яті й виконанням однієї команди. Залежно від типу й формату команди, способів адресації й кількості операндів командний цикл може містити в собі різну кількість звертань до пам'яті й зовнішніх пристроїв й мати різну тривалість.

Будь-який командний цикл (КЦ) починається з добування з пам'яті першого байта команди за адресою, що зберігається в лічільнку команд. Команди можуть мати довжину 1, 2 або 3 байти, причому в першому байті міститься інформація про довжину команди. У випадку 2- або 3-байтової команди реалізуються додаткові звертання до пам'яті за сусідніми (більшими) адресами.

Після зчитування команди починається її виконання, причому в процесі виконання може знадобитися ще одне або кілька звертань до пам'яті або зовнішнього пристрою (читання операнда, запис результату).

Дії МПС з передачі в/із МП одного байта даних/команди називаються *машинним циклом*.

Командний цикл являє собою послідовність машинних циклів (МЦ). МЦ обов'язково містить у собі дії з передачі байта інформації. Крім того у деяких МЦ додатково реалізуються дії з пересилання і/або перетворення інформації усередині МП. Тому тривалість МЦ може бути різною: за рахунок різної кількості машинних тактів, що вміщуються в них (Т1, Т2...).

Машинний такт пов’язаний із сигналами тактового генератора, тому тривалість такту постійна – період тактового генератора.

Таким чином, проглядається ієрархія процедур при роботі мікропроцесора:

Командний цикл → *Машинний цикл* → *Машинний такт.*

Кожному такту відповідає певний стан керуючого автомата. Будь-який МЦ обов'язково містить такти, призначені для передачі байта інтерфейсом. МЦ, у яких здійснюється передача і/або перетворення інформації в МП, містять додатково один або два такти.

Таким чином, у машинному циклі виконуються наступні дії:

- видача адреси;
- видача інформації про початий МЦ (PSW);
- аналіз значення вхідних сигналів;
- при необхідності – очікування сигналу READY = 1;
- прийом/видача даних;
- при необхідності – внутрішня обробка/пересилання даних.

При реалізації одного МЦ процесор може:

1) прийняти з пам'яті байт команди;
2) прийняти з пам'яті байт даних;
3) прийняти з ПВВ байт даних;
4) прийняти зі стека байт даних;
5) прийняти вектор переривання;
6) видати на запис байт даних;
7) видати в стек байт даних;
8) видати на ПВВ байт даних.

Усі команди являють собою послідовність байт, що містить комбінації 1 і 0. Конструкція команд зі специфікацією різних груп біт називається форматом команди.

Частина команди, що визначає виконувану дію, називається *кодом операції (КОП).*

Будь-яка адреса, або дані, необхідні для виконання операції, називається *операндом*.

Найбільш гнучка операція вимагає до 3 операндів.

Якщо байт містить 8 біт, а для завдання адреси потрібно 16 біт, то чотирьохоперандова команда займе 8 байт пам'яті, без коду операції.

Тобто якщо основний критерій – максимальна гнучкість, то виходить повільний МП з величезною пам'яттю, отже, у більшості МП для команди використають не більше 2 операндів. Це досягається:

- адреса команди вказується тільки в командах переходу. В інших командах основна команда вибирається з комірок пам'яті;

- використання комірки, у якій перебуває один з операндів для запам'ятовування операндів.

Подальше скорочення кількості байт досягається розміщенням одного з них або обох у регістрах ЦП, що мають короткі адреси.

У деяких командах обходяться тільки одним операндом – одноопернадні команди.

У двоопернадних командах один операнд звичайно змінюється командою, а другий – ні. Тому що інформація береться тільки з одного операнда, цей операнд називається джерелом. Операнд, зміст якого змінюється, називається *одержувачем*.

Локалізацію й звертання до операнду забезпечують режими адресації. При введенні декількох режимів адресації необхідно відвести в команді біти, що вказують режим адресації для кожного операнда.

У всіх форматах команд перший байт приділяється для коду операції. Інші байти повинні визначати операнди або їхні комірки, і тому вони використовуються для запам'ятовування адреси регістра, адреси пам'яті.

Класифікація команд за основними ознаками подана на рисунку 1.5. Найважливішим структурним елементом формату будь-якої команди є код операції (КОП), що визначає дію, яка має бути виконана. Велика кількість КОП у процесорі дуже важлива, тому що апаратна реалізація команд заощаджує пам'ять і час. Кількість бітів, що відводиться під КОП, є функцією повного набору реалізованих команд. При використанні фіксованої кількості бітів під КОП для кодування всіх команд необхідно в поле КОП виділити достатню кількість двійкових розрядів. Однак, з огляду на обмежену довжину слова МП, різне функціональне призначення команд, джерела й приймачі результатів операцій, а також те, що не всі команди містять адресну частину для звертання до пам'яті й периферійних пристроїв, у МП для кодування команд широко використовується принцип кодування зі змінною кількістю бітів під поле КОП для різних груп команд.

Для взаємодії з різними модулями в МПС повинні бути засоби ідентифікації комірок зовнішньої пам'яті, комірок внутрішньої пам'яті, регістрів МП і регістрів пристроїв уведення/виводу. Тому кожній із запам'ятовувальних комірок привласнюється адреса, тобто однозначна комбінація біт. Кількість біт визначає кількість ідентифікованих комірок.

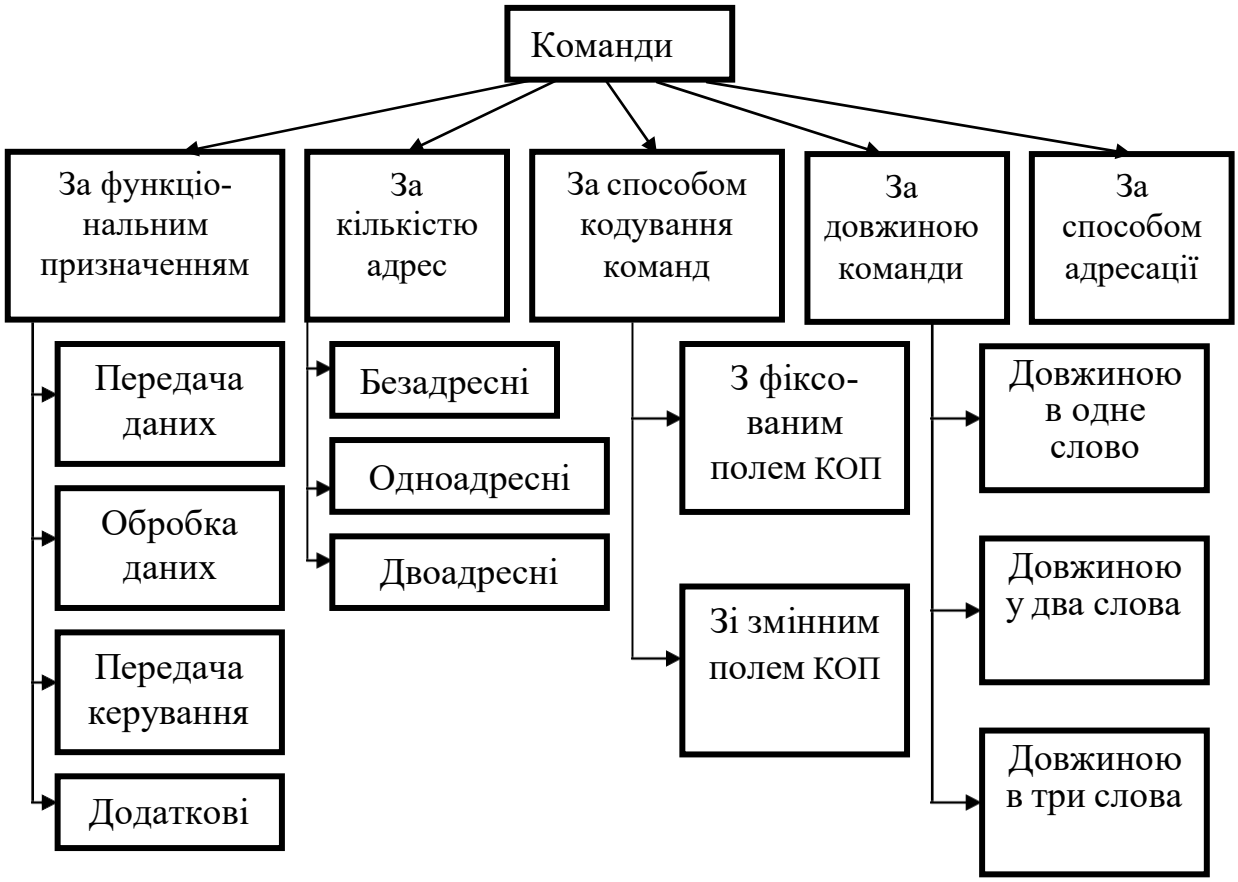


Рисунок 1.5 – Класифікація команд

Звичайно МП має різні адресні простори пам'яті й регістрів МП, а іноді – окремі адресні простори регістрів пристроїв уведення/виводу й внутрішньої пам'яті. Крім того пам'ять зберігає як дані, так і команди. Тому для МП розроблена безліч способів звертання до пам'яті, називаних режимами адресації.

Режим адресації пам'яті – це процедура або схема перетворення адресної інформації про операнда на його виконавчу адресу.

Усі способи адресації пам'яті можна поділити на:

1 Неявну адресацію: адреса джерела й приймача зазначені в команді неявно (вони убудовані в команді MOV A, B).

2 Безпосередню адресацію: адреса команди є безпосередньою частиною самої команди, за кодом операції записані дані, які надає програміст при запису команди.

3 Пряму адресацію: 1-й байт – код операції, 2-й байт – адреса, звідки беруться дані, 3-й байт – куди записується результат.

4 Регістрову адресацію: схожа на неявну, де адреси регістрів є частиною команди.

5 Непряму адресацію: у комірці, адреса якої визначається як частина команди, визначається адреса операнда, комірка може бути регістром, отже, регістр непрямої адресації або комірка пам'яті.

6 Базову адресацію: адреса утвориться в результаті додавання вмісту комірки пам'яті і регістра з певним числом, використається з непрямою адресацією, застосовна до масивів або для переміщення програми в пам'яті.

7 Індексну адресацію: декремент адреси, коли МП звертається до суміжних або рівно розподілених адрес, досягається шляхом послідовного зменшення адреси або шляхом додатка до фіксованої адреси числа, на якому відбувається інкремент або декремент; індексування в основному застосовується для послідовної адресації елементів масиву.

8 Автоінкремент: автоматичне збільшення або зменшення адреси на 1.

9 Адресацію базової сторінки: різновид непрямої адресації, коли зазначена адреса є адресою комірки в сторінці, а вміст комірки являє собою потрібну адресу.

10 Відносну адресацію: адреса дорівнює сумі числа й поточного вмісту програмного лічильника, число – адреса операнда щодо команди й звичайно є частиною команди, але може втримуватися в робочому регістрі.

1.6 Організація передачі інформації в МПС. Інтерфейс

До складу мікропроцесорної системи крім мікропроцесора (або мікропроцесорів) залежно від її призначення може входити різна кількість пристроїв постійної й оперативної пам'яті й периферійних пристроїв – зовнішніх запам'ятовувальних пристроїв на магнітних дисках і стрічках і різноманітних пристрїв уведення/виводу (дисплеїв, друкувальних пристроїв, аналого-цифрових і цифро-аналогових перетворювачів, різних датчиків і приймачів інформації й т.п.). При цьому повинна забезпечуватися можливість зв'язку між цими пристроями й обміну інформацією між ними з необхідною швидкістю.

Передача інформації із процесора й пам'яті в периферійний пристрій називається операцією виводу, а з периферійного пристрою в процесор або пам'ять – операцією введення.

Пристрої МПС зв'язуються один з одним за допомогою сполучень, називаних інтерфейсами.

Інтерфейс являє собою сукупність ліній і шин, сигналів, електронних схем, алгоритмів (протоколів) процедур, що забезпечують обмін інформацією між пристроями системи. Продуктивність, надійність і ефективність використання МПС визначається не тільки характеристиками пристроїв, які входять до її складу, але й характеристиками інтерфейсів, що зв'язують пристрої системи. Створення ефективної й гнучкої організації взаємодії й обміну інформацією між пристроями ускладнюється тим, що поєднувані в систему пристрої розрізняються за фізичними принципами дії, виконуваними робочими операціями, використовуваними командами і наказами, керуючими сигналами (кодами) і форматами даних, швидкостями передачі інформації. Входячі до складу системи, периферійні пристрої, а також устаткування, пов'язане із МП – системою технологічного процесу – працюють асинхронно один до одного й до процесора (програми), і запити з їх боку на встановлення зв'язку й обмін інформацією можуть виникати в довільні моменти часу.

Організація взаємодії й обміну інформацією між пристроями (модулями) МПС повинна забезпечувати:

- можливість реалізувати мікропроцесорні системи з різною конфігурацією (різним складом пристроїв), включати в систему нові пристрої без яких-небудь переробок в апаратурах, а лише шляхом додавання програм, що обслуговують ці пристрої;

- можливість ефективної реалізації обміну інформацією в системі, що містить пристрої зі значно відмінними швидкостями, передачі даних, причому в умовах, коли запити на операції введення/виводу від пристроїв системи й із зовнішнього, стосовно системи, середовища надходять у довільні моменти часу (асинхронно щодо програми, виконуваної процесором) і мають різну відносну терміновість виконання;

- можливість паралельного в часі виконання процесором програми, а периферійними пристроями – операцій введення/виводу;

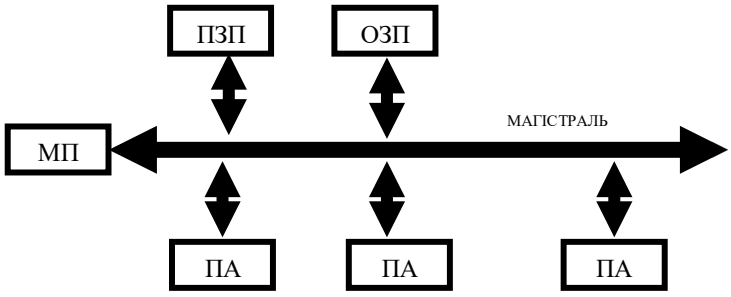
- спрощення й уніфікацію програмування операцій уведення - виводу з виключенням необхідності урахування особливостей того або іншого периферійного пристрою.

Зазначені вище вимоги вдається реалізувати на основі наступних архітектурних рішень, характерних для побудови мікропроцесорних засобів.

Магістрально-модульна організація мікропроцесорних систем, яка полягає в тому, що окремі мікропроцесорні засоби виконуються у вигляді конструктивно закінчених модулів, які являють собою відповідну БІС, укладені в корпуси з виводами, і ці модулі поєднуються в систему за допомогою загальних шин (магістралей), поділюваних пристроями (модулями) у часі при операціях передачі інформації (рис. 1.6).

Формати команд уведення/виводу й формати даних, використовуваних в операціях передачі інформації через інтерфейс, уніфіковані (не залежні від типу периферійного пристрою (ПП)). Перетворення уніфікованих форматів даних і команд на спеціальні формати й специфічні накази (керуючі коди) і сигнали, що відповідають окремим ПУ, виробляються в спеціальних електронних блоках керування (адаптерах або контролерах (ПА)), через які периферійні пристрої підключаються до загальних шин. Уніфікація поширюється на сімейство (серію) мікропроцесорних засобів.

Уніфікований інтерфейс – це уніфікований за складом й призначенням набір ліній і шин, уніфіковані схеми підключення, сигнали й алгоритми керування передачею інформації через інтерфейс.



Рисинок 1.6 – Магістрально-модульна структура МПС

У складі інтерфейсу є засоби (ліній керування й спеціалізованих схем) для здійснення системи переривань із програмно-керованим пріоритетом, що забезпечує обслуговування формованих периферійними пристроями запитів на передачу інформації через інтерфейс. Особливістю архітектури мікропроцесорних засобів є зв'язок системи переривання із шинами й процедурами роботи інтерфейсу.

Наявність декількох способів (режимів) передачі інформації між пристроями через інтерфейс підвищує гнучкість інтерфейсу, дозволяючи вибрати найбільш придатний режим з урахуванням характеристик ПП й структури переданого повідомлення (окреме слово або масив слів).

Важливою особливістю організації обміну інформацією в МПС є використання спеціалізованих інтерфейсних БІС (контролер прямого доступу, контролер переривань, програмувальний периферійний адаптер, програмувальний зв'язувальний адаптер, програмувальний таймер), що дозволяють значною мірою звільняти процесор від керування операціями введення/виводу й виконання допоміжних процедур перетворення форматів даних, підрахунку переданих байт та ін. Програмне налагоджування інтерфейсних БІС дає винятково широкі можливості для побудови гнучких і ефективних мікропроцесорних систем керування й обробки даних.

Підключення зовнішніх пристроїв до системної шини здійснюється за допомогою електронних схем, називаних контролерами ВВ (інтерфейсами ВВ або ПА). Вони узгоджують рівні електричних сигналів, а також перетворять машинні дані на формат, необхідний пристрою, і навпаки. Звичайно контролери ВВ конструктивно оформляються разом із процесором у вигляді інтерфейсних плат.

У процесі введення/виводу передається інформація двох видів: керуючі дані (слова) і саме дані, або дані-повідомлення. Керуючі дані від процесора, називані також командними словами або командами, ініціюють дії, не зв'язані безпосередньо з передачею даних, наприклад: запуск пристрою, заборона переривань і т.п. Керуючі дані від зовнішніх пристроїв називаються словами стану, вони містять інформацію про певні ознаки, наприклад: про готовність пристрою до передачі даних, про наявність помилок при обміні й т.п. Стан звичайно подається в декодованій формі – один біт для кожної ознаки.

Регістр, що містить групу біт, до якої процесор звертається в операціях ВВ, утворить порт ВВ. Таким чином, найбільш загальна програмна модель зовнішнього пристрою, що може виконувати введення й вивід, містить чотири регістри ВВ: регістр вихідних даних (вихідний порт), регістр вхідних даних (вхідний порт), регістр керування й регістр стану (рис. 1.7). Кожний із цих регістрів повинен мати однозначну адресу, що ідентифікується дешифратором адреси. Залежно від особливостей пристрою загальна модель конкретизується, наприклад, окремі регістри стану й керування поєднуються в один регістр, у пристрої введення (виводу) є тільки регістр вхідних (вихідних) даних, для введення й виводу використовується двонаправлений порт.

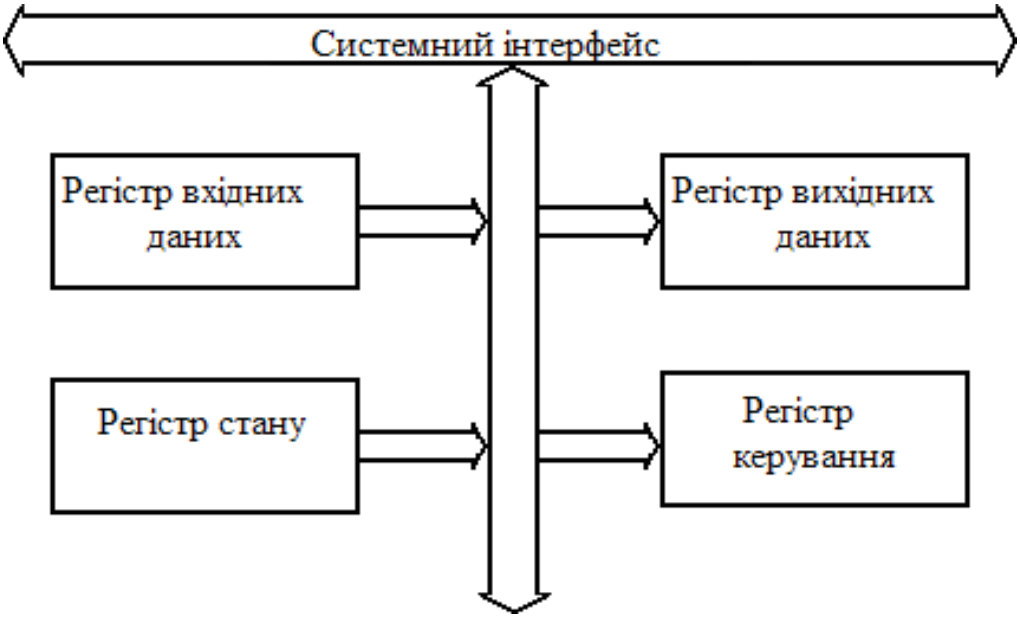


Рисунок 1.7 – Програмна модель зовнішнього пристрою

Безпосередні дії, пов'язані з уведенням/виводом, реалізуються одним із двох способів, що розрізняються адресацією регістрів ВВ.

Інтерфейс із ізольованими шинами характеризується роздільною адресацією пам'яті й зовнішніх пристроїв при обміні інформацією. Ізольований ВВ припускає наявність спеціальних команд уведення/виводу, загальний формат яких показаний на рисунку 1.8. При виконанні команди уведення IN уміст адресованого вхідного регістра PORT передається у внутрішній регістр REG процесора, а при виконанні команди OUT уміст регістра REG передається у вихідний порт PORT. У процесорі можуть бути й інші команди, що відносяться до ВВ і пов'язані з перевіркою й модифікацією вмісту регістра керування й стану.

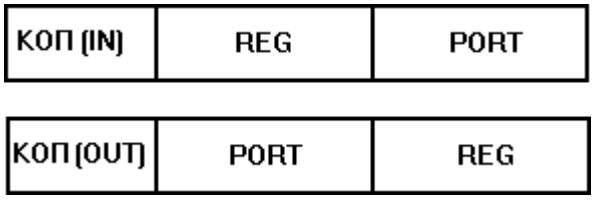


Рисунок 1.8 – Команди уведення/виводу (загальний формат)

Неважко помітити, що в цьому способі адресний простір портів уведення й виводу ізольовано від адресного простору пам'яті, тобто в МПС ту саму адресу можуть мати порт ВВ і комірка пам'яті. Поділ адресних просторів здійснюється за допомогою керуючих сигналів, що відносяться до систем ВВ і пам'яті (MEMRD# – зчитування даних з пам'яті, MEMWR# –

запис даних у пам'ять, IORD# – читання порту ВВ, IOWR# – запис у порт ВВ).

У МПС, розрахованій на ізольований ВВ, неважко перейти до ВВ, відображеному на пам'ять. Якщо, наприклад, адресний простір пам'яті становить 64 Кбайти, а для програмного забезпечення досить 32 Кбайта, то область адрес від 0 до 32 Кбайта використовується для пам'яті, від 32 Кбайт адо 64 Кбайта – для введення/виводу. При цьому ознакою, що диференціює звертання до пам'яті й портів ВВ, може бути старший біт адреси.

Таким чином, інтерфейс із загальними шинами (уведення/вивід з відображенням на пам'ять) має організацію, при якій частина загального адресного простору приділяється для зовнішніх пристроїв, регістри яких адресуються так само, як і комірки пам'яті. У цьому випадку для адресації портів ВВ використовуються повні адресні сигнали: READ – читання, WRITE – запис.

В операційних системах МПС є набір підпрограм (драйверів ВВ), керуючих операціями ВВ стандартних зовнішніх пристроїв. Завдяки їм користувач може не знати багатьох особливостей ЗП й інтерфейсів ВВ, а застосовувати чіткі програмні протоколи.

1.6.1 Способи передачі інформації

Способи передачі інформації: асинхронний, синхронний та синхронний-асинхронний.

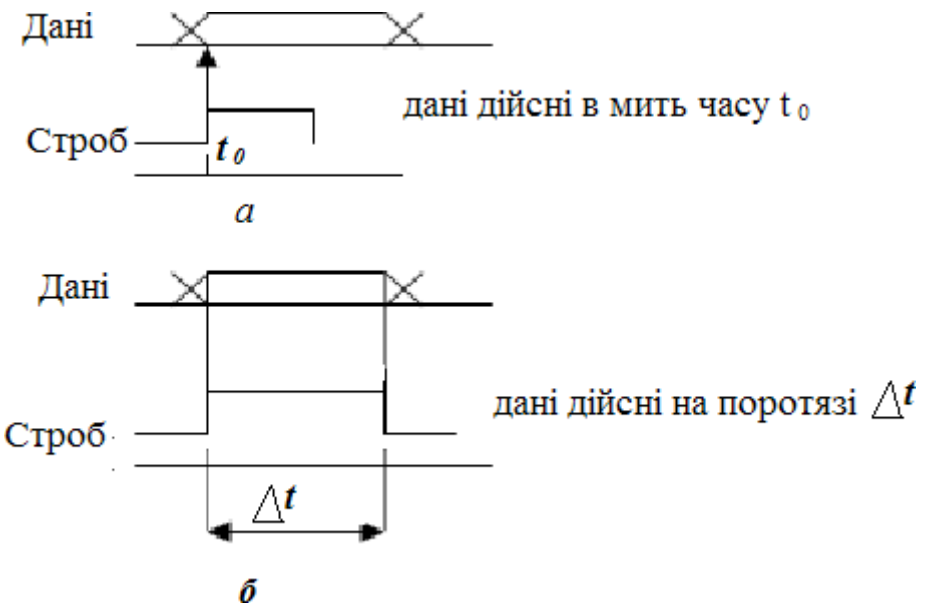
Асинхронний – сигнали передаються з довільними проміжками у часі .

Синхронний – сигнали передаються точно періодично в часі.

Синхронний-асинхронний – байти передаються асинхронно, а біти усередині байта передаються точно синхронно.

Існують два варіанти асинхронного способу: стробувння і «запит – відповідь» або квитування.

Стробування (строб – додатковий сигнал, що є підтвердженням дійсності інших сигналів) може здійснюватися фронтом або за рівнем (рис. 1.9).



а – стробування фронтом; б – стробування за рівнем

Рисунок 1.9 – Варіанти стробування

Варіант *а)* легко реалізується в апаратурі і має високу швидкодію, але момент перемикання програмним шляхом зафіксувати складно. Варіант б*)* легко реалізується апаратно і програмно, але має меншу швидкодію.

Варіант «запит – відповідь» пояснено на рисунку 1.10.

t1 – передавач виставляє дані (перед цим перевіряє на відсутність строб-відповіді).

t2 – передавач із затримкою виставляє сигнал строб-запиту.

t3 – приймач аналізує стан лінії строб-запиту, виявляє наявність активного сигналу й у цьому ж моменті здійснює прийом даних лінією даних.

t4 – передавач постійно сканує строб-відповідь лінію, виявляє, що строб-відповідь активний і скидає сигнал строб-запит.

t5 – приймач, скануючи строб-запит виявляє, що строб-запит не активний і скидає строб-відповідь.

Після t5 система перейшла у початковий стан.

Спосіб дозволяє сполучати апаратури з істотно різною швидкодією і легко реалізується програмним шляхом.

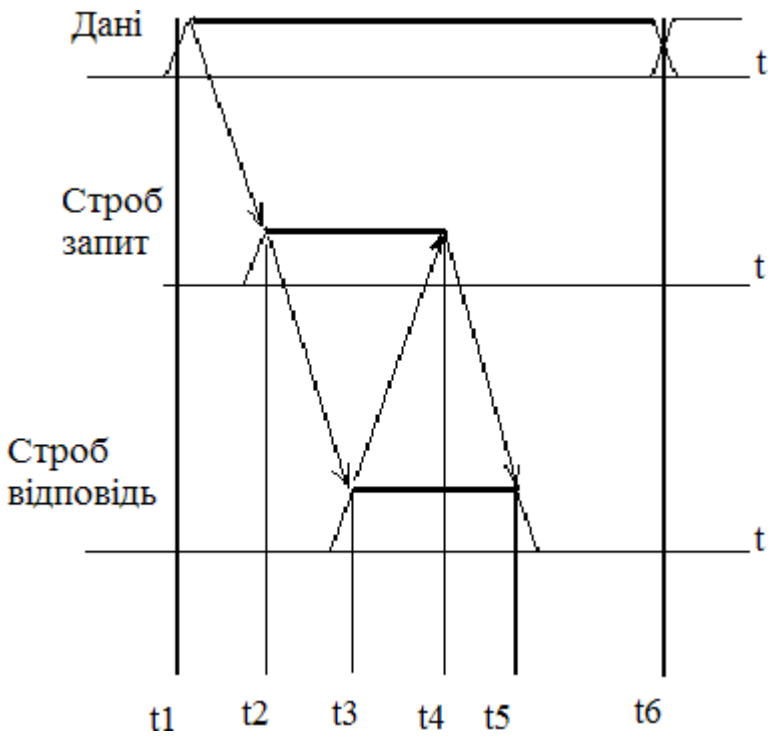


Рисунок 1.10 – Асинхронній спосіб «запит – відповідь»

Підсистема уведення/виводу у загальному випадку повинна забезпечувати виконання наступних функцій:

1) *узгодження форматів даних*, тому що процесор завжди видає/приймає дані в паралельній формі, а деякі ЗП (наприклад, НМД) – у послідовній. Із цього погляду розрізняють пристрої *паралельного* й *послідовного* обміну. У рамках паралельного обміну не відбувається перетворення форматів переданих слів, у той час як при послідовному обміні здійснюється перетворення паралельного коду на послідовний і навпаки. Усі варіанти, коли довжина слова ЗП (понад 1 біт) не збігається з довжиною слова МП, зводяться до різновидів паралельного обміну;

2) *организация режиму обміну* – формування й прийом керуючих сигналів, що ідентифікують наявність інформації на різних шинах, її тип, стан ЗП (*Готовий, Зайнято, Аварія)*, що регламентують тимчасові параметри обміну. За способом зв'язку процесора й ЗП (активний або пасивний) розрізняють *синхронний* і *асинхронний* обміни. При синхронному обміні тимчасові характеристики обміну повністю визначаються МП, що не аналізує готовність ЗП до обміну й фактичний час завершення обміну. Синхронний обмін можливий тільки із пристроями, завжди готовими до нього (наприклад, двійкова індикація). При асинхронному обміні МП аналізує стан ЗП й/або момент завершення обміну. Часові характеристики обміну в цьому випадку можуть визначатися ЗП;

3) *адресна селекція* зовнішнього пристрою.

Паралельна передача даних між контролером і ВУ є за своєю організацією найбільш простим способом обміну. Для організації паралельної передачі даних крім шини даних, кількість ліній у якій дорівнює кількості одночасно переданих бітів даних, використовується мінімальна кількість керуючих сигналів. Спосіб передечі – синхронний для пристроїв, завжди готових до обміну, та асинхронний «запит – відповідь» для пристроїв з меншою, ніж ЦП швидкодією. При паралельній передачі забезпечується досить висока швидкість обміну даними, що обмежується тільки швидкодією ЗП.

Послідовна передача даних. Використання послідовних ліній зв'язку для обміну даними із зовнішніми пристроями покладає на контролери ЗП додаткові в порівнянні з контролерами для паралельного обміну функції. По-перше, виникає необхідність перетворення формату даних: з паралельного формату, у якому вони надходять у контролер ЗП із системного інтерфейсу МПС, на послідовний при передачі у ЗП й з послідовного на паралельний при прийманні даних з ЗП. По-друге, потрібно реалізувати відповідному режиму роботи зовнішнього пристрою спосіб обміну даними: синхронний або асинхронний.

При організації послідовного обміну ключовими можуть уважатися дві проблеми:

1) синхронізація бітів передавача й приймача;
2) фіксація початку сеансу передачі.

Синхронний спосіб

Синхронізація буває внутрішня і зовнішня.

*Зовнішня синхронізація***.** Сигнали синхронізації надходять разом з даними. У цьому випадку форма сигналів може бути не правильною. Тому зовнішня синхронізація використовується тільки при передачі на невеликі відстані(тобто усередині плати). При синхронному методі передавач генерує дві послідовності – інформаційну Tx і синхроімпульси CLK, які передаються на приймач різними лініями (рис. 1.11).

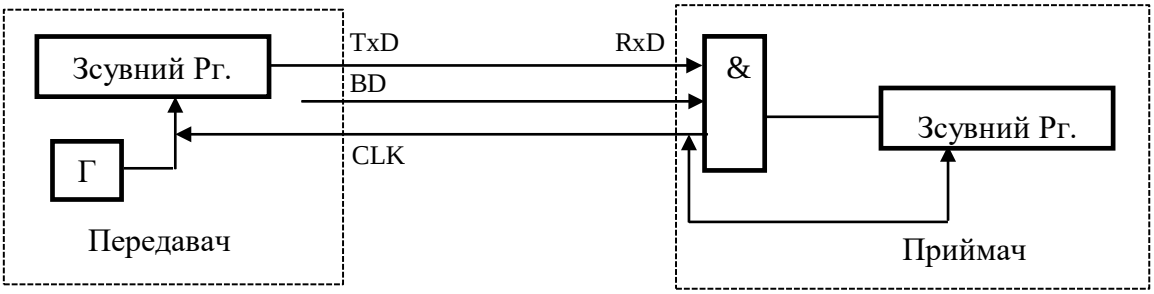


Рисунок 1.11 – Послідовний синхронний обмін із зовнішньою синхронізацією

Синхроімпульси забезпечують синхронізацію переданих бітів, а початок передачі відзначається по-різному. При організації *зовнішньої синхронізації* сигнал початку передачі BD генерується передавачем і передається на приймач спеціальною лінією (рис. 1.11), або, при

відсутності лінії BD, передача починається з появою імпульсів на лінії CLK.

У системах *із внутрішньою синхронізацією* відсутня лінія BD, а на лінію даних генеруються спеціальні коди довжиною 1…2 байта – символи синхронізації. Для кожного приймача попередньо визначаються конкретні синхросимволи, у такий спосіб можна здійснювати адресацію конкретного абонента з декількох, що працюють на одній лінії. Кожний приймач постійно приймає біти з Tx, формує символи й порівнює із власними синхросимволами. При збігу синхросимволов наступні біти надходять у канал даних приймача (рис. 1.12).

Рисунок 1.12 – Внутрішня синхронізація

SYN – спеціальний сигнал, що запускає тактовий генератор приймача.

Переваги:

- досить двох ліній(сигнал і земля);
- висока частота;
- висока надійність зв'язку;
- довжина пакета визначається взаємною синхронністю передавача й приймача.

Асинхронно-синхронний спосіб

Організація асинхронного послідовного обміну даними із зовнішнім пристроєм ускладнюється тим, що на передавальній і прийомній сторонах послідовної лінії зв'язку використовуються настроєні на одну частоту, але фізично різні генератори тактових імпульсів і, отже, загальна синхронізація відсутня.

Припустимо, що ми вміємо перетворювати кожний байт на потік одиниць і нулів, тобто біти, які можуть бути передані через середовище зв'язку (наприклад, телефонну лінію). Справді, універсальний асинхронний прийомопередавач (UART) виконує точно таку функцію. Звичайно, у той час, як лінія перебуває в режимі очікування, для демонстрації того, що лінія в порядку, нею передається одиниця, позначаючи незайнятість лінії. З іншого боку, коли лінія перебуває в стані логічного нуля, говориться, що вона перебуває в режимі витримування інтервалів. Таким чином, логічні одиниця й нуль розглядаються відповідно як MARK і SPACE.

В асинхронному зв'язку зміна умови стану лінії з MARK на SPACE означає початок символу (рис. 1.13). Це називається стартовим бітом. За стартовим бітом йде комбінація бітів, що являє символ, і потім біт контролю парності. Нарешті, лінія переходить у стан очікування MARK, що являє собою стоповий біт і означає кінець поточного символу. Кількість бітів, використовуваних для подання символу, називається довжиною слова й звичайно дорівнює семи або восьми. Контрольний біт (біт парності) використовується для виконання елементарної перевірки на наявність

помилки, формується за наступним правилом: якщо прийнято контроль на парність, то сума всіх бітів за модулем 2, включаючи біт парності, дорівнює 0.

Тривалість кожного біта визначається генераторами тактових імпульсів приймача й передавача. Відзначимо, однак, що генератори в приймачі й передавачі повинні мати ту саму частоту, але не потрібно, щоб вони були синхронізованими. Вибір частоти генератора залежить від швидкості передачі в бодах, що означає кількість змін стану лінії щосекунди. Номінально, тактова частота «16-кратна швидкість передачі в бодах» означає, що лінія перевіряється досить часто для надійного розпізнавання стартового біта.

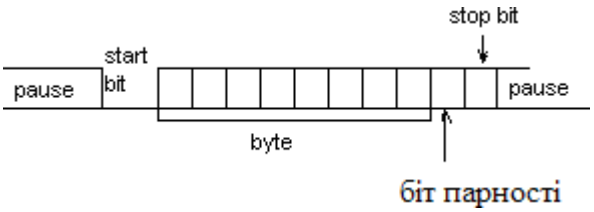


Рисунок 1.13 – Асинхронно-синхронний спосіб передачі інформації

Розглянуті принципи асинхронного послідовного зв'язку реалізовані в ряді стандартів для передачі інформації, серед яких найбільш популярним є стандарт RS-232C.

1.6.2 Методи обміну інформацією в мікропроцесорній системі

При пересиланні інформації між пристроями виникають проблеми:
- інформація подається в різних формах;
- різниця в логічних рівнях поданої інформації;
- різниця у швидкостях роботи різних пристроїв

Класифікація методів обміну інформацією в МПС показана на рисунку 1.14.

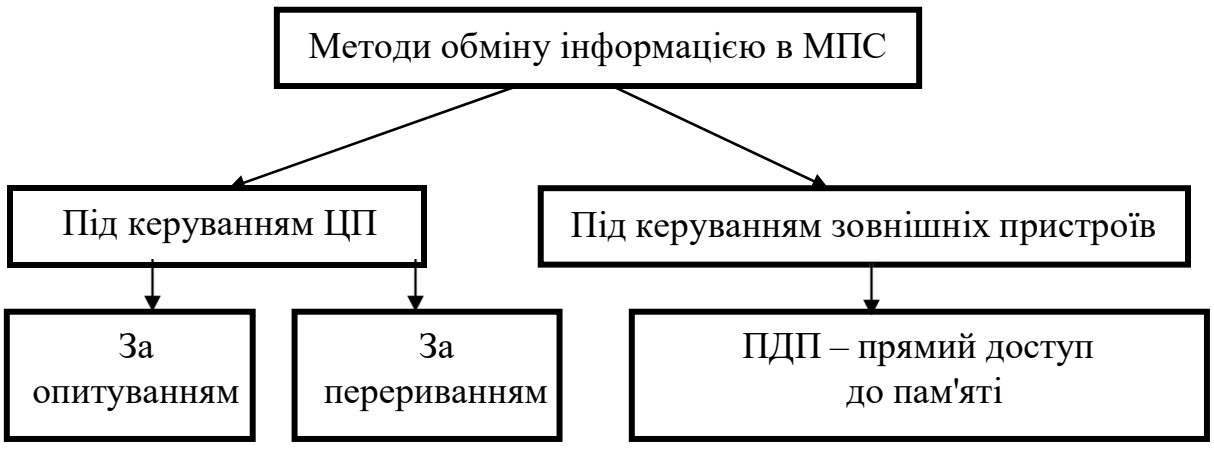


Рисунок 1.14 – Класифікація методів обміну інформацією в МПС

У МПС застосовуються три методи уведення/виводу: програмно-керований ВВ (називаний також програмним або за опитуванням або неформованим ВВ), ВВ за перериванням (форсований ВВ) і прямий доступ до пам'яті. Перший з них характеризується тим, що ініціювання й керування ВВ здійснюється програмою, виконуваною процесором, а зовнішні пристрої грають порівняно пасивну роль і сигналізують тільки про свій стан, зокрема, про готовність до операцій уведення/виводу. У другому режимі ВВ ініціюється не процесором, а зовнішнім пристроєм, що генерує спеціальний сигнал переривання. Реагуючи на цей сигнал готовності пристрою до передачі даних, процесор передає керування підпрограмі обслуговування пристрою, що викликав переривання. Дії, виконувані цією підпрограмою, визначаються користувачем, а безпосередніми операціями ВВ керує процесор. Нарешті, у режимі прямого доступу до пам'яті, що використовується, коли пропускної здатності процесора недостатньо, дії процесора припиняються, він відключається від системної шини й не бере участь у передачах даних між основною пам'яттю й швидкодіючим ВУ. Помітимо, що у всіх указаних вище випадках основні дії, виконувані на системній магістралі МПС, підкоряються двом основним принципам:

1 У процесі взаємодії будь-яких двох пристроїв МПС один з них обов'язково виконує активну, керуючу роль і є задатчиком, другий виявляється керованим, виконавцем. Найчастіше задатчиком є процесор.

2 Іншим важливим принципом, закладеним у структуру інтерфейсу, є принцип квитування (запиту – відповіді): кожний керуючий сигнал, посланий задатчиком, підтверджується сигналом виконавця. При відсутності відповідного сигналу виконавця протягом заданого проміжку часу формується так званий тайм-аут, задатчик фіксує помилку обміну й припиняє дану операцію.

Програмно-кероване введення/вивід. Даний режим характеризується тим, що всі дії щодо уведенню/виводу реалізуються командами прикладної програми. Найбільш прості ці дії виявляються для завжди готових зовнішніх пристроїв, наприклад індикатори на світлодіодах. При необхідності ВВ у відповідному місці програми використовуються команди IN або OUT. Така передача даних називається синхронним або безумовним ВВ.

Однак для більшості ЗП до виконання операцій ВВ треба переконатися в їхній готовності до обміну, тобто ВВ є асинхронним. Загальний стан пристрою характеризується прапором готовності READY, називаним також прапором готовності/зайнятості (READY/BUSY). Іноді стан готовності й зайнятості ідентифікуються окремими прапорами READY і BUSY, що входять до слова стану пристрою.

Процесор перевіряє прапор готовності за допомогою однієї або декількох команд. Якщо прапор установлений, то ініціюються саме уведення або вивід одного або декількох слів даних. Коли ж прапор скинутий, процесор виконує цикл із 2…3 команд із повторною перевіркою прапора READY доти, поки пристрій не буде готовий до операцій ВВ

(рис. 1.15). Даний цикл називається циклом очікування готовності ЗП й реалізується в різних процесорах по-різному.

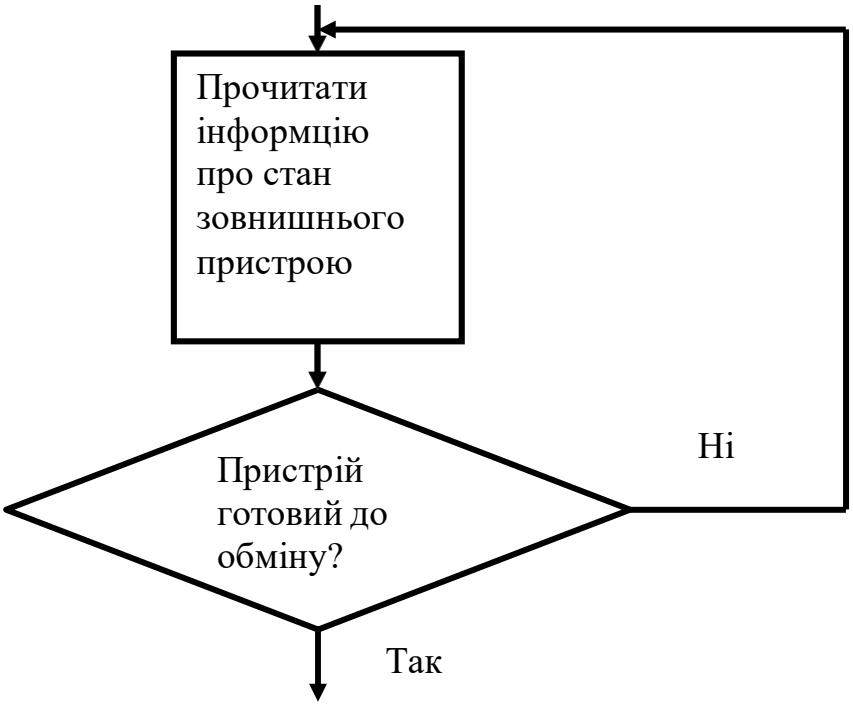


Рисунок 1.15 – Цикл програмного очікування готовності зовнішнього пристрою

Основний недолік програмного ВВ пов'язаний з непродуктивними втратами часу процесора в циклах очікування. До достоїнств варто віднести простоту його реалізації, що не вимагає додаткових апаратних засобів.

Організація переривань у МПС. Обмін з перериванням програми відрізняється від асинхронного програмно-керованого обміну тим, що перехід до виконання команд, що фізично реалізують обмін даними, здійснюється за допомогою спеціальних апаратних засобів. Команди обміну даними в цьому випадку виділяють в окремий програмний модуль – підпрограму обробки переривання. Завданням апаратних засобів обробки переривання в МПС саме і є припинення виконання однієї програми (її ще називають основною програмою) і передача керування підпрограмі обробки переривання. Дії, виконувані при цьому процесором, як правило, ті самі, що й при звертанні до підпрограми. Тільки при звертанні до підпрограми вони ініціюються командою, а при обробці переривання – керуючим сигналом від ЗП, який називають «Запит на переривання» або «Вимога переривання».

Ця важлива особливість обміну з перериванням програми дозволяє організувати обмін даними з ЗП в довільні моменти часу, що не залежать від програми, виконуваної в МПС. Таким чином, з'являється можливість обміну даними з ЗП в реальному масштабі часу, обумовленому зовнішнім стосовно МПС середовищем. Обмін з перериванням програми істотно

заощаджує час процесора, затрачуваний на обмін. Це відбувається за рахунок того, що зникає необхідність в організації програмних циклів очікування готовності ЗП, на виконання яких витрачається значний час, особливо при обміні з повільними ЗП.

Переривання програми на вимогу ВУ не повинне робити на перервану програму ніякого впливу крім збільшення часу її виконання за рахунок припинення на час виконання підпрограми обробки переривання. Оскільки для виконання підпрограми обробки переривання використовуються різні регістри процесора (лічильник команд, регістр стану й т.д.), то інформацію, що втримується в них у момент переривання, необхідно зберегти для наступного повернення в перервану програму.

Звичайно завдання збереження вмісту лічильника команд і регістра стану процесора покладається на апаратні засоби обробки переривання. Збереження вмісту інших регістрів процесора, використовуваних у підпрограмі обробки переривання, виробляється безпосередньо в підпрограмі. Звідси досить очевидний факт: чим більший обсяг інформації про перервану програму зберігається програмним шляхом, тим більший час реакції МПС на сигнал переривання, і навпаки. Кращими з погляду підвищення продуктивності МПС (скорочення часу виконання підпрограм обробки, а, отже, і основної програми) є зменшення кількрсті команд, що забезпечують збереження інформації про перервану програму, і реалізація цих функцій апаратними засобами.

Основними функціями системи переривання є:

- запам'ятовування стану перериваної програми і здійснення переходу до перериваючої програми;

- відновлення стану перерваної програми й повернення до неї.

При наявності декількох джерел запитів переривання має бути встановлений певний порядок в обслуговуванні запитів, що надійшли, тобто повинні бути встановлені пріоритетні співвідношення, що визначають, який з декількох запитів підлягає обробці в першу чергу, і має право або не має даний запит перервати ту або іншу програму.

Пріоритетний вибір запиту для виконання входить до процедури переходу до перериваючої програми.

Оцінка ефективності системи переривання визначається наступними характеристиками:

- загальна кількість запитів переривання, тобто кількість входів до системи переривання;

- час реакції ($\mathbf{t_p}$), тобто час між появою запиту переривання й початком виконання перериваючої програми для того ж самого запиту затримки у виконанні перериваючої програми залежать від того, скільки програм зі старшим пріоритетом очікує на обслуговування, тому час реакції визначається для запиту з найвищим пріоритетом;

- витрати часу на перемикання програм:

$$t_{пер} = t_з + t_в ,$$

де $t_з$ – час запам'ятовування;

$t_в$ – час відновлення.

Спрощена часова діаграма процесу переривання показана на рисунку 1.16.

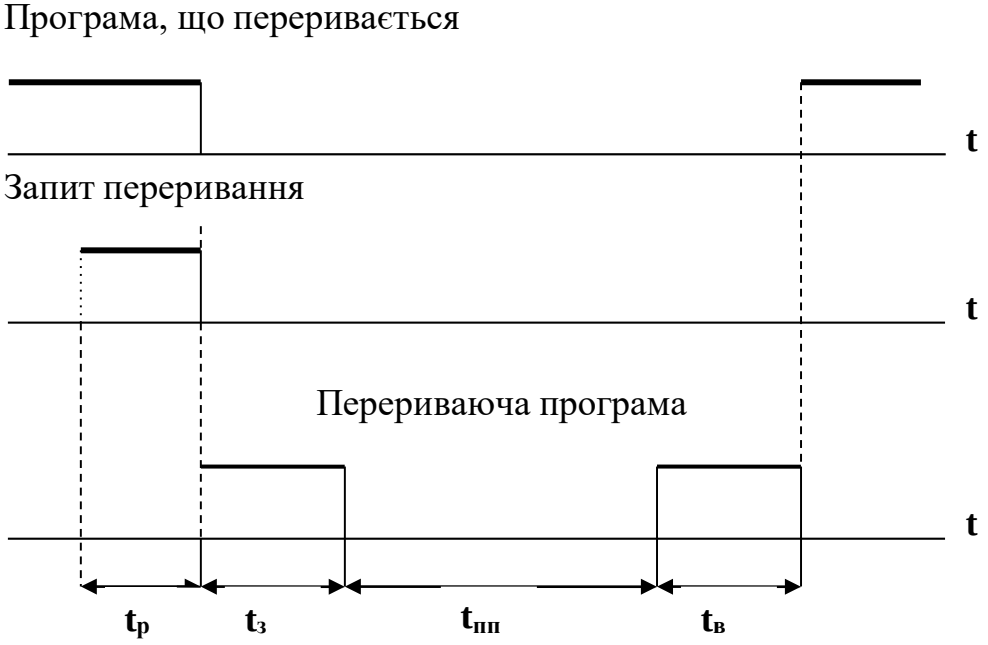


Рисунок 1.16 – Спрощена часова діаграма процесу переривання

Якщо після переходу до перериваючої програми і аж до її закінчення прийом інших запитів забороняється, то говорять, що система має глибину переривання, яка дорівнює 1. Глибина переривання дорівнює n, якщо допускається послідовне переривання до n програм. Таким чином, глибина переривання характеризується максимальною кількістю програм, які можуть перервати один одного. Глибина переривання звичайно еквівалентна кількості рівнів пріоритету в системі переривання. Системи з більшим значенням глибини переривання забезпечують більш швидку реакцію на екстрені запити.

Якщо запит не обслуговують до моменту надходження нового запиту від того ж джерела, то виникає насичення системи переривання. У такому випадку попередній запит переривання від зазначеного джерела буде загублений, що неприпустимо. Тому при конструюванні МПС швидкодія, характеристики системи переривання, кількість джерел переривання й частота виникнення запитів мають бути взаємоузгоджені таким чином, що б насичення системи було неможливе.

Формування сигналів переривань – запитів ЗП на обслуговування відбувається в контролерах відповідних ЗП. У найпростіших випадках як сигнал переривання може використовуватися сигнал «Готовність ЗП», що

надходить із контролера ЗП у системний інтерфейс МПС. Однак таке просте рішення має істотний недолік – процесор не має можливості керувати перериваннями, тобто дозволяти або забороняти їх для окремих ЗП. У результаті організація обміну даними в режимі переривання з декількома ЗП істотно ускладнюється.

Для рішення цієї проблеми регістр стану й управління контролера ЗП доповнюють ще одним розрядом – «Дозвіл переривання». Запис 1 або 0 у розряд «Дозвіл переривання» виробляється програмним шляхом за однією з ліній шини даних системного інтерфейсу. Керуючий сигнал системного інтерфейсу «Запит на переривання» формується за допомогою схеми збігу тільки при наявності одиниць у розрядах «Готовність ЗП» і «Дозвіл переривання» регістра стану й управління контролера.

Аналогічним шляхом вирішується проблема керування перериваннями в МПС у цілому. Для цього в регістрі стану процесора виділяється розряд, уміст якого визначає дозволені або заборонені переривання від зовнішніх пристроїв. Значення цього розряду може встановлюватися програмним шляхом.

У МПС звичайно використовується однорівнева система переривань, тобто сигнал «Запит на переривання» від всіх ЗП надходить на один вхід процесора. Тому виникає проблема ідентифікації ЗП, що запросили обслуговування, і реалізації заданої черговості (пріоритету) обслуговування ЗП при одночасному надходженні декількох сигналів переривання. Існують два основних способи ідентифікації ЗП, що запросили обслуговування:

- програмне опитування регістрів стану (розряд "Готовність ЗП") контролерів усіх ЗП;
- використання векторів переривання.

Організація переривань із програмним опитуванням готовності припускає наявність у пам'яті МПС єдиної підпрограми обслуговування переривань від всіх зовнішніх пристроїв. Структура такої підпрограми наведена на рисунку 1.17.

Обслуговування ЗП за допомогою єдиної підпрограми обробки переривань здійснюється в такий спосіб. Наприкінці останнього машинного циклу виконання чергової команди основної програми процесор перевіряє наявність вимоги переривання від ЗП. Якщо сигнал переривання є й у процесорі переривання дозволений, то процесор перемикається на виконання підпрограми обробки переривань.

Після збереження вмісту регістрів процесора, використовуваних у підпрограмі, починається послідовне опитування регістрів стану контролерів усіх ЗП, що працюють у режимі переривання. Як тільки підпрограма виявить готовий до обміну ЗП, відразу виконуються дії щодо його обслуговування. Завершується підпрограма обробки переривання після опитування готовності всіх ЗП й відновлення вмісту регістрів процесора.

Пріоритет ЗП в МПС із програмним опитуванням готовності зовнішнього пристрою однозначно визначається порядком їхнього опитування в підпрограмі обробки переривань. Чим раніше в підпрограмі опитується готовність ЗП, тим менше час реакції на його запит і вище пріоритет. Необхідність перевірки готовності всіх зовнішніх пристроїв істотно збільшує час обслуговування тих ЗП, які опитуються останніми. Це є основним недоліком розглянутого способу організації переривань. Тому обслуговування переривань із опитуванням готовності ЗП використовується тільки в тих випадках, коли відсутні тверді вимоги на час обробки сигналів переривання зовнішніх пристроїв.

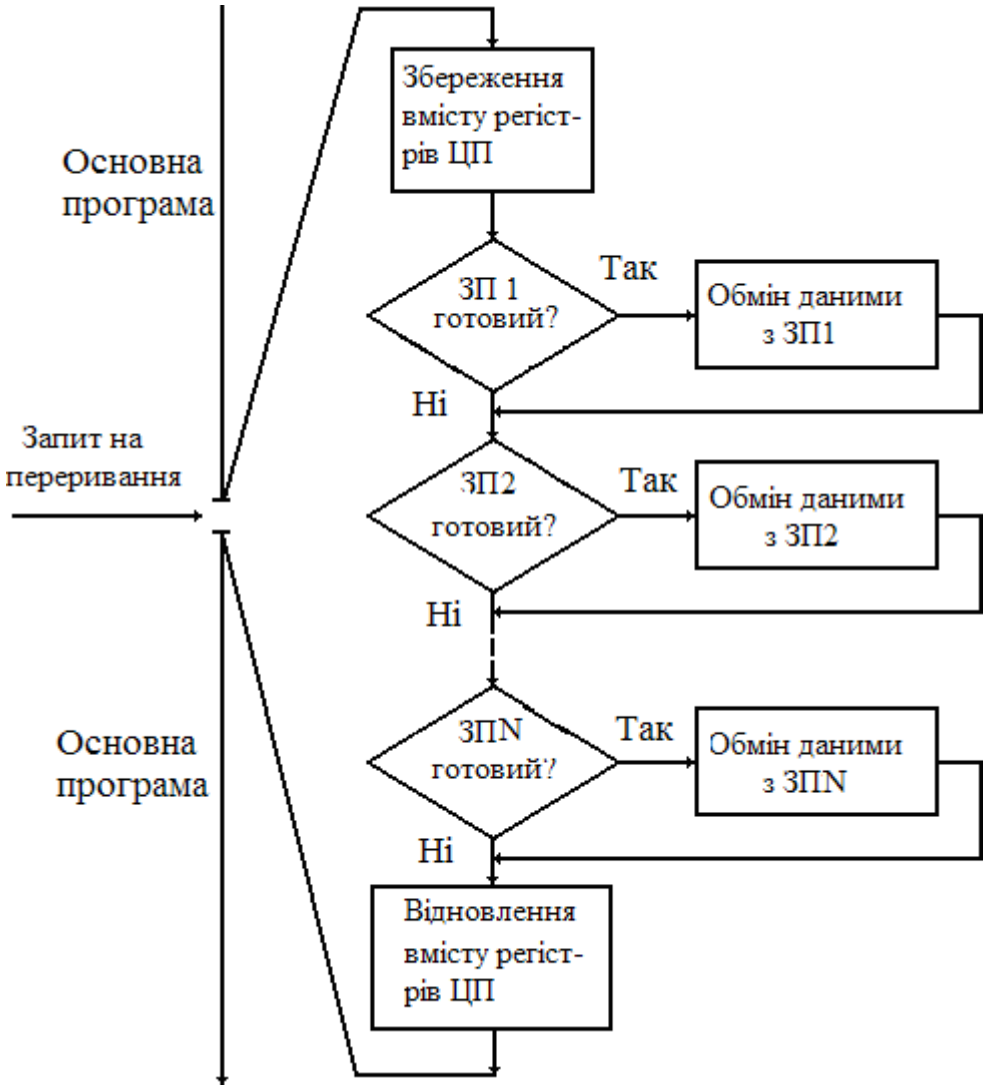


Рисунок 1.17 – Структура єдиної підпрограми обробки переривань і її зв'язок з основною програмою

Організація системи переривань у МПС із використанням *векторів переривань* дозволяє усунути зазначений недолік. При такій організації системи переривань ЗП, що запросив обслуговування, сам ідентифікує себе за допомогою вектора переривання – адреси комірки основної пам'яті

МПС, у якій зберігається або перша команда підпрограми обслуговування переривання даного ЗП, або адреса початку такої підпрограми. Таким чином, процесор, одержавши вектор переривання, відразу перемикається на виконання необхідної підпрограми обробки переривання. У МПС із векторною системою переривання кожний ЗП повинен мати власну підпрограму обробки переривання.

Розрізняють векторні системи з інтерфейсним і позаінтерфейсним вектором. У першому випадку вектор переривання формує контролер ЗП, що запросив обслуговування, у другому – контролер переривань, загальний для всіх пристроїв, що працюють у режимі переривань (IBM-сумісні персональні комп'ютери).

Розглянемо організацію векторної системи з інтерфейсним вектором. Вектор переривання видається контролером не одночасно із запитом на переривання, а тільки з дозволу процесора, як це реалізовано в схемі на рисунку 1.18. Це робиться для того, щоб виключити одночасну видачу векторів переривання від декількох ЗП. У відповідь на сигнал контролера ЗП «Запит на переривання» процесор формує керуючий сигнал «Надання переривання», що дозволяє контролеру ЗП, який запросив обслуговування, видачу вектора переривання на шину адреси системного інтерфейсу. Для цього в контролері використовуються регістр вектора переривання й схема збігу. Регістр вектора переривання звичайно реалізується за допомогою перемичок або перемикачів, що дозволяє користувачеві встановлювати для конкретних ВУ необхідні значення векторів переривання.

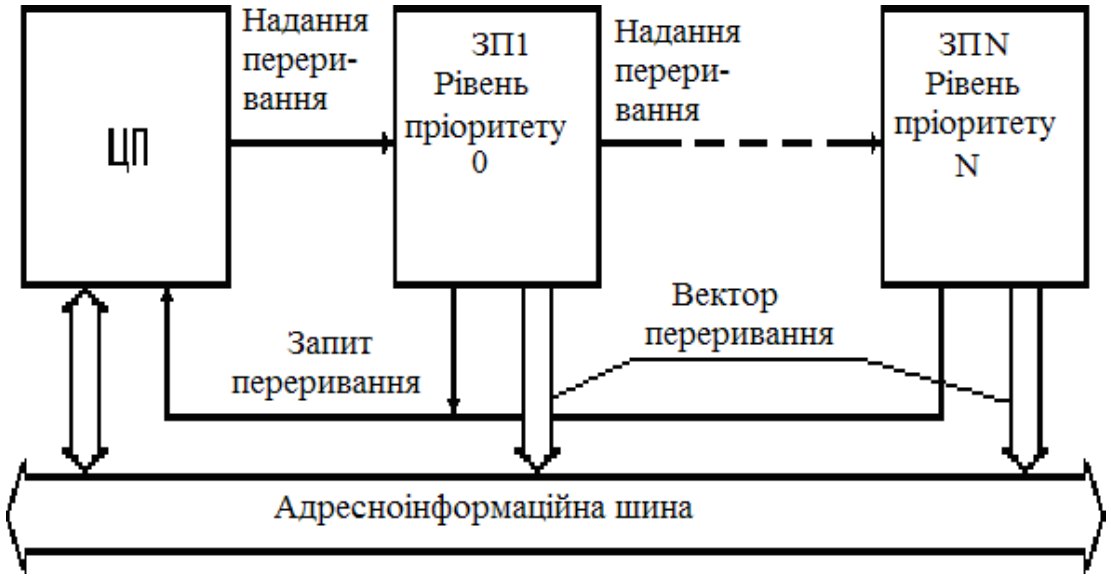


Рисунок 1.18 – Реалізація пріоритетів ЗП в МПС із векторною системою переривань, з інтерфейсним вектором

Процесор при надходженні в нього загальною лінією системного інтерфейсу «Запит переривання» формує керуючий сигнал «Надання переривання», що надходить спочатку в контролер ЗП з найвищим пріоритетом. Якщо цей пристрій не вимагав обслуговування, то його

контролер пропускає сигнал «Надання переривання» на наступний контролер, інакше подальше поширення сигналу припиняється й контролер видає вектор переривання на адресоінформаційну шину.

Апаратне опитування готовності ЗП виробляється набагато швидше, ніж програмне. Але якщо обслуговування запросили одночасно два або більше ЗП, обслуговування менш пріоритетних ЗП буде відкладено на час обслуговування більше пріоритетних, як і в системі переривання із програмним опитуванням.

Розглянута векторна система переривань практично повністю відповідає системі переривань, реалізованої в мікроЕОМ «Електроніка-60».

Векторна система із позаінтерфейсним вектором переривання використовується в IBM-сумісних персональних комп'ютерах. У цих комп'ютерах контролери зовнішніх пристроїв не мають регістрів для зберігання векторів переривання, а для ідентифікації пристроїв, що запросили обслуговування, використовується загальний для всіх ЗП контролер переривань. БІС програмувального контролера переривань (ПКП) являє собою пристрій, що реалізує до восьми рівнів запитів на переривання з можливістю програмного маскування й зміни порядку обслуговування переривань.

За рахунок каскадного включення БІС ПКП кількість рівнів переривання може бути поширена до 64. Структурна схема ПКП наведена на рисунку 1.19.

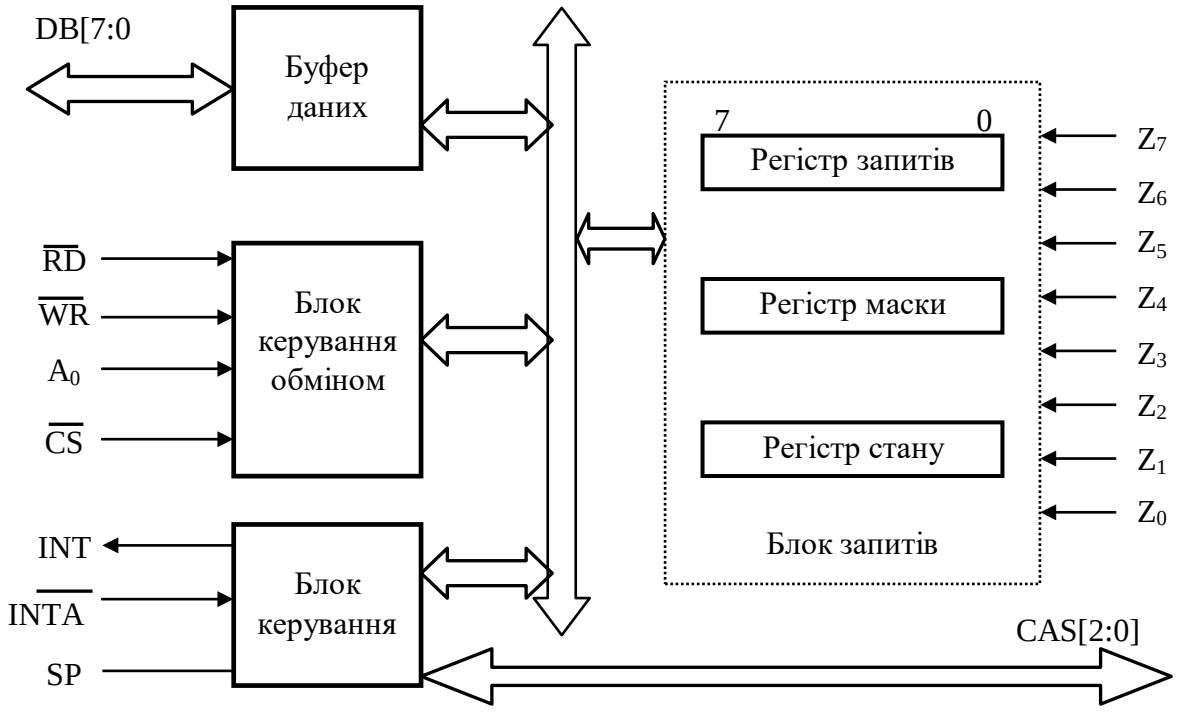


Рисунок 1.19 – Структура контролера переривань

Програмувальний контролер переривань дозволяє організувати більш гнучку й ефективну підсистему переривань у МПС, він генерує код

команди CALL, тому область векторів переривань може розташовуватися за довільними адресами пам'яті.

Програмувальний контролер переривань містить у собі блоки зв'язку із системною шиною, керування й запитів. Блок запитів містить три 8-розрядних регістри: регістр запитів , у якому фіксуються запити від джерел переривань, регістр маски, що визначає підмножину джерел, яким дозволені переривання, й регістр стану, у якому фіксуються запити, прийняті на обслуговування.

Склад керуючих ліній контролера включає стандартні лінії підключення до системної шини: D[7:0], A0, WR\, RD\, CS\; лінії, що передають запит на переривання процесору й відповідь МП INT і INTA\ відповідно.

Лінії запитів z7..z0 з'єднують контролер із джерелами переривань. Програмування контролера здійснюється шляхом завантаження в спеціальні регістри двох або трьох керуючих слів.

Прямий доступ до пам'яті. Одним зі способів обміну даними з ЗП є обмін у режимі прямого доступу до пам'яті (ПДП). У цьому режимі обмін даними між ЗП й основною пам'яттю МПС відбувається без участі процесора. Обміном у режимі ПДП керує не програма, виконувана процесором, а електронні схеми, зовнішні стосовно процесора. Звичайно схеми, що керують обміном у режимі ПДП, розміщаються в спеціальному контролері, що називається контролером прямого доступу до пам'яті.

Обмін даними в режимі ПДП дозволяє використовувати в МПС швидкодіючі зовнішні запам'ятовувальні пристрої, такі, наприклад, як накопичувачі на твердих магнітних дисках, оскільки ПДП може забезпечити час обміну одним байтом даних між пам'яттю й ЗП, рівний циклу звертання до пам'яті.

Для реалізації режиму прямого доступу до пам'яті необхідно забезпечити безпосередній зв'язок контролера ПДП і пам'яті МПС. Для цієї мети можна було б використати спеціально виділені шини адреси й даних, що зв'язують контролер ПДП із основною пам'яттю. Але таке рішення не можна визнати оптимальним, тому що це призведе до значного ускладнення МПС у цілому, особливо при підключенні декількох ЗП. З метою скорочення кількості ліній у шинах МПС контролер ПДП підключається до пам'яті за допомогою шин адреси й даних системного інтерфейсу (рис. 1.20). При цьому виникає проблема спільного використання шин системного інтерфейсу процесором і контролером ПДП. Можна виділити два основних способи її рішення: реалізація обміну в режимі ПДП із "захопленням циклу" і режим ПДП із блокуванням процесора.

Існують два різновиди прямого доступу до пам'яті з «захопленням циклу». Найбільш простий спосіб організації ПДП полягає в тому, що для обміну використовуються ті машинні цикли процесора, у яких він не обмінюється даними з пам'яттю. У такі цикли контролер ПДП може обмінюватися даними з пам'яттю, не заважаючи роботі процесора. Однак

виникає необхідність виділення таких циклів, щоб не відбулося тимчасового перекриття обміну ПДП із операціями обміну, які виконує процесор. У деяких процесорах формується спеціальний керуючий сигнал, що вказує цикли, у яких процесор не звертається до системного інтерфейсу. При використанні інших процесорів для виділення таких циклів необхідне застосування в контролерах ПДП спеціальних селекторних схем, що ускладнює їхню конструкцію. Застосування розглянутого способу організації ПДП не знижує продуктивності МПС, але при цьому обмін у режимі ПДП можливий тільки у випадкові моменти часу окремими байтами або словами.

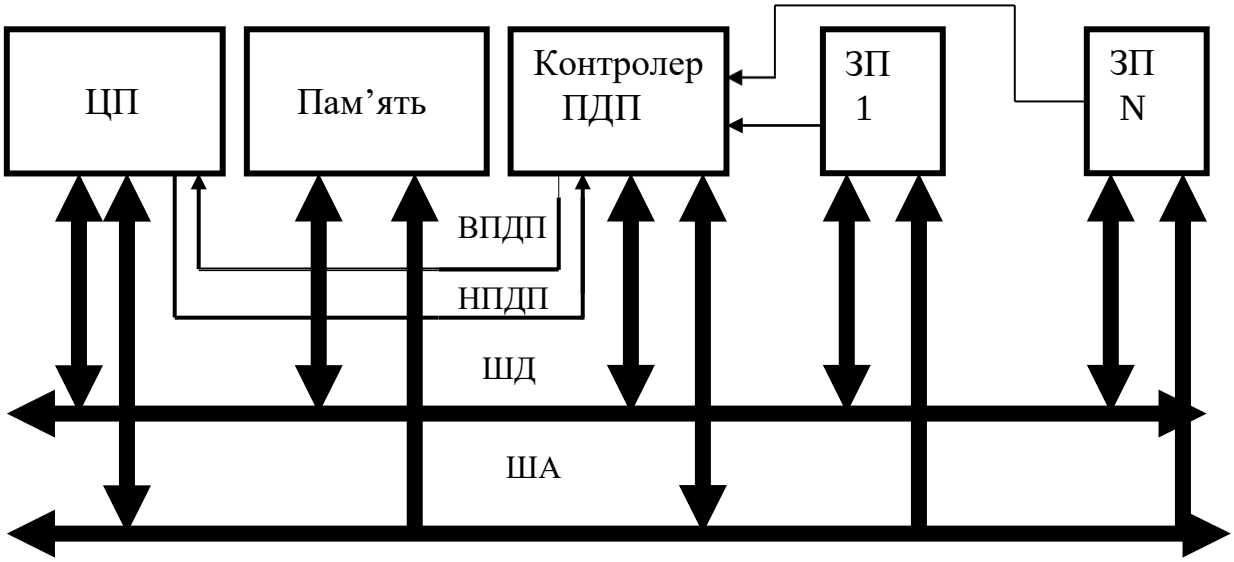


Рисунок 1.20 – Підключення контролера ПДП до системних шин

Більш розповсюдженим є ПДП із «захопленням циклу» і примусовим відключенням процесора від шин системного інтерфейсу. Для реалізації такого режиму ПДП системний інтерфейс микроЕВМ доповнюється двома лініями для передачі керуючих сигналів: «Вимогою прямого доступу до пам'яті» (ВПДП) і «Наданням прямого доступу до пам'яті» (НПДП).

Керуючий сигнал ВПДП формується контролером прямого доступу до пам'яті. Процесор, одержавши цей сигнал, припиняє виконання чергової команди, не чекаючи її завершення, видає на системний інтерфейс керуючий сигнал НПДП і відключається від шин системного інтерфейсу. Від цього моменту всі шини системного інтерфейсу керуються контролером ПДП. Контролер ПДП, використовуючи шини системного інтерфейсу, здійснює обмін одним байтом або словом даних з пам'яттю МПС і потім, знявши сигнал ВПДП, повертає керування системним інтерфейсом процесору. Як тільки контролер ПДП буде готовий до обміну наступним байтом, він знову «захоплює» цикл процесора й т.д. У проміжках між сигналами ВПДП процесор продовжує виконувати команди програми. Тим самим виконання програми вповільнюється, але в меншому ступені, ніж при обміні в режимі переривань.

Застосування в МПС обміну даними з ЗП у режимі ПДП завжди вимагає попередньої підготовки, а саме: для кожного ЗП необхідно виділити область пам'яті, використовувану при обміні, і вказати її розмір, тобто кількість записуваних у пам'ять або зчитуваних із пам'яті байт інформації. Отже, контролер ПДП повинен обов'язково мати у своєму складі регістр адреси й лічильник байт (слів). Перед початком обміну з ЗП в режимі ПДП процесор повинен виконати програму завантаження. Ця програма забезпечує запис у зазначені регістри контролера ПДП початкової адреси виділеної ЗП пам'яті і її розміру в байтах або словах залежно від того, якими порціями інформації ведеться обмін. Сказане не ставиться до початкового завантаження у пам'ять програм у режимі ПДП. У цьому випадку вміст регістра адреси й лічильника байт слів установлюється перемикачами або перемичками безпосередньо на платі контролера.

Блок-схема простого контролера ПДП, що забезпечує уведення даних у пам'ять МПС із ініціативи ЗП у режимі ПДП «Захоплення циклу», наведена на рисунку 1.21.

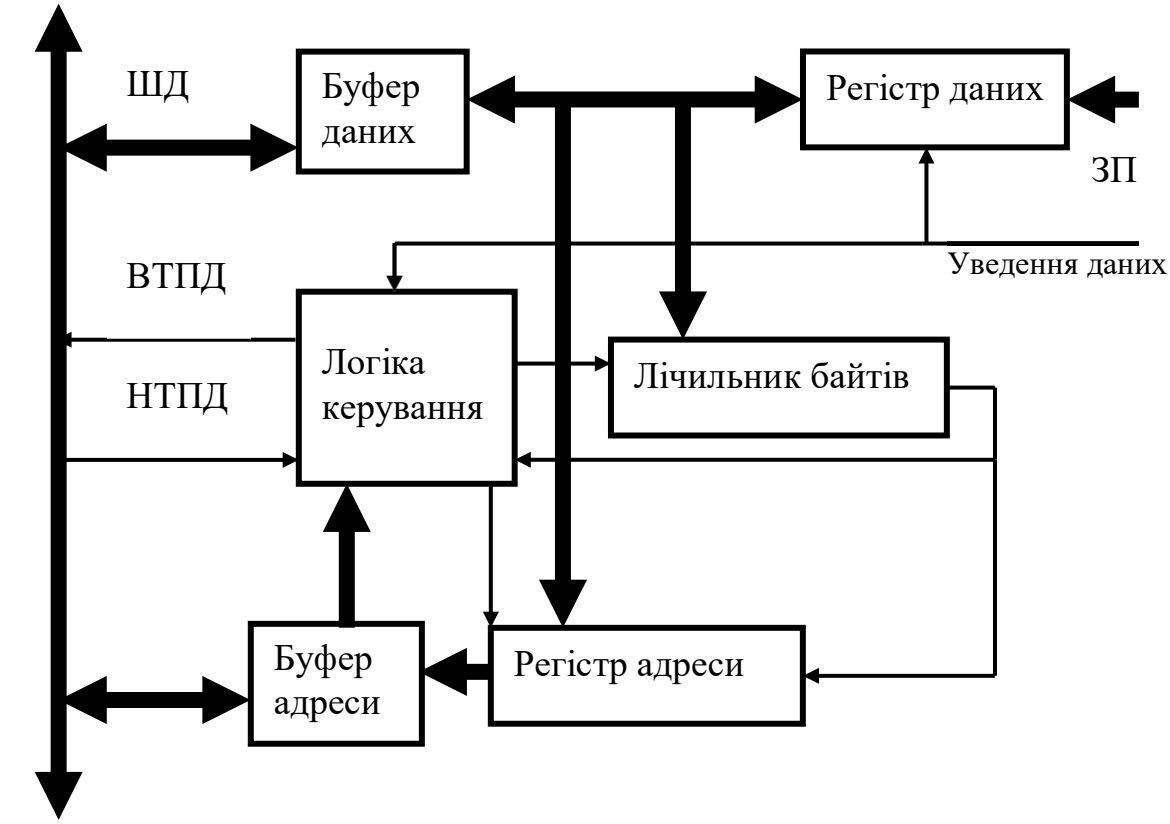


Рисунок 1.21 – Контролер ПДП для уведення даних з ЗП в режимі «Захоплення циклу» і відключенням процесора від шин системного інтерфейсу

Перед початком чергового сеансу уведення даних з ЗП процесор завантажує в регістри його контролера наступну інформацію: у лічильник байтів – кількість прийнятих байтів даних, а в регістр адреси – початкову

адресу області пам'яті для даних, що вводяться. Тим самим контролер підготовляється до виконання операції уведення даних з ЗП в пам’ять МПС у режимі ПДП.

Байти даних з ЗП надходять у регістр даних контролера в постійному темпі. При цьому кожний байт супроводжується керуючим сигналом з ЗП «Уведення даних», що забезпечує запис байта даних у регістр даних контролера. За цим сигналом й при ненульовому стані лічильника байт контролер формує сигнал ВПДП. За відповідним сигналом процесора НПДП контролер виставляє на шини адреси й даних системного інтерфейсу вміст своїх регістрів адреси й даних відповідно. Формуючи керуючий сигнал «Вивід», контролер ПДП забезпечує запис байта даних зі свого регістра даних у пам’ять МПС. Сигнал НПДП використовується в контролері й для модифікації лічильника байтів і регістра адреси. За кожним сигналом НПДП із умісту лічильника байтів віднімається одиниця, і як тільки вміст лічильника стане дорівнюватись нулю, контролер припинить формування сигналу «Вимога прямого доступу до пам'яті».

На прикладі простого контролера ПДП ми розглянули тільки процес підготовки контролера й безпосередньо передачу даних у режимі ПДП. На практиці будь-який сеанс обміну даними з ЗП в режимі ПДП завжди ініціюється програмою, виконуваною процесором, і включає два наступні етапи.

1 На етапі підготовки ЗП до чергового сеансу обміну процесор у режимі програмно-керованого обміну опитує стан ЗП (перевіряє його готовність до обміну) і посилає у ЗПУ команди, що забезпечують підготовку ЗП до обміну. Така підготовка може зводитися, наприклад, до переміщення головок на необхідну доріжку в накопичувачі на жорсткому диску. Потім виконується завантаження регістрів контролера ПДП. На цьому підготовка до обміну в режимі ПДП завершується й процесор перемикається на виконання іншої програми.

2 Обмін даними в режимі ПДП починається після завершення підготовчих операцій у ЗП з ініціативи або ЗП, як це було розглянуто вище, або процесора. У цьому випадку контролер ПДП необхідно доповнити регістром стану й керування, уміст якого буде визначати режим роботи контролера ПДП. Один з розрядів цього регістра буде ініціювати обмін даними з ЗП. Завантаження інформації в регістр стану й управління контролера ПДП відбувається програмним шляхом.

Найпоширенішим є обмін у режимі ПДП *із блокуванням процесора*. Він відрізняється від ПДП із «захопленням циклу» тим, що керування системним інтерфейсом передається контролеру ПДП не на час обміну одним байтом, а на час обміну блоком даних. Такий режим ПДП використовується в тих випадках, коли час обміну одним байтом з ЗП дорівнюється циклу системної шини.

У МПС можна використовувати декілька ЗП, що працюють у режимі ПДП. Надання таким ЗП шин системного інтерфейсу для обміну даними відбувається на пріоритетній основі. Пріоритети ЗП реалізуються так само, як і при обміні даними в режимі переривання, але замість керуючих

сигналів «Вимога переривання» і «Надання переривання» використовуються сигнали «Вимога прямого доступу» і «Надання прямого доступу», відповідно.

2 МІКРОКОНТРОЛЕРИ

Мікроконтролер – обчислювально-керуючий пристрій, призначений для виконання функцій контролю й керування периферійним устаткуванням.

Ухил убік керування накладає відбиток на особливість архітектури мікроконтролерів. Основною із цих особливостей є те, що поряд із процесорним ядром мікроконтролери мають у своєму складі підсистему введення/виводу й, можливо, підсистему пам'яті. В останньому випадку прийнято говорити про однокристальну мікро-ЕОМ.

Засоби обчислювальної техніки, використовувані для цілей автоматизації керування, розвиваються за двома взаємодоповнюючими напрямами:

1 Удосконалювання архітектури (внутрішньої організації), що дозволяє на кожному новому витку складності алгоритмів керування забезпечити адекватну продуктивність обчислювальних засобів.

2 Зниження рівня енергоспоживання й підвищення рівня надійності, які роблять функціонально зроблені засоби керування не тільки технічно реалізованими, але й доцільними зі споживчої й економічної точки зору.

Аналізуючи шляхи вдосконалювання мікропроцесорних засобів керування, будь то мікропроцесорна елементна база для вбудованих систем (спочатку мікропроцесорні комплекти ІС, а потім однокристальні мікроконтролери) або засоби промислової автоматизації - програмувальні логічні контролери (ПЛК/PLC) і універсальні мікропроцесорні комплекси, - можна помітити, що розвиток їхньої архітектури подібний спіралі із двома яскраво вираженими полюсами на кожному витку:

- На першій стадії (один полюс) продуктивність процесорного ядра не тільки достатня, але навіть перевищує вимоги алгоритмів керування, для яких вона призначена. Тому багато завдань можуть бути вирішені суто програмними засобами. Структура периферійних пристроїв на цьому етапі ще далека від оптимальної.

- На наступному етапі (другий полюс) збережена продуктивність процесорного ядра стає мінімально достатньою. Удосконалювання структури периферійних модулів дозволяє розвантажити процесорне ядро від невластивих йому операцій.

- Коли всі можливості щодо оптимізації структури периферії вичерпуються, починається новий виток розвитку, відзначений стрибкоподібним збільшенням продуктивності процесорного ядра.

Так було при переході від 8-розрядних мікроконтролерів до 16- і 32-розрядних: збільшення розрядності оброблюваного слова навіть при незмінній частоті тактування істотно збільшує продуктивність. Потім з'явилися процесори цифрової обробки сигналів (DSP). І щоразу новий

рівень продуктивності процесорного ядра супроводжувався вдосконалюванням структури периферійних пристроїв.

Однак збільшення продуктивності центрального процесора не обов'язково пов'язане зі збільшенням розрядності оброблюваного слова. Це повною мірою справедливо для алгоритмів з великим обсягом обчислювальних операцій. А для алгоритмів з перевагою логічних операцій збільшення розрядності оброблюваного слова практично не позначається на продуктивності. Ця обставина є головною причиною життєздатності 8-розрядних мікропроцесорів.

Елементна база 8-розрядних мікропроцесорів (МП) не стоїть на місці. Вона вже зробила два витки у своєму розвитку. Перший з них припадає на МП комплекти ІС і перші однокристальні мікроконтролери (МК). Другий, відзначеним підвищенням практично на порядок продуктивності 8-розрядного процесорного ядра й перетворенням в автономні модулі периферійних пристроїв, завершується в цей час. І вже з'явилися перші ластівки наступного витка спірали. Це RISC МК Scenix із частотою тактирування до 50 МГц. Поки ці МК не мають убудованих периферійних модулів, усі функції периферії емулюються програмними засобами. Чи надовго досягнута швидкодія буде досить великою, щоб обходитися без апаратних засобів?

Які відмітні ознаки сучасної елементної бази 8-розрядних мікропроцесорів?

1 Завершився перехід від МП-системи, виконаної на основі декількох великих інтегральних схем, до *однокристальних МК,* які об'єднали в одному кристалі всі основні елементи МП-системи керування: центральний процесор, постійний запам'ятовувальний пристрій (ПЗП), оперативний запам'ятовувальний пристрій (ОЗП), порти введення/виводу, таймери.

2 Відбувся перехід до *закритої архітектури МК,* що характеризується відсутністю ліній магістралей адреси й даних на виводах корпуса МК. МК являє собою закінчену систему обробки даних, нарощування пам'яті або периферійних пристроїв з використанням паралельних магістралей адреси й даних не передбачається.

3 Нові сімейства 8-розрядних МК будуються на основі принципу *модульної організації,* при якій на базі одного процесорного ядра (центрального процесора) проектується ряд МК, що розрізняються обсягом і типом пам'яті програм, обсягом пам'яті даних, набором периферійних модулів, частотою синхронізації. У результаті перехід від однієї модифікації МК до іншої не супроводжується великими матеріальними витратами на освоєння нової елементної бази як на рівні виробника, так і на рівні розроблювача систем. Усі окремі модулі МК (процесор, ПЗП, ОЗП, периферійні модулі) на рівні топології виконуються геометрично незалежними із приєднанням до внутрішніх магістралей адреси й даних. Це дозволяє значно прискорити проектування топології МК шляхом об'єднання в одному кристалі вже відпрацьованих топологічних блоків.

4 Відбулося виділення *типових функціональних периферійних модулів* МК. Серед таких модулів варто назвати таймери, процесори подій, контролери послідовних інтерфейсів, аналого-цифрові перетворювачі.

Істотно поширилася кількість режимів роботи модулів, які задаються в процесі ініціалізації *регістрів спеціальних функцій* периферійних модулів. Алгоритми роботи однойменних модулів у МК різних виробників мають незначні відмінності. Конкуренція йде на рівні деталізації режимів роботи й параметрів цих модулів.

Достаток пропозицій на ринку 8-розрядних МК є потужним стимулом до їхнього постійного вдосконалювання. Підвищення продуктивності при усвідомленій необхідності залишатися в рамках 8-розрядного ядра ведеться двома напрямами:

1 Удосконалювання архітектури центрального процесора.

2 Підвищення частоти тактування.

Однак висока продуктивність не є визначальним чинником широкого поширення 8-розрядних МК. З погляду масового використання в цей час більш важливими є наступні напрями розвитку:

- *Розмаїтість структурної організації.* Дозволяє розроблювачеві для кожного завдання знайти МК практично без надлишкових ресурсів архітектури, що спричиняє низьку вартість кінцевого виробу. Модульний принцип побудови МК – шлях до рішення цього завдання.
- *Удосконалювання технічних характеристик периферійних модулів.* Дозволяє звести до мінімуму кількість периферійних ІС на платі МП-контролера: один зі шляхів мініатюризації вбудованих МП-систем.
- *Сполучення з периферійними ІС високошвидкісним послідовним інтерфейсом***.** Забезпечує мінімізацію площі провідників на друкованій платі. Ще один шлях до мініатюризації вбудованих МП-систем.
- *Мінімізація енергії споживання.* Дозволяє зменшити розміри корпуса МК і габаритні розміри джерела живлення. Третій шлях мікромініатюризації.
- *Розширення діапазону напруги живлення.* Одночасно з мінімізацією енергії споживання дозволяє перевести системи із МК на довгострокове живлення від автономних джерел (батарейок і акумуляторів), що дозволяє їх вбудовувати у переносні вироби.
- *Перехід до нових технологій пам'яті програм.* Дрібносерійне виробництво змушує відмовитися від МК із масочним ПЗП. Збереження низької вартості елементної бази МК диктує необхідність переходу на нові технології створення резидентної пам'яті програм.
- *Підвищення надійності.* Здатність відновлення нормального функціонування програмного забезпечення при його порушеннях через електромагнітні перешкоди й при короткочасних провалах напруги живлення відкриває для МК нові області застосування.
- *Зниження вартості процесу налагодження.* Сприяє розширенню кола розроблювачів найпростіших МП-систем. Впливає на вартість кінцевого виробу.
- *Підвищення технологічності занесення програми до пам'яті МК.* Підвищує надійність збереження програми в пам'яті МК. Перехід до технології програмування в пристрої дозволяє відмовитися від розміщення

МК у спеціальних контактних панельках. Останні займають додаткову площу на платі й до того ж мають чималу вартість.

2.1 Структура МК

МК являють собою закінчену МП-систему обробки інформації, що реалізована у вигляді однієї великої інтегральної мікросхеми. МК поєднує в межах одного напівпровідникового кристала основні функціональні блоки МП керуючої системи: центральний процесор, постійний запам'ятовувальний пристрій (ПЗП), оперативний запам'ятовувальний пристрій (ОЗП), периферійні устрої для уведення й виводу інформації.

Широке розмаїття моделей МК, можливість розробки й виробництва нових моделей у короткий термін забезпечує *модульний принцип побудови* МК, що взятий на озброєння всіма провідними компаніями. При модульному принципі побудови всі МК одного сімейства містять у собі базовий функціональний блок, який однаковий для всіх МК сімейства, і змінюваний функціональний блок, який відрізняє МК різних моделей у межах одного сімейства (рис. 2.1).

Базовий функціональний блок включає:

- Центральний процесор.
- Внутрішні магістралі адреси, даних і керування.
- Схему формування багатофазної імпульсної послідовності для тактування центрального процесора й міжмодульних магістралей.
- Устрій керування режимами роботи МК, такими, як активний режим, у якому МК виконує прикладну програму, режими зниженого енергоспоживання, в один із яких МК переходить, якщо за умовами роботи виконання програми може бути припинено, стан початкового запуску (скидання) і переривання.

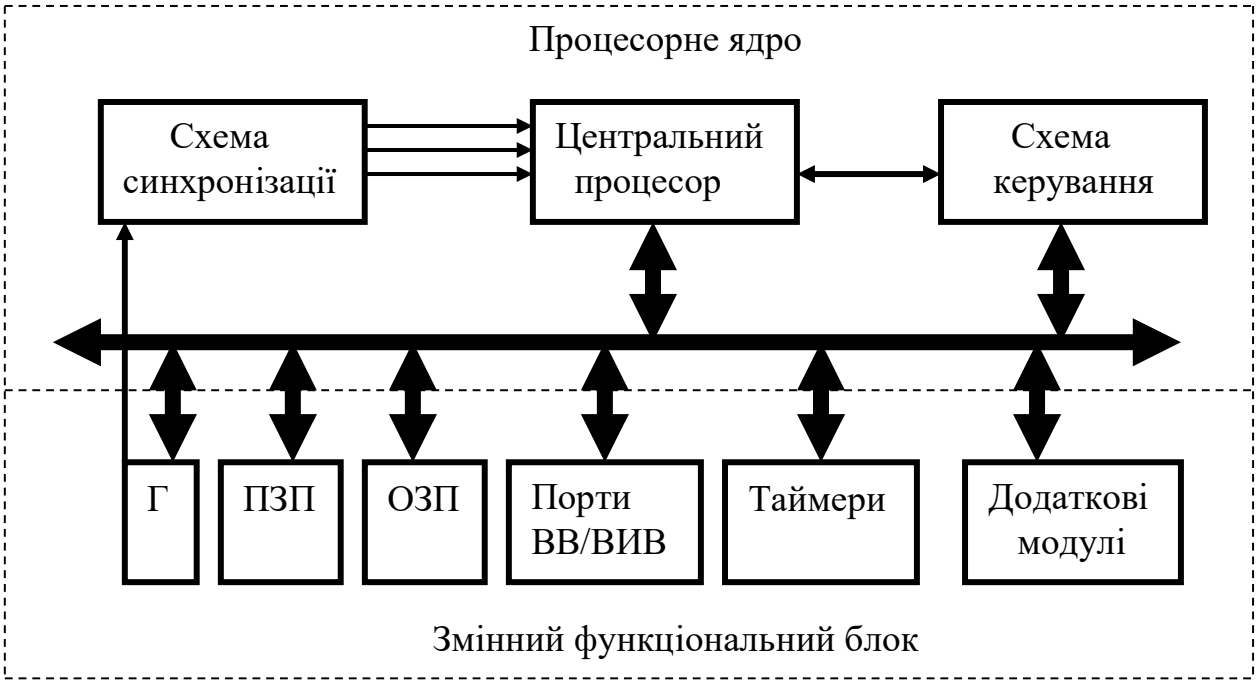


Рисунок 2.1 – Модульна будова МК

Базовий функціональний блок заведено називати *процесорним ядром* МК. Процесорне ядро позначають ім'ям сімейства МК, основою якого воно є. Наприклад, ядро HC05 – процесорне ядро сімейства Motorola MC68HC05, ядро MCS-51 – ядро сімейства МК Intel 8xc51, ядро PIC16 – процесорне ядро Microchip PIC16.

Змінюваний функціональний блок включає модулі різних типів пам'яті, модулі периферійних устроїв, модулі генераторів синхронізації й деяких додаткових модулів спеціальних режимів роботи МК. Кожний модуль має виводи для підключення його до внутрішніх магістралей МК. Це дозволяє на рівні топологічного проектування «приєднувати» ті або інші модулі до магістралей процесорного ядра, створюючи в такий спосіб різноманітні за структурою МК у межах одного сімейства. Сукупність модулів, які розроблені для певного процесорного ядра, заведено називати бібліотекою периферійних модулів. Термін «бібліотека периферійних модулів» недостатньо точно відображає сучасні тенденції структурної організації МК. Якщо раніше під довільно поєднуваними до складу МК модулями малися на увазі тільки модулі периферійних устроїв, то тепер вибирати можна в кожній з п'яти функціональних груп:

- Модулі пам'яті.
- Модулі убудованих генераторів синхронізації.
- Модулі периферійних устроїв.
- Модулі контролю за напругою живлення й ходом виконання програми.
- Модулі внутрішньосхемного налагодження й програмування.

Термін «модуль пам'яті» у застосуванні до МК став використатися на етапі переходу до нових технологій створення резидентної пам'яті програм і даних. Енергонезалежна пам'ять типу FLASH і EEPROM має не тільки режими зберігання й читання інформації, що була в неї записана до початку експлуатації виробу на етапі програмування, але й режими стирання й програмування під керуванням прикладної програми. Унаслідок цього енергонезалежна пам'ять типу FLASH і EEPROM вимагає керування режимами роботи, для чого обладнана додатковими блоками керування й регістрами спеціальних функцій. Масив комірок пам'яті, доступних для читання, стирання й запису інформації, додаткові аналогові й цифрові схеми керування, а також регістри спеціальних функцій об'єднані у функціональний блок, що і зветься модулем пам'яті.

Істотної зміни зазнали також генератори синхронізації МК. По-перше, відбувся функціональний поділ саме генератора синхронізації, що виділився в окремий модуль, і схеми формування багатофазної послідовності імпульсів для тактування центрального процесора й міжмодульних магістралей, що є невід'ємною частиною процесорного ядра. По-друге, з'явилася можливість вибору зовнішнього часозадавального елемента: кварцовий або керамічний резонатор, RC-ланцюг. Схемотехника виконання підсилювачів з позитивним зворотним зв'язком визначається

типом часозадавального елемента, відповідно з'явилися різні модифікації модулів убудованого генератора синхронізації. По-третє, підвищення продуктивності процесорного ядра МК пов'язане з підвищенням частоти тактування центрального процесора й міжмодульних магістралей. Однак застосування високочастотних кварцових резонаторів у якості часозадавального елемента підвищує рівень електромагнітного випромінювання, тобто зростає інтенсивність генерації перешкод. Крім того високочастотні кварцові резонатори є відносно дорогим елементом. Тому сучасні генератори синхронізації мають у своєму складі множувач частоти з коефіцієнтом, що вибирається програмно. Множувач частоти часто виконується за схемою синтезатора з контуром фазового автопідстроювання частоти (англомовна абревіатура PLL – Phase Loop Lock). Схема синтезатора частоти й регістри спеціальних функцій для керування режимами його роботи об'єднані в модуль синхронізації.

Група модулів периферійних устроїв включає більшість із відомих типів адаптерів сполучення з об'єктом:

- Паралельні порти введення/виводу з можливістю індивідуального настроювання напряму передачі кожної лінії.
- Багаторежимні таймери/лічильники, таймери періодичних переривань, процесори подій.
- Контролери послідовного інтерфейсу зв'язку декількох типів (SCI, SPI, CAN).
- Багатоканальний АЦП.
- Контролери ЖК-індикаторів і світлодіодної матриці.

Відносно новими для 8-розрядних МК є дві останні групи модулів. Перші здійснюють діагностику деяких підсистем МК і дозволяють відновити працездатність устрою на основі МК при порушеннях програмного характеру, збоях у системі синхронізації, зниженні напруги живлення. Другі є апаратною основою режимів *внутрішньосхемного налагодження й програмування в системі***,** які дозволяють налагоджувати прикладну програму й заносити коди програми до енергонезалежної пам'яті МК прямо на платі кінцевого виробу, без використання додаткових апаратних систем налагодження й програмування.

2.2 Процесорне ядро МК

Процесорне ядро являє собою нерозривну єдність трьох складових його технічного рішення:

1 Архітектури центрального процесора із властивим їй набором регістрів для зберігання проміжних даних, організацією пам'яті й *способами адресації* операндів у просторі пам'яті, *системою команд***,** що визначає набір можливих дій над операндами, організацією процесу вибірки й виконання команд.

2 Схемотехники втілення архітектури, що визначає послідовність переміщення даних по внутрішніх магістралях МК між регістрами, арифметично-логічним устроєм і комірками пам'яті, процес виконання кожної команди.

3 Технології виробництва напівпровідникової БІС МК, що дозволяє розмістити схему тієї або іншої складності на напівпровідниковому кристалі, визначає припустиму частоту перемикань у схемі й енергію споживання.

Ці три складові нерозривно зв'язані один з одним і в остаточному підсумку визначають найважливіший параметр процесорного ядра МК – його продуктивність.

Ядро сучасних 8-розрядних МК реалізує один із двох принципів побудови МП:

- МП із CISC-архітектурою - МП із повною системою команд (Complicated Instruction Set Computer).
- МП із RISC-архітектурою - МП зі скороченою системою команд (Reduced Instruction Set Computer).

У застосуванні до 8-розрядних МК мікропроцесор з CISC-архітектурою має одно-, дво- і трибайтовий (рідко чотирибайтовий) формати команд. Вибірка команди з пам'яті здійснюється побайтно протягом декількох циклів синхронізації МК. Час виконання кожної команди з урахуванням часу вибірки становить від 1 до 10 циклів. Тривалість циклу синхронізації дорівнює періоду частоти тактування внутрішніх магістралей $f_{BUS.}$ До МК із CISC-архітектурою відносяться сімейства HC05 і HC08 фірми Motorola, МК із ядром MCS-51, що було запропоновано фірмою Intel, а в цей час підтримується відразу декількома виробниками (Atmel, Philips, Dallas), МК сімейства 3500 фірми Infineon і низка інших.

МП з RISC-архітектурою має формат команди фіксованої довжини (наприклад, 12 або 14 біт), вибірка команди з пам'яті і її виконання здійснюються за один цикл синхронізації МК. До МК із RISC-архітектурою відносяться МК AVR фірми Atmel, МК PIC фірми Microchip, МК фірми Scenix.

За визначенням МК із RISC-архітектурою повинні мати більш високу продуктивність у порівнянні з CISC МК при одній і тій самій частоті f_{BUS}, тому що перші виконують кожну команду за один такт, а останні – за декілька. Однак на практиці питання про продуктивність є значно більш складним і майже завжди неоднозначним:

- Зазначена в довідкових даних частота синхронізації звичайно відповідає частоті підключеного кварцового резонатора, f_{XCLK}, у той час як тривалість циклу центрального процесора визначається частотою обміну по внутрішніх магістралях адреси й даних f_{BUS}. Співвідношення f_{XCLK} і f_{BUS} індивідуально для кожного ядра МК. Так, для ядра Intel MCS-51 $f_{XCLK} / f_{BUS} = 12$, для ядра Motorola HC05 $f_{XCLK} / f_{BUS} = 2$, для Microchip PIC16 $f_{XCLK} / f_{BUS} = 4$, для Scenix $f_{XCLK} / f_{BUS} = 1$. У МК Motorola HC08 тактування здійснюється

з використанням множувача частоти й $f_{BUS} > f_{XCLK}$. Тому при оцінці продуктивності варто порівнювати тільки максимальну частоту тактування міжмодульних магістралей f_{BUS}.

- Продуктивність МП, і МК у тому числі, заведено оцінювати кількістю операцій пересилання «регістр-регістр», які можуть бути виконані протягом однієї секунди. Для МК із RISC-архітектурою час виконання будь-якої операції становить $1/f_{BUS,}$ отже, їхня продуктивність дорівнює f_{BUS} оп/с. Наприклад, продуктивність PIC16 становить 5 млн оп/с, Scenix – 25 млн оп/с. У МК із CISC-архітектурою кількість циклів виконання операції «регістр-регістр» становить від 1 до 3, що знижує продуктивність.
- Однак така оцінка продуктивності є загальною. Вона не враховує особливості алгоритмів керування, використовуваних у кожній конкретній області застосування. Так, при розробці швидкодіючих регуляторів основну увагу варто приділяти часу виконання операцій множення й ділення, які потрібні при реалізації рівнянь різних передатних функцій. А при реалізації кнопкової станції кабіни ліфта варто оцінювати час виконання тільки логічних функцій, які використовуються при опитуванні клавіатури й при генерації протоколу послідовного інтерфейсу зв'язку з контролером керування руху, що оптимізує переміщення між поверхами відразу декількох кабін ліфта. У завданнях оптимального керування за таблицями, які характерні для устроїв силової електроніки, на перший план виходить можливість швидкого перебору більших таблиць даних. Тому в критичних ситуаціях, пов'язаних з вимогами високої швидкодії, варто оцінювати продуктивність на основі тих операцій, які переважно використаються в алгоритмі керування й мають обмеження за часом виконання.
- У завданнях керування об'єктом у реальному часі існує ще один дуже важливий фактор продуктивності, що ніяк не відображається кількістю операцій за секунду. Це час переходу на підпрограму переривання за запитом зовнішнього устрою або периферійного модуля. У процесі переходу на підпрограму переривання кожний МК повинен розпізнати запит на переривання, дочекатися завершення виконання поточної команди, зберегти значення програмного лічильника PC і деякі регістри центрального процесора в стеці, завантажити вектор переривання, виконати деякі допоміжні команди й лише потім почати виконання алгоритму обслуговування устрою, що викликало це переривання. Сумарний час переходу на підпрограму переривання визначається архітектурою процесорного ядра МК і частотою його тактирування.

2.3 Резидентна пам'ять МК

Закрита архітектура сучасних 8-розрядних МК стала реалізованою лише за умови інтеграції на кристал МК модулів пам'яті двох типів:

енергонезалежного запам'ятовувального пристрою для зберігання кодів прикладних програм (ПЗП) і оперативного запам'ятовувального пристрою для зберігання проміжних результатів обчислень (ОЗП). З моменту появи МК технологія енергонезалежних запам'ятовувальних пристроїв зазнала безлічі змін, які дозволили не тільки підвищити інформаційну ємність, швидкодію, надійність зберігання інформації, але й викликали появу принципово нових технологій програмування резидентної пам'яті МК. З погляду користувачів МК варто розрізняти п'ять типів енергонезалежної резидентної пам'яті:

- *ПЗП масочного типу – mask-ROM.* Уміст осередків ПЗП цього типу записується на заводі-виготовлювачі МК за допомогою масок і не може бути замінений або «допрограмований» в області раніше не використаного сегмента пам'яті. Тому МК із таким типом пам'яті програм варто використати тільки після досить тривалої досвідченої експлуатації виробів. Перші зразки масочних ПЗП з'явилися на початку 60-х рр., але навіть сьогодні ПЗП масочного типу – найдешевше й ефективне рішення при великих обсягах апаратури, що випускається. Використання МК із масочним ПЗП економічно стає рентабельним при партії в кілька десятків тисяч штук. Крім сприятливих економічних аспектів, рішення із ПЗП масочного типу мають й іншу перевагу. Вони забезпечують високу надійність зберігання інформації із причини програмування в заводських умовах з наступним контролем якості. Недолік ПЗП масочного типу очевидний: будь-яка зміна прикладної програми потребує нової серії ІС, що може виявитися досить дорогим і трудомістким рішенням.
- *ПЗП одноразово програмувальні користувачем – OTPROM (One-Time Programmable ROM).* У незапрограмованому стані кожна комірка пам'яті модуля одноразово програмувального ПЗП при зчитуванні повертає код $FF. Програмуванню підлягають тільки ті розряди, які після програмування повинні містити 0. Якщо в процесі програмування деякі розряди якої-небудь комірки пам'яті були встановлені в 0, то відновити в цих розрядах одиничне значення вже неможливо. Тому розглянутий тип пам'яті й зветься «одноразово програмувальні ПЗП». Однак ті розряди, які в процесі попереднього сеансу програмування не змінювалися, тобто мають одиничні значення, можуть бути піддані програмуванню надалі й «до установлені» в 0. Кількість можливих сеансів програмування модуля одноразового програмувального ПЗП в складі МК не має обмежень. Технологія програмування складається в багаторазовому додатку імпульсів підвищеної напруги до елементарних осередків адресованого байта пам'яті (бітам), які програмуються. Рівень напруги програмування, кількість імпульсів та їхніх часових параметрів повинні в точності відповідати технічним умовам. У противному випадку комірки пам'яті можуть відновити одиничне значення по закінченні деякого часу (іноді декількох років) або при зміні умов роботи. МК із одноразово програмувальним ПЗП рекомендується використати у виробах, що випускаються невеликими партіями.

- *ППУ, програмувальні користувачем, з ультрафіолетовим стиранням – EPROM (Erasable Programmable ROM).* ПЗП даного типу допускають багаторазове програмування. Технологія програмування близька до технології одноразово програмувальних ПЗП. Перед кожним сеансом програмування для відновлення одиничного значення раніше запрограмованих комірок пам'яті весь модуль ПЗП підлягає операції стирання за допомогою ультрафіолетового опромінення. Для цього корпус МК виконаний зі спеціальним скляним вікном, усередині якого розташована пластина ІС МК. Але якщо деякі розряди комірок пам'яті повинні бути «до програмовані» з 1 на 0 при незмінному стані раніше запрограмованих розрядів, то операція стирання може бути пропущена. Кількість сеансів стирання/програмування ПЗП даного типу обмежене й становить 25...100 за умови дотримання технології програмування (напруга, кількість й тривалість імпульсів програмування) і технології стирання (хвильовий діапазон джерела ультрафіолетового випромінювання). МК із ПЗП даного типу мають високу вартість, тому їх рекомендується використати тільки в експериментальних зразках виробів.
- *ПЗУ, програмувальні користувачем, з електричним стиранням – EEPROM, або E²PROM (Electrically Erasable Programmable ROM).* ПЗП з електричним програмуванням і поєднали в собі три позитивних якості розглянутих вище типів пам'яті. По-перше, ПЗП типу EEPROM програмуються користувачем; по-друге, ці ПЗП можуть бути багаторазово піддані операції стирання й, отже, багаторазово програмуються користувачем; по-третє, ці ПЗП дешевше ПЗП з ультрафіолетовим стиранням. Максимальна кількість циклів стирання/програмування ПЗП типу EEPROM у складі МК звичайно дорівнює 10000. Для порівняння: той самий тип пам'яті в окремому корпусі допускає 10^6 циклів стирання/програмування. Технологія програмування пам'яті типу EEPROM дозволяє реалізувати побайтове стирання й побайтове програмування, для чого до обраної комірки пам'яті повинна бути прикладена відносно висока напруга 10...20 В. Однак допускається також одночасне стирання деякої кількості комірок пам'яті з послідовними адресами, тобто стирання блоку пам'яті. Незважаючи на очевидні переваги, рідкі моделі сучасних МК використовують ПЗП типу EEPROM для зберігання програм. Провиною тому дві обставини. По-перше, ПЗП типу EEPROM мають обмежену ємність і можуть використовуватися як резидентна пам'ять програм тільки в маловивідних МК із невеликим обсягом пам'яті. По-друге, майже одночасно з EEPROM ПЗП з'явилися ПЗП типу FLASH, які забезпечують близькі користувальницькі характеристики, але значно дешевші.
- *ПЗУ з електричним стиранням типу FLASH – FLASH ROM.* ПЗП типу FLASH були призначені для заповнення «ніші» між дешевими одноразово програмувальними ПЗП великої ємності й дорогими EEPROM ПЗП малої ємності. ПЗП типу FLASH зберегли переваги, властиві EEPROM: можливість багаторазового стирання й програмування за допомогою додатка підвищеної напруги. Однак для збільшення обсягу

пам'яті транзистор адресації кожного елементарного осередку був вилучений, що не дає можливості програмувати кожний біт пам'яті окремо. *Пам'ять типу FLASH стирається й програмується сторінками або блоками.* Сторінка, як правило, становить 8, 16 або 32 байта пам'яті, блоки можуть поєднувати деяку кількість сторінок, аж до повного обсягу резидентного ПЗП МК (до 60 кбайт). Спрощення декодувальних схем, що відбулося через зменшення кількості транзисторів і, як наслідок, зниження вартості й розмірів призвело до того, що МК із FLASH-пам'яттю програм у цей час стають конкурентоспроможними у відношенні не тільки до МК із однократно програмувальним ПЗП, але й з масочним ПЗУ також.

Технологія створення резидентної FLASH-пам'яті МК безупинно вдосконалюється. Одні із кращих показників досягнуті в МК сімейства HC08 фірми Motorola:

- Гарантована кількість циклів стирання/програмування становить 10^5.
- Гарантований період зберігання записаної інформації дорівнює 10 рокам, тобто становить життєвий цикл виробу.
- Модулі FLASH-пам'яті працюють і програмуються при напрузі живлення МК від 1.8 до 2.7 В.
- Еквівалентний час програмування одного байта пам'яті знижене до 60 мкс, що дозволяє виконати програмування МК із 32 кбайт пам'яті протягом 2 с.

Перспективні технології FLASH-пам'яті припускають збільшення швидкості програмування до 1 Мбит/с.

Крім ПЗУ до складу МК входить також і *статичний* оперативний запам'ятовувальний пристрій. Визначення «статичний» виділено не випадково: сучасні 8-розрядні МК допускають зниження частоти тактування до як завгодно малих значень із метою зниження енергії споживання. Уміст комірок ОЗП при цьому зберігається на відміну від динамічної пам'яті. У якості ще однієї особливості слід зазначити, що більшість МК у технічному описі має параметр «напруга зберігання інформації» – $U_{STANDBY}$. При зниженні напруги живлення нижче мінімально припустимого рівня U_{DDMIN}, але вище напруги зберігання $U_{STANDBY}$, програма керування мікроконтролером виконуватися не буде, але інформація в ОЗП збережеться. Тоді при відновленні напруги живлення можна буде виконати скидання МК і продовжити виконання програми без втрати даних. Рівень напруги зберігання становить приблизно 1 В. Це дозволяє, якщо буде потреба, перевести МК на живлення від автономного джерела (батарейки або акумулятора) і зберегти тим самим дані ОЗП. Великої витрати енергії споживання в цьому випадку не буде, тому що система тактування МК може бути відключена. Останнім часом з'явилися МК, які в корпусі мають автономне джерело живлення, що гарантує збереження даних в ОЗП протягом 10 років (МК DS5000 фірми Dallas Semiconductor).

Регістри МК. Як і всі МПС, МК мають набір регістрів, які

використовуються для керування його ресурсами. До цих регістрів входять звичайно регістри процесора (акумулятор, регістри стану, індексні регістри), регістри керування (регістри керування перериваннями, таймером), регістри, що забезпечують уведення/вивід даних (регістри даних портів, регістри керування паралельним, послідовним або аналоговим уведенням/виводом). Звертання до цих регістрів може здійснюватися по-різному.

У МК із RISC-процесором усі регістри (часто й акумулятор) розташовуються за адресами, що задають явно. Це забезпечує більш високу гнучкість при роботі процесора.

Одним з важливих питань є розміщення регістрів в адресному просторі МК. У деяких МК усі регістри й пам'ять даних розташовуються в одному адресному просторі. Це означає, що пам'ять даних сполучена з регістрами. Такий підхід називається «відображенням ресурсів МК на пам'ять».

В інших МК адресний простір пристроїв уведення/виводу відокремлено від загального простору пам'яті. Окремий простір уведення/виводу дає деяку перевагу процесорам з гарвардською архітектурою, забезпечуючи можливість зчитувати команду під час звертання до регістра уведення/виводу.

Стік МК. У мікроконтролерах ОЗП даних використовується також для організації виклику підпрограм і обробки переривань. При цих операціях уміст програмного лічильника й основних регістрів (акумулятор, регістр стану й інші) зберігається й потім відновлюється при поверненні до основної програми.

У фон-неймановській архітектурі єдина область пам'яті використовується, у тому числі, і для реалізації стека. При цьому знижується продуктивність пристрою, тому що одночасний доступ до різних видів пам'яті неможливий. Зокрема при виконанні команди виклику підпрограми наступна команда вибирається після того, як у стек буде поміщений уміст програмного лічильника.

У гарвардській архітектурі стекові операції здійснюються в спеціально виділеній для цієї мети пам'яті. Це означає, що при виконанні програми виклику підпрограм процесор з гарвардською архітектурою робить кілька дій одночасно.

Необхідно пам'ятати, що МК обох архитектур мають обмежену ємність пам'яті для зберігання даних. Якщо в процесорі є окремий стек і обсяг записаних у нього даних перевищує його ємність, то відбувається циклічна зміна вмісту покажчика стека, і він починає посилатися на раніше заповнений осередок стека. Це означає, що після занадто великої кількості викликів підпрограм у стеці виявиться неправильна адреса повернення. Якщо МК використовує загальну область пам'яті для розміщення даних і стека, то існує небезпека, що при переповненні стека відбудеться запис в область даних або буде зроблена спроба запису даних, що завантажують у стек, в область ПЗП.

2.4 Порти уведення/виводу

Кожний МК має деяку кількість ліній уведення/виводу, які об'єднані в 8-розрядні паралельні порти уведення/виводу PTx («x» – ім'я порту, використовуване в технічному описі). У карті пам'яті МК кожний порт уведення/виводу поданий регістром даних порту DPTx. У режимі уведення логічні рівні сигналів на лініях порту PTx відображаються нулями й одиницями у відповідних розрядах регістра DPTx. У режимі виводу дані, записані під керуванням програми в регістр DPTx, передаються на виводи МК, які відзначені як лінії порту PTx. Звертання до регістра даних DPTx здійснюється тими ж командами, що й звертання до комірок оперативної пам'яті. Крім того у багатьох МК окремі розряди портів можуть бути опитані командами бітового процесора.

Залежно від функцій, які реалізують ті або інші порти уведення/виводу, розрізняють наступні типи паралельних портів:

1 Односпрямовані порти, призначені у відповідності до специфікації МК тільки для уведення або тільки для виводу інформації.

2 Двонаправлені порти, напрямок передачі яких (уведення або вивід) визначається в процесі ініціалізації системи.

3 Порти з альтернативною функцією. Окремі лінії цих портів зв'язані з убудованими в МК периферійними пристроями, такими, як таймер, АЦП, контролери послідовних приємопередатчиків. Якщо відповідний периферійний модуль МК не використовується, то його виводи можна задіяти як звичайні лінії уведення/виводу. Навпроти, якщо модуль активізований, то приналежні йому лінії введення/виводу автоматично конфігуруються відповідно до функціонального призначення в модулі й не можуть бути використані як лінії уведення/виводу.

4 Порти зі змінюваною програмно керованою схемотехнікою вхідного буфера.

По суті, порти служать як пристрій узгодження масштабів часу, у яких функціонують об'єкт керування і ядро МК, що робить обробку інформації. Об'єкт керування й МПС на основі МК асинхронні стосовно один одного. Для якісного керування інформація з датчиків стану об'єкта має бути отримана в моменти часу, обумовлені поводженням об'єкта. Однак не завжди саме в ці моменти часу інформація може бути сприйнята МПС. У цьому випадку порти уведення/виводу виконують функцію накопичувача, що запам'ятовує стан об'єкта в один момент часу й передає його в процесор в інший момент часу. Розрізняють три типи алгоритмів обміну між МК і зовнішнім пристроєм через паралельні порти введення/виводу:

1 Режим простого програмного уведення/виводу.

2 Режим уведення/виводу зі стробуванням.

3 Режим уведення/виводу з повним набором сигналів квитування.

Схемотехнічні особливості буферів ліній уведення/виводу МК. По-перше, слід зазначити, що у всіх сучасних МК двонаправлені порти уведення/виводу виконані з можливістю незалежного завдання напрямку передачі кожної лінії. Об'єднання груп ліній у порти дозволяє організувати звертання до них як до комірок пам'яті, що зручно при організації обміну в паралельному форматі. Але якщо буде потреба, кожна лінія може бути сконфігурована індивідуально й обслужена командами бітового процесора, незалежно від інших ліній того самого порту введення/виводу. З огляду на цю обставину, схемотехника портів уведення/виводу розглядається на рівні однієї лінії.

Розрізняють три типи драйверів уведення/виводу:

- Двонаправлені лінії, які настроюються на уведення або на вивід програмуванням біта в регістрі напрямку передачі DDPTx.
- Квазідвонаправлені лінії, які не вимагають попередньої ініціалізації, але мають деякі особливості при зчитуванні.
- Двонаправлені лінії з можливістю програмного підключення «підтягаючих» резисторів.

Прикладом драйверів першого типу можуть бути драйвери ліній уведення/виводу МК HC05 і HC08 фірми Motorola (рис. 2.2).

Кожній лінії порту поставлений відповідно однойменний розряд регістра напрямку передачі DDPTx. Нульове значення розряду конфігурує лінію на уведення, одиничне – на вивід. Після скидання МК усі лінії настроєні на уведення. На рисунку 2.2 видно, що в режимі уведення безпосередньо в момент зчитування логічний рівень сигналу лінії PTx передається на внутрішню магістраль даних, минаючи регістр даних порту DPTx. У процесі читання стан лінії не запам'ятовується в регістрі DPTx, і, отже, кожне нове звертання до порту уведення може повертати нове значення. У режимі уведення транзистори VT1 і VT2 закриті, буфер перебуває у високоомному стані (Z-стан). Можлива ситуація, при якій операція читання непідключеного входу буде повертати нульове значення. Тому, якщо як джерело сигналу використовується відкритий колекторний вихід або релейний контакт, то рівень сигналу при розімкнутому контакті в загальному випадку не визначений. Для завдання одиничного логічного рівня сигналу при розімкнутому контакті варто підключити зовнішній резистор, що звичайно позначають R_{PULLUP}. У режимі виводу транзистори VT1 і VT2 управляються сигналом з виходу тригера регістра даних DPTx.

Прикладом квазідвонаправлених драйверів портів можуть служити порти МК Intel MCS-51 (рис. 2.3). Особливість цих драйверів полягає в тому, що при зчитуванні повертається значення, що дорівнює логічному добутку сигналу на лінії й вмісту однойменного тригера регістра даних порту DPTx. Із цієї причини ті розряди порту, які будуть зчитуватися, повинні бути попередньо встановлені в 1 командою запису в порт і лише потім прочитані. Квазідвонаправлені порти не мають регістра напрямку передачі й, отже, не повинні ініціалізуватися. Крім того драйвер лінії портів цього

типу містить «підтягуючий» резистор, тому операція читання розімкнутого контакту буде повертати 1.

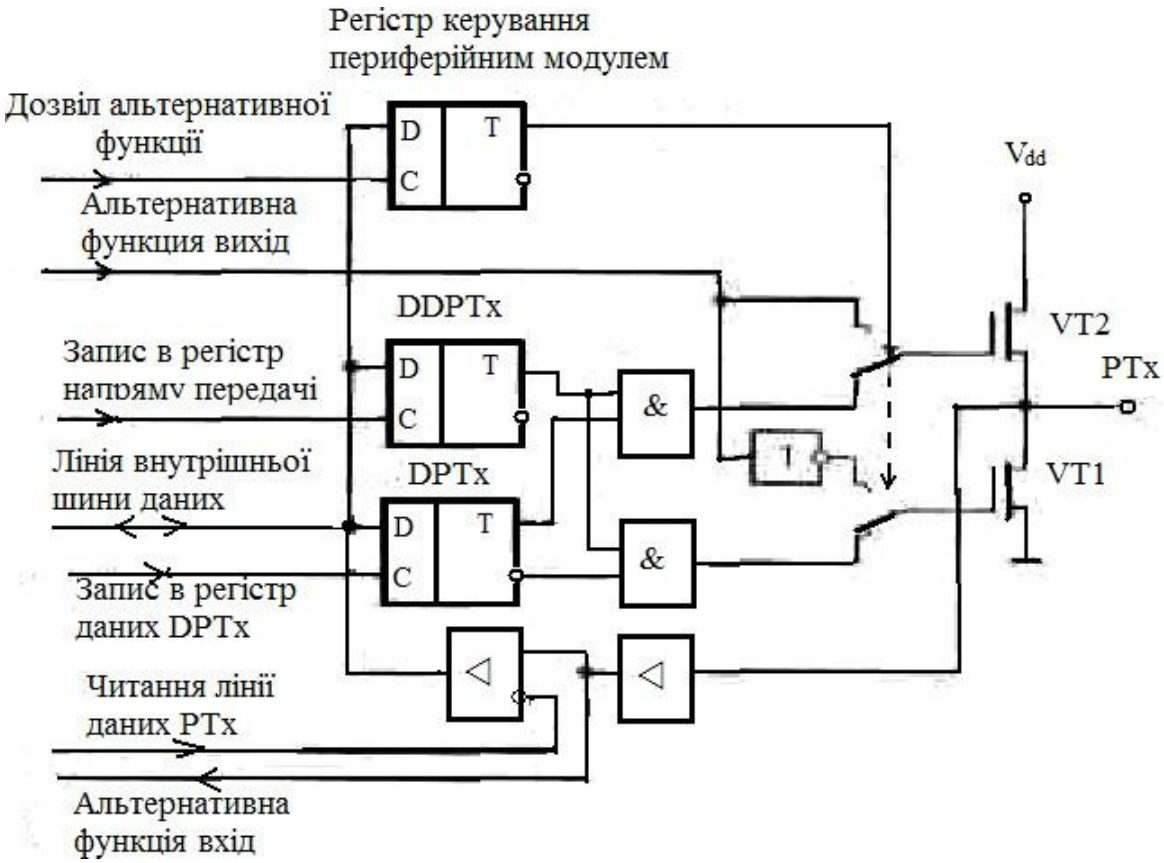


Рисунок 2.2 – Схемотехніка двонаправленої лінії порту

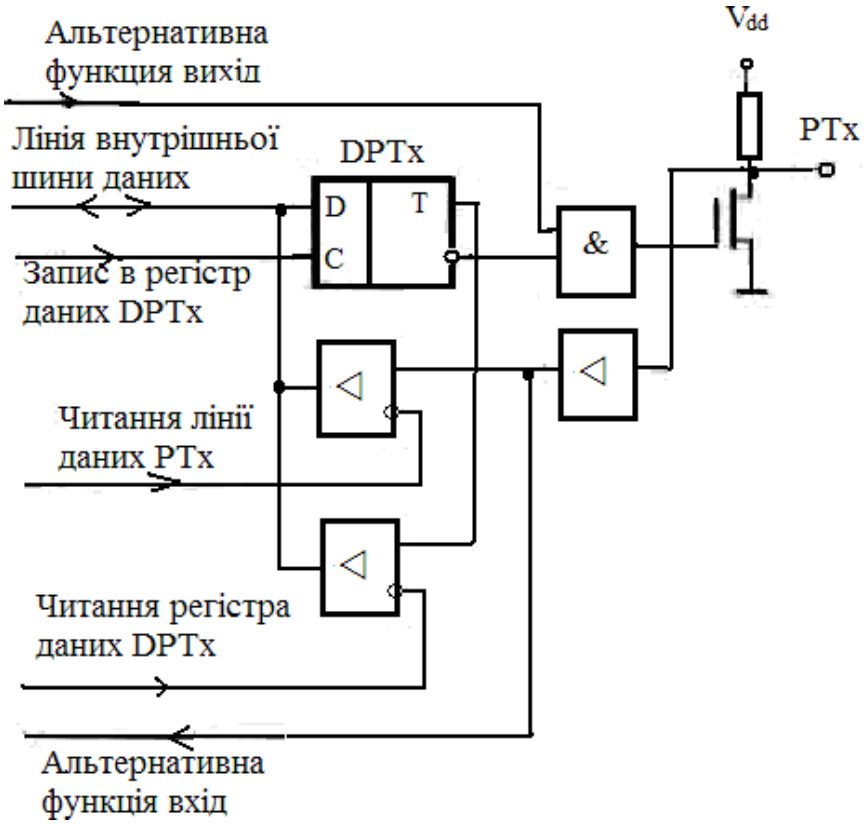


Рисунок 2.3 – Схемотехніка квазідвонаправленої лінії порту

Драйвери ліній зі змінюваною схемотехнікою можуть бути виконані двома способами, однак переслідувана мета одна – скоротити кількість навісних елементів плати МП контролера. У першому випадку драйвер кожної лінії містить «підтягуючий до 1» резистор, що забезпечує рівень логічної одиниці на вході при розімкнутому контакті, у другому випадку драйвер доповнений «підтягуючим до 0» резистором , що здатний служити навантажувальним резистором датчика, вихідний каскад якого виконаний за схемою емітерного повторювача. Логіка керування убудованими «підтягуючи ми» резисторами однакова для обох типів драйверів:

- Підключення «підтягуючи» резисторов допускається апаратними засобами драйвера тільки при конфігуруванні лінії порту на уведення.
- Спеціальний біт регістра конфігурації МК дозволяє програмне підключення «підтягуючих» резисторів на всіх лініях уведення одночасно, але не виконує це підключення.
- Комутацією «підтягуючих» резисторів кожної лінії керує однойменний біт регістра вхідного опору PTUEx. Значення цього біта може багаторазово змінюватися в ході виконання прикладної програми, тим самим здійснюється динамічне керування вхідним опором лінії порту уведення й струмом споживання цієї лінії.

Звичайно не рекомендується залишати лінії не приєднаними до джерела сигналу. У цьому випадку знижується перешкодозахищеність. Тому якщо датчик являє собою релейний нормально розімкнутий контакт, то не слід відключати резистор R_{PULLUP} на час, коли стан датчика не опитується. Навпроти, якщо використовується релейний датчик з нормально замкнутим контактом, то з метою зниження споживаного струму «підтягуючий» резистор варто комутувати тільки на час опитування.

Розглядаючи особливості драйверів ліній уведення/виводу, не можна не зупинитися на понятті навантажувальної здатності лінії. Розрізняють лінії з нормальною й підвищеною навантажувальною здатністю. Якщо мова йде про нормальну навантажувальну здатність, то варто орієнтуватися на наступні цифри: $I^0_{ВИХ}$ =1.6...2.0 ма, $I^1_{ВИХ}$ = 0.4...2.0 ма= Типові значення підвищеної навантажувальної здатності: $I^0_{ВИХ} = I^1_{ВИХ}$ =25 ма. Граничне значення підвищеної навантажувальної здатності на сьогоднішній день становить $I^1_{ВИХ}$ = 60 ма для Microchip PIC 17. Варто помітити, що кількість виводів з підвищеною навантажувальною здатністю звичайно обмежено. Крім того в довідкових даних зазначений максимальний сумарний струм усіх ліній уведення/виводу, що обмежений тепловідводом корпуса МК.

2.5 Таймери і процесори подій

Більшість завдань керування, які покладають на МПС, повинні виконуватися в реальному часі. Поняття «*керування в реальному часі»* означає здатність МПС одержати інформацію про стан керованого об'єкта, виконати необхідні розрахунки й сформувати керуючі впливи протягом інтервалу часу, після закінчення якого ці впливи викличуть бажану зміну поводження об'єкта. Можливість використання того або іншого МК для керування конкретним пристроєм у реальному часі визначається в першу чергу продуктивністю процесорного ядра, тому що МК повинен встигнути за обмежений час виконати розрахунок коригувального впливу. Однак тільки високої продуктивності недостатньо. Необхідно організувати прийом інформації з датчиків і видачу керуючих сигналів таким чином, щоб при збереженні необхідної точності на ці операції витрачалося якнайменше часу. У противному випадку не залишиться часу для виконання обчислень. Ефективний розподіл завдань керування між різними модулями МК забезпечує можливість якісного керування в реальному часі. Багато підсистем МК використовується для рішення цих завдань, але в першу чергу серед них зазначають *підсистему переривань і модуль таймера.* Розвинена підсистема переривань дозволяє скоротити час реакції МПС на зміни стану об'єкта. Модулі таймерів служать для прийому інформації від датчиків із часоімпульсними виходами, а також для формування керуючих впливів у вигляді послідовності імпульсів з параметрами, що змінюються.

Досвід побудови МПС дозволяє зазначити типові завдання, які повинен уміти вирішувати МК для ефективного керування в реальному часі:

- Відлік рівних інтервалів часу заданої тривалості, повтор алгоритму керування після закінчення кожного такого часового інтервалу. Звичайно цю функцію називають формуванням міток реального часу.
- Контроль за зміною стану лінії уведення МК.
- Вимір тривалості сигналу заданого логічного рівня на лінії уведення МК.
- Підрахунок кількості імпульсів зовнішнього сигналу на заданому часовому інтервалі.
- Формування на лінії виводу МК сигналу заданого логічного рівня із програмувальною затримкою стосовно зміни сигналу на лінії уведення.
- Формування на лінії виводу МК імпульсного сигналу із програмувальною частотою й програмувальним коефіцієнтом заповнення.

Кожне з перелічених завдань окремо може бути виконано тільки програмними засобами, без використання спеціальних апаратних рішень. Так, для формування інтервалів часу варто завантажити в регістр центрального процесора число, а потім виконувати команду декрементирування цього регістра доти, поки вміст регістра не стане

дорівнювати нулю. Звичайно таку «зв'язку» команд називають програмною затримкою. Для контролю за зміною стану лінії уведення варто організувати постійне опитування лінії однією з команд читання порту уведення/виводу. Таке рішення називають *поллінгом*. Схожі рішення можна запропонувати й для інших перелічених завдань. Однак усі ці рішення будуть мати один недолік: неможливість виконання обчислень одночасно з відліком часового інтервалу або контролем стану лінії уведення/виводу. Тому для виконання функцій, пов'язаних з керуванням у реальному часі, до складу МК включаються спеціальні апаратні засоби, які називають *таймерами.*

Модуль таймера 8-розрядного МК являє собою 16-розрядний лічильник зі схемою керування (рис. 2.4**)**.

У карті пам'яті МК лічильник відображається двома регістрами: TH – старший байт лічильника, TL – молодший байт. Регістри доступні для читання й для запису. Напрямок рахунку лічильника – тільки прямий, тобто при надходженні тактових імпульсів десятковий еквівалент двійкового коду лічильника змінюється убік збільшення. Залежно від програмних настроювань лічильник може використати одне із двох джерел тактуваня:

- Імпульсну послідовність із виходу керованого дільника частоти f_{BUS}.
- Зовнішню імпульсну послідовність, що надходить на один із входів МК.

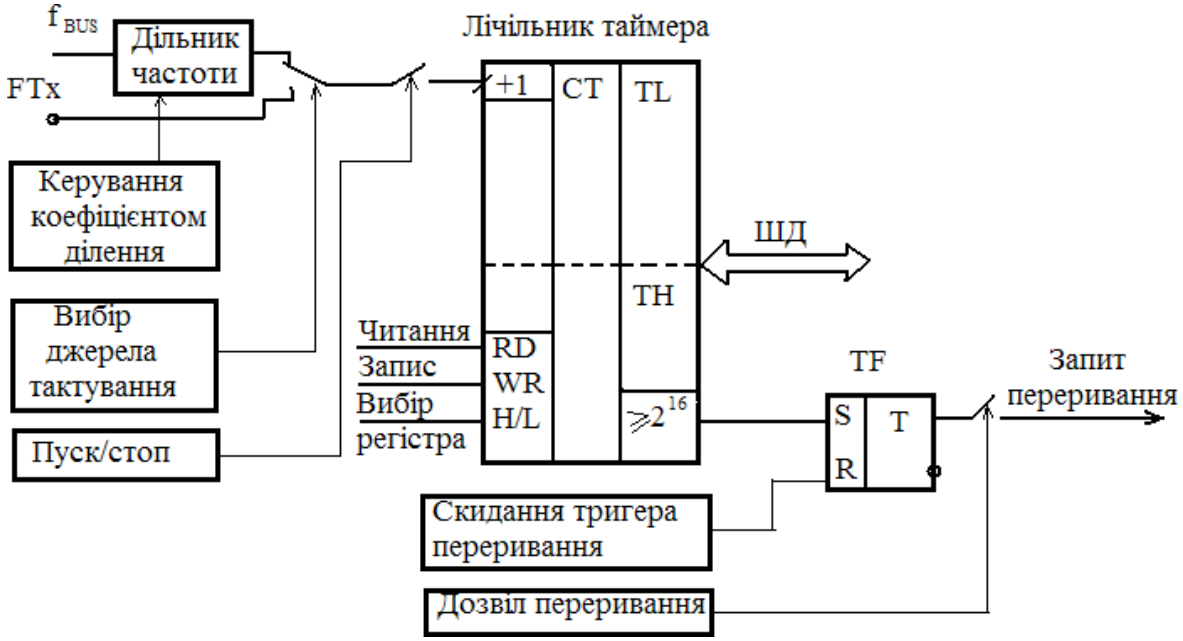


Рисунок 2.4 – Структура модуля таймера/лічільника

У першому випадку говорять, що лічильник працює в *режимі таймера,* у другому – у *режимі лічильника подій.* При переповненні лічильника встановлюється тригер переповнення TF, що генерує запит на

переривання, якщо переривання від таймера дозволені. Після переповнення робота лічильника триває. Послідовність зміни кодів наступна: $FFFF, $0000, $0001 і т.д. Пуск і зупинка лічильника можуть виконуватися тільки під управлінням програми за допомогою установлення/скидання відповідного біта. Програма також може встановити старший (TH) і молодший (TL) байти лічильника в довільний стан або прочитати поточний код лічильника. Однак ці операції не слід виконувати в процесі рахунку, тому що тривалість операції читання або запису може перевищити тривалість періоду частоти тактування лічильника. Тоді за час читання одного з байтів другий встигне змінитися. У результаті буде прочитана недостовірна інформація. З цієї причини може виявитися помилковою операція запису. Наприклад, користувач записує в регістри лічильника код $0005: спочатку старший байт $00, а потім – молодший $05. Якщо поточний стан лічильника користувачеві невідомий, то може виявитися, що на момент завершення операції запису старшого байта молодший дорівнює $FF. Тоді в процесі запису молодшого байта старший, раніше записаний, встигне змінитися, і кінцевий код лічильника буде дорівнювати $0105.

Якщо розглянутий таймер використовується для виміру часового інтервалу t_x (рис. 2.5), то необхідно виконати наступну послідовність дій:

1 Перервати виконання поточної програми при зміні сигналу на лінії PTx з 0 на 1. У підпрограмі переривання встановити регістри лічильника таймера в $0000 і дозволити рахування.

2 При зміні сигналу на лінії PTx з 1 на 0 ще раз перервати виконання програми МК. У підпрограмі переривання зупинити рахунок. Код у регістрах TH і TL буде дорівнювати тривалості інтервалу t_x, вираженому кількістю періодів частоти тактування лічильника таймера.

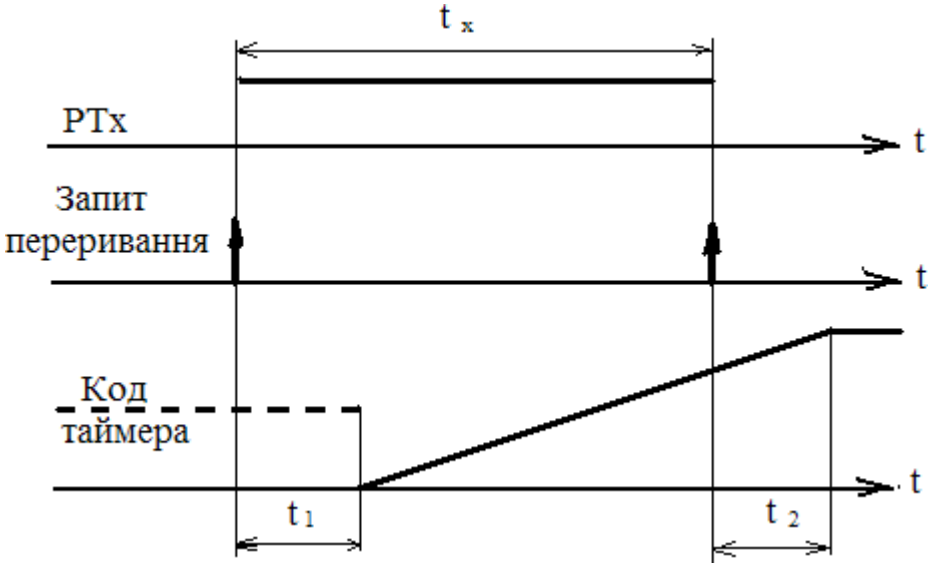


Рисунок 2.5 – Вимірювання часового інтервалу за допомогою «класичного» таймера

Однак такий спосіб виміру годиться для сигналів, тривалість яких становить 1 мс і більше. Моменти дозволу рахунку таймера t_1 і його зупинки t_2 не збігаються з моментами зміни сигналу на вході РТх, тому що пуск і зупинка виконуються в підпрограмі переривання. Помилка рахунку дорівнює $t_1 - t_2$. Кожне зі значень t_1 або t_2 визначається часом переходу МК до виконання підпрограми переривання й часом виконання деякої кількості інструкцій, що передують команді дозволу або зупинці рахунку таймера. Друга величина є систематичною помилкою й може бути врахована при виконанні подальших розрахунків, у той час як перша – часом переходу на підпрограму переривання – величина випадкова, котра залежить від особливостей виконання програмного забезпечення МПС. Так, якщо розглянутих каналів виміру мало й зміна сигналів на входах $РТх_i$ відбулася одночасно, то в першому з обслужених за перериванням каналів помилка буде мінімальною, а в останньому – максимальною. Ця максимальна помилка може скласти кілька десятків мікросекунд, тому розглянутий метод не може бути використаний для виміру часових інтервалів мікросекундного діапазону. У противному випадку точність виміру буде недостатньою. Поряд із завданням виміру часового інтервалу може виникнути необхідність одночасного формування часоімпульсних сигналів за декількома каналами. Тоді бажано, щоб МК мав у своєму складі кілька таймерів. Збільшення кількості модулів таймерів, інтегрованих на кристал МК, є об'єктивною тенденцією вдосконалювання структури сучасних МК.

Розглянутий «класичний» модуль таймера найчастіше використовується в МК із архітектурою MCS-51. Він зазнав деяких вдосконалень:

- Додаткова логіка рахункового входу дозволяє тактовим імпульсам надходити на вхід лічильника, якщо рівень сигналу на одній з ліній уведення дорівнює 1. Таке рішення підвищує точність виміру часового інтервалу, тому що інтервали t_1 і t_2 тепер не є складовою погрішності виміру.
- Реалізовано режим перезавантаження лічильника довільним кодом у момент переповнення. Це дозволяє формувати мітки реального часу з періодом, відмінним від періоду повного коефіцієнта рахунку, який дорівнює 2^{16}.

Однак ці доповнення не усувають головного недоліку модуля «класичного» таймера – неможливості одночасного обслуговування (виміру або формування імпульсного сигналу) відразу декількох каналів.

Удосконалювання структури підсистеми реального часу 8-розрядних МК ведеться двома напрямами:

1 Просте збільшення кількості модулів таймерів. Цей шлях характерний для частини МК компаній Philips і Atmel зі структурою MCS-51, для МК компаній Mitsubishi і Hitachi.

2 Модифікація структури модуля таймера, при якій збільшення кількості каналів досягається не збільшенням кількості лічильників, а

введенням додаткових апаратних засобів *вхідного захоплення й вихідного порівняння.* Такий шлях характерний для 8-розрядних МК Motorola, Microchip, Philips, Infineon.

Типова структура вдосконаленого модуля таймера зображена на рисунках 2.6 і 2.8. Лічильник таймера доповнений апаратними засобами вхідного захоплення й вихідного порівняння. Ці засоби прийнято називати каналом вхідного захоплення IC (Input Capture) і вихідного порівняння OC (Output Compare).

Принцип дії каналу вхідного захоплення пояснює рисунок 2.6.

Схема детектора події «спостерігає» за рівнем напруги на одному з входів МК. Звичайно це одна з ліній порту уведення/виводу. При зміні рівня логічного сигналу на вході детектора з 0 на 1 або навпаки виробляється строб запису й поточний стан лічильника таймера записується в 16-розрядний регістр даних TIC каналу захоплення. Описану дія в мікропроцесорній техніці називають подією захоплення. Передбачено три типи зміни сигналу на вході детектора, які сприймаються як подія захоплення:

- Зміна логічного рівня з 0 на 1 (наростаючий фронт сигналу).
- Зміна логічного рівня з 1 на 0 (падаючий фронт сигналу).
- Будь-яка зміна логічного рівня сигналу.

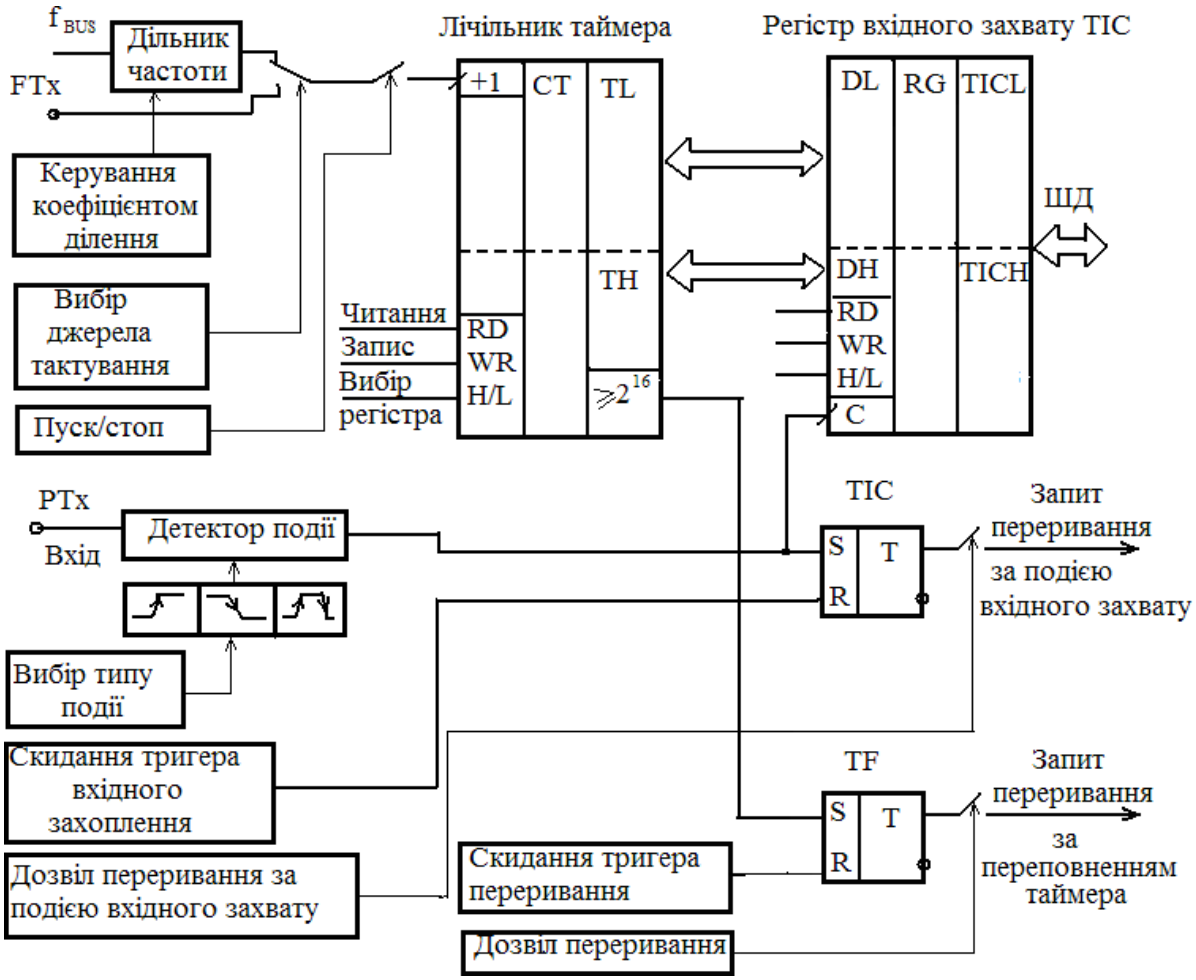


Рисунок 2.6 – Структура каналу вхідного захоплення

Вибір типу події захоплення встановлюється в процесі ініціалізації модуля таймера й може багаторазово змінюватися під час виконання програми. Кожна подія захоплення відзначається установленням у 1 тригера ТІС. Стан тригера може бути считаний програмно, а якщо переривання за подією захоплення дозволені, то генерується запит на переривання.

Часові діаграми на рисунку 2.7 пояснюють процес виміру часового інтервалу з використанням режиму вхідного захоплення.

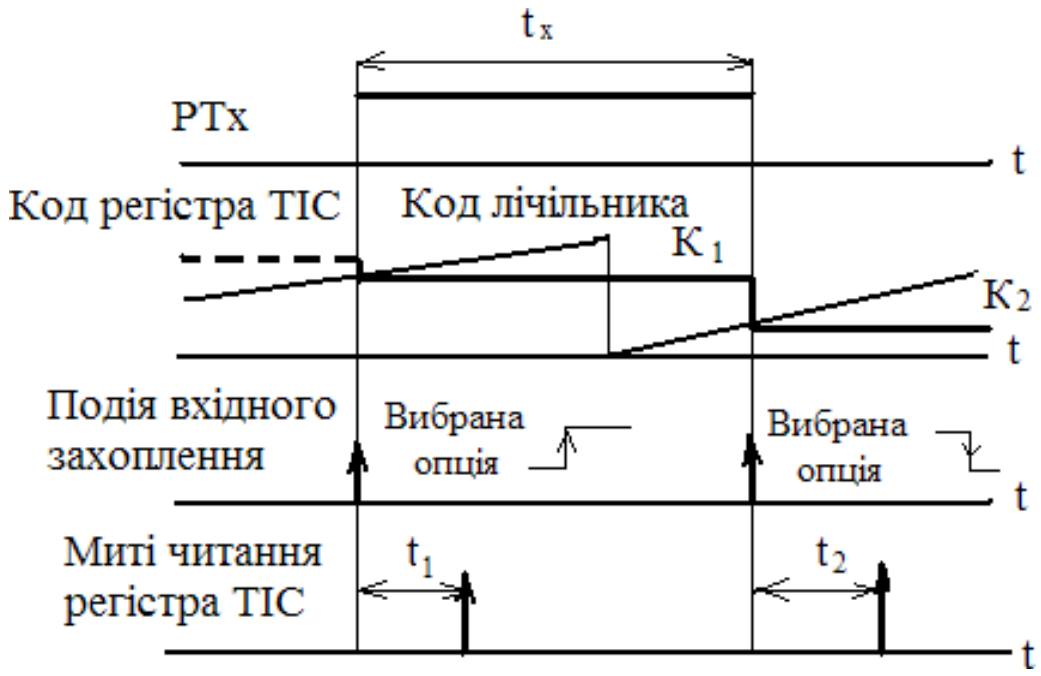


Рисунок 2.7 – Вимірювання часового інтервалу за допомогою засобів каналу вхідного захоплення

Спочатку детектор події ініціалізується на контроль за наростаючим фронтом сигналу на лінії РТх. При зміні рівня сигналу з 0 на 1 код лічильника $К_1$ копіюється в регістр каналу захоплення ТІС. Тригер ТІС установлюється в 1, одночасно формується запит на переривання: таймер «повідомляє» МК про те, що вимірюваний інтервал почався. Із затримкою часу t_1 стосовно моменту появи запиту на переривання МК зчитує код $К_1$ з регістра ТІС, скидає тригер ТІС і ініціалізуєдетектор події на контроль за падаючим фронтом сигналу РТх. При зміні рівня сигналу з 1 на 0 детектор знову фіксує подію захоплення й код лічильника $К_2$ копіюється в регістр ТІС. Знову виставляється запит на переривання, із затримкою t_2 цей код зчитується у пам'ять МК. Різниця кодів $К_2 – К_1$ і є тривалість вимірюваного часового інтервалу, виражена в числі періодів частоти тактування лічильника таймера. Максимальна помилка виміру дорівнює двом періодам

частоти тактування, тому що похибка детектора подій не може перевищувати одиниці квантування таймера. Час переходу до підпрограм переривання t_1 або t_2 не робить впливу на точність виміру, тому що копіювання поточного стану лічильника здійснюється апаратними, а не програмними засобами. Однак час переходу на підпрограму переривання t_1 накладає обмеження на тривалість вимірюваного інтервалу t_x, тому що обговорюваний метод реалізуємий за умови, що друга подія захоплення відбудеться пізніше, ніж код K_1 буде зчитаний у пам'ять МК.

Структура апаратних засобів каналу вихідного порівняння представлена нарисунку 2.8.

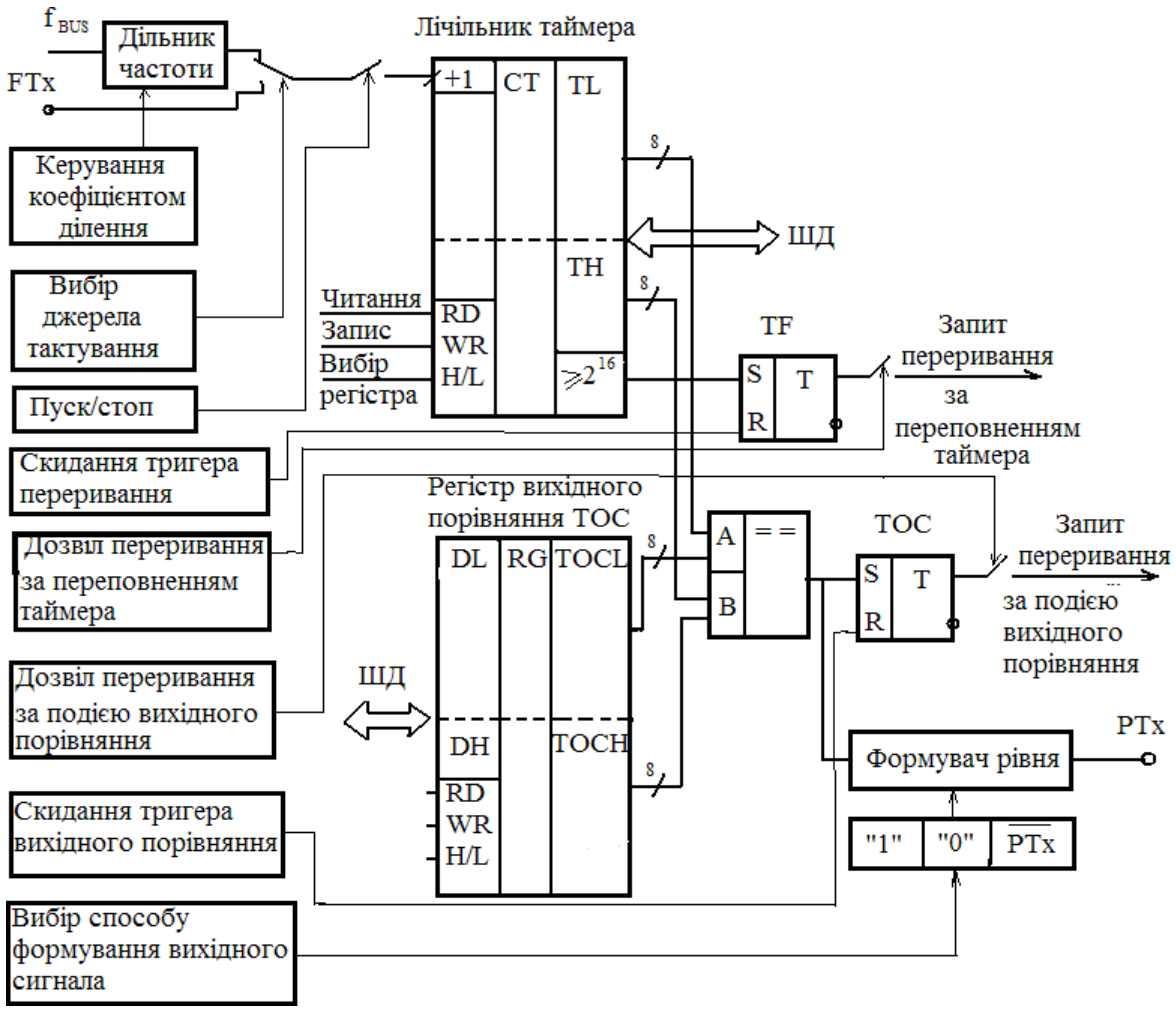


Рисунок 2.8 – Структура каналу вихідного порівняння

Багаторозрядний цифровий компаратор безупинно порівнює код лічильника таймера з кодом, що записаний в 16-розрядному регістрі TOC каналу порівняння. У момент рівності кодів на одному з виводів МК (PTx на рис. 2.8) установлюється заданий рівень логічного сигналу. Розглянуту дію називають подією вихідного порівняння. Передбачено три типи зміни сигналу на виході PTx у момент події вихідного порівняння:

- Інвертування сигналу на виході.
- Установка низького логічного рівня.
- Установка високого логічного рівня.

При настанні події захоплення встановлюється в 1 тригер ТОС. Аналогічно попередньому випадку, стан тригера може бути зчитаний програмно, а якщо переривання за подією вихідного порівняння дозволені, то генерується запит на переривання. Часові діаграми на рисунку 2.9 ілюструють спосіб формування часового інтервалу попередньо розрахованої тривалості t_x з використанням апаратних засобів каналу вихідного порівняння.

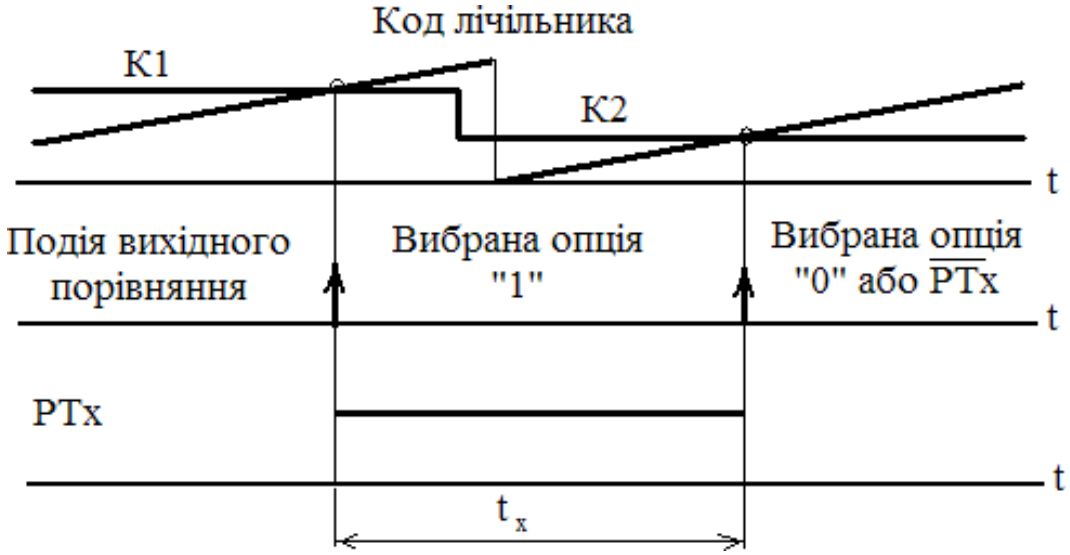


Рисунок 2.9 – Формування часового інтервалу з використанням апаратних засобів каналу вихідного порівняння.

Перша подія порівняння в момент t_1 формує наростаючий фронт сигналу РТх. Одночасно генерується запит на переривання МК і в підпрограмі обробки переривання відбувається завантаження нового коду порівняння К2. Час, необхідний для запису нового значення в регістр каналу порівняння ТОС, обмежує мінімальну тривалість формованого часового інтервалу. У момент t_2 настає друга подія порівняння й вихід РТх установлюється в 0. Таким чином, тривалість сформованого часового інтервалу t_x визначається тільки різницею кодів і не залежить від особливостей програмного забезпечення МК.

У розглянутих прикладах кожній події модуля таймера (зміні рівня логічного сигналу на вході або виході МК) ставиться відповідно код лічильника, тобто лічильник використовується для створення безупинно змінних міток часу. На відміну від «класичного» таймера, де лічильник використовується безпосередньо для формування коду вимірюваного часового інтервалу, в удосконаленому таймері лічильник лише створює образ часу, подібно годинникам. А всі дії з формування або виміру часових інтервалів роблять апаратні засоби порівняння/захоплення. Тому лічильник

у складі модуля вдосконаленого таймера називають «лічильником часової бази» або просто «часовою базою». Ця ж термінологія збережеться й надалі в модулях процесорів подій.

Апаратні засоби вдосконаленого таймера дозволяють вирішити багато завдань керування в реальному часі. Однак процес удосконалювання алгоритмів керування висуває все нові вимоги до структури МК. І, як наслідок, усе більш чітко проявляються обмеження модулів удосконаленого таймера:

- Недостатня кількість каналів порівняння й захоплення, що належать одному лічильнику часової бази. У результаті неможливо сформувати синхронізовані між собою багатоканальні імпульсні послідовності.
- Однозначно певна конфігурація каналів (або захоплення, або порівняння) часто не задовольняє користувача.
- З використанням засобів вихідного порівняння можливе формування сигналу за способом широтно-імпульсної модуляції (ШІМ), однак несуча частота ШІМ-сигнала тим менше, чим більше обчислень потрібно виконувати при реалізації алгоритму керування й чим більше кількість ШІМ-каналів потрібно реалізувати.

Наступний етап у розвитку модулів підсистеми реального часу – *модулі процесорів подій.* Уперше модулі процесорів подій були запропоновані фірмою Intel у складі МК 8xc51 FA/FB/FC/GB, пізніше аналогічний модуль з'явився в МК із ядром MCS-51 фірми Philips. Модуль, що входить до складу перелічених МК, зветься програмувальним рахунковим масивом PCA (Programmable Counter Array). У МК інших фірм аналогічні за функціональним призначенням модулі позначають CAPCOM (Infineon), TIM08 (сімейство HC08 Motorola).

Структурна схема процесора подій наведена на рисунку 2.10.

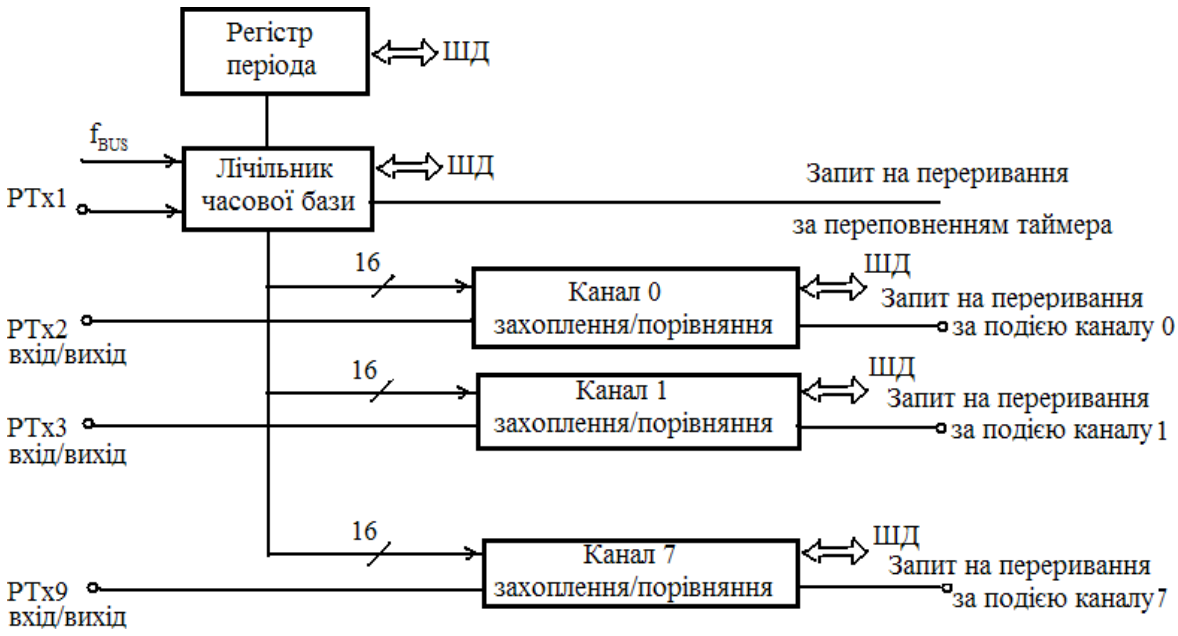


Рисунок 2.10 – Структурна схема процесора подій

Модуль процесора подій містить у собі 16-розрядний лічильник часової бази й декілька універсальних каналів захоплення/порівняння. Лічильник може тактуватися імпульсною послідовністю з виходу програмувального дільника частоти стробування міжмодульних магістралей f_{BUS} або зовнішнім генератором. Лічильник має опції пуску/зупинки й скидання в 0. У деяких моделях процесора подій лічильник часової бази доступний для читання «на льоту». Режим читання «на льоту» передбачає автоматичне копіювання вмісту старшого й молодшого байтів лічильника в спеціальні буферні регістри під час виконання операції читання зазначеного в специфікації байта лічильника (старшого або молодшого). Тоді при читанні другого байта лічильника вертається код з відповідного буферного регістра. Тим самим виключається помилка зчитування через зміну стану лічильника часової бази в процесі читання. Найбільш сучасні моделі процесора подій 8-розрядних МК допускають зміну коефіцієнта рахунку лічильника часової бази або, що те саме, зміну періоду його роботи. Для цього в складі модуля є двобайтовий програмно доступний регістр періоду й багаторозрядний цифровий компаратор (не плутати з каналом захоплення). При збігу поточного коду лічильника часової бази з кодом періоду тригери лічильника часової бази автоматично скидаються в 0.

Універсальні канали захоплення/порівняння повністю ідентичні один одному й залежно від програмних настроювань можуть працювати в одному із трьох режимів:

- Режим вхідного захоплення.
- Режим вихідного порівняння.
- Режим широтно-імпульсної модуляції (ШІМ).

Перші два режими за принципом дії нічим не відрізняються від аналогічних режимів модуля вдосконаленого таймера. Програмно-логічна модель кожного каналу включає двобайтовий регістр даних каналу й тригер події. Залежно від обраного режиму регістр даних каналу використовується апаратними засобами для запису коду часової бази в момент настання вхідного захоплення або для зберігання коду вихідного порівняння. Тригер установлюється при настанні кожного із цих подій. При роботі каналу в режимі вихідного порівняння можуть виникати порушення алгоритму роботи, що призводять до неправильного формування сигналу на виході РТхі-модуля. Причиною таких збоїв є зміна під управлінням програми величини коду порівняння в процесі роботи каналу. Найбільш сучасні моделі процесора подій передбачають для таких випадків спеціальний режим буферованого порівняння, при якому: 16-розрядний регістр коду порівняння дублюється; у кожний момент часу до входу компаратора виявляється підключеним один з регістрів даних, а для запису виявляється доступним інший; у момент настання події вихідного порівняння регістри автоматично міняються місцями.

У режимі широтно-імпульсної модуляції на виводу PTxi МК формується послідовність імпульсів з періодом, рівним періоду роботи лічильника часової бази. Тривалість імпульсу (у деяких моделях – тривалість паузи) прямо пропорційна коду в регістрі даних каналу. Режим ШІМ надзвичайно зручний з погляду програмного обслуговування. Якщо зміни коефіцієнта заповнення γ не потрібні, то досить один раз занести код γ до регістра даних і проініціалізувати режим ШІМ, й імпульсна послідовність буде відтворюватися з необхідними параметрами без подальшого втручання з боку програми.

Режим ШИМ у різних моделях процесорів подій має істотні відмінності. У модулях програмувального рахункового масиву PCA код коефіцієнта заповнення має однобайтовий формат, отже, дискретність регулювання коефіцієнта заповнення становить 1/256 періоду ШІМ-сигналу. Причому 16-розрядний регістр даних універсального каналу PCA в режимі ШИМ «розпадається» на два однобайтових регістри. Доступним для запису є тільки один з регістрів. На очатку кожного періоду ШІМ-сигналу вміст цього регістра копіюється в другий регістр, що використовується апаратними засобами для формування тривалості імпульсу в поточному періоді. Таке буферування дозволяє уникнути порушень при формуванні імпульсної послідовності у випадках, коли до настання моменту рівності кодів регістра й лічильника відбувається зміна коду коефіцієнта γ, і нове значення γ менше поточного коду лічильника. Тоді порівняння кодів *у* поточному періоді ШІМ-сигналу не настане, й імпульс буде пропущений. Крім недоліку щодо низької дискретності регулювання коефіцієнта заповнення модуль PCA має обмежений набір несучих частот сигналу ШІМ. Це відбувається тому, що коефіцієнт лічильника тимчасової бази не може бути змінений, а регулювання частоти досягається тільки зміною коефіцієнта розподілу попереднього дільника частоти.

Модулі «класичних» таймерів і таймерів зі схемами захоплення/порівняння – досить складні пристрої. Їхня функціональна гнучкість у найпростіших системах керування часто виявляється надлишковою. Тому в деяких маловивідних МК, виконаних у корпусах з 16 і 20 виводами, з метою зниження вартості ІС реалізуються спрощені таймери. Таймери містять 8-розрядний лічильник, що постійно діє, із тригером переповнення й програмувальний дільник частоти. Такі таймери можуть формувати лише мітки реального часу з періодом проходження, що визначається набором коефіцієнтів розподілу програмувального дільника частоти. Тенденція збільшення кількості таймерів/лічильників, інтегрованих на кристал МК знайшла своє продовження й для модулів процесорів подій. Так, у багатьох сучасних МК удосконалювання підсистеми реального часу йде не просто шляхом збільшення кількості каналів процесора подій. Зростає кількість модулів процесорів подій, надаючи розроблювачеві можливість використання декількох «часових баз», що дозволяє обробляти й формувати сигнали різного часового масштабу. Крім того до складу МК уводяться спрощені модулі таймерів, здатні «розвантажити» процесор подій від виконання найпростіших

функцій. І це в тому числі додаткова «часова база».

2.6 Аналогово-цифрові й цифро-аналогові перетворювачі

Відмінна риса багатьох сучасних 8-розрядних МК – інтегрований на кристал МК модуль багатоканального аналого-цифрового перетворювача (АЦП). Модуль АЦП призначений для уведення в МК аналогових сигналів з датчиків фізичних величин і перетворення цих сигналів на двійковий код з метою наступної програмної обробки. Структурна схема типового модуля АЦП зображена на рисунку 2.11.

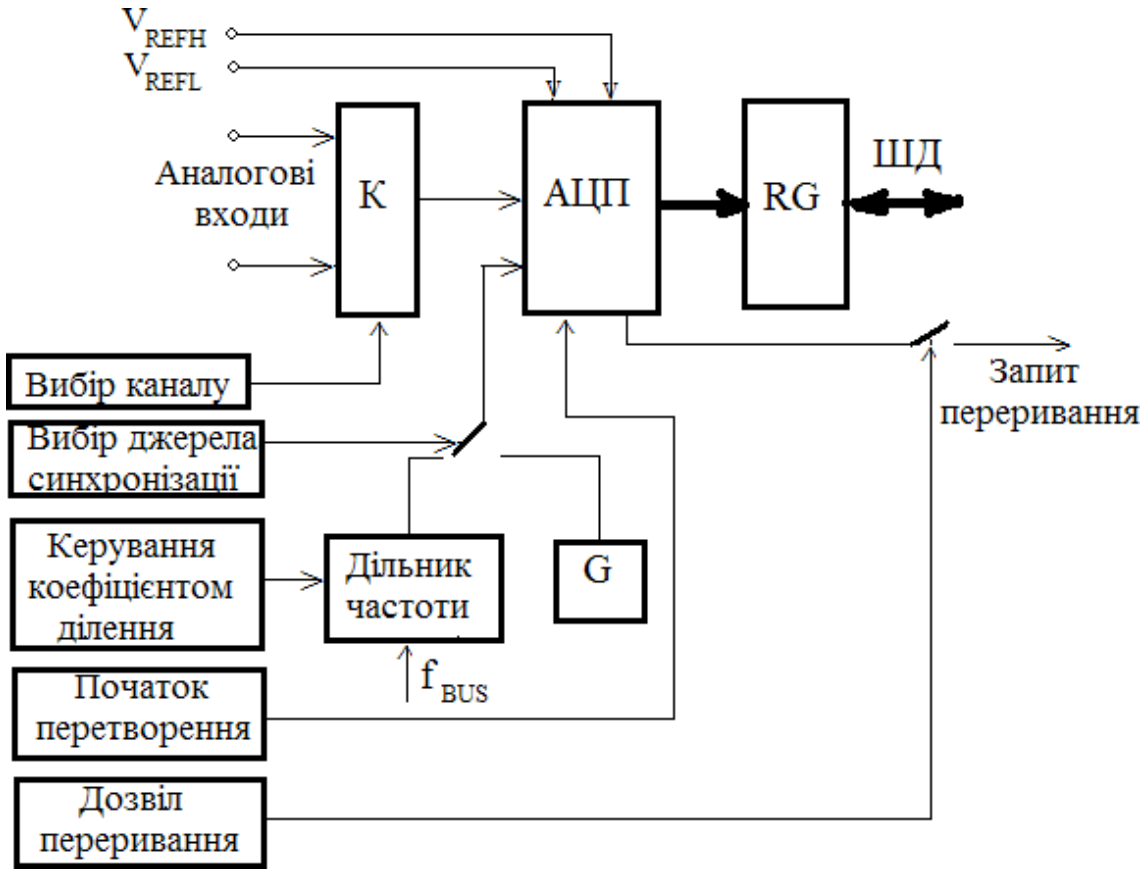


Рисунок 2.11 – Структурна схема модуля АЦП

Багатоканальний аналоговий комутатор служить для підключення одного із джерел аналогових сигналів до входу АЦП. Вибір джерела сигналу для виміру здійснюється за допомогою запису номера каналу комутатора у відповідні розряди регістра керування АЦП. Зазначимо, що в модулях АЦП 8-розрядних МК передбачена тільки програмна установка номера каналу. Режим автоматичного послідовного сканування каналів із записом результату виміру кожного каналу в індивідуальну комірку пам'яті не реалізується.

Діапазон вимірюваних значень напруги аналогових входів визначається опорною напругою $U_{оп.}$ Розв'язна здатність АЦП становить

$U_{оп}$ /2^n, де n – кількість двійкових розрядів у слові результату. Максимальне значення опорної напруги, як правило, дорівнює значенню напрузі живлення МК. Два виводи модуля АЦП використовуються для завдання опорної напруги: V_{REFH} – верхня межа $U_{оп}$. V_{REFL} – нижня межа. Різниця потенціалів на входах V_{REFH} і V_{REFL} і становить $U_{оп}$. Якщо вимірювана напруга $U_{вим} \geq V_{REFH,,}$ то результат перетворення буде дорівнювати FF, код 00 відповідає напругам $U_{вим} \leq V_{REFL}$. Для досягнення максимальної точності виміру варто вибрати максимально припустиме значення $U_{оп}$. У цьому випадку напруга зсуву нуля вхідного буфера й нелінійність передатної характеристики АЦП будуть вносити відносно малі погрішності.

Джерелом синхронізації модуля АЦП може служити убудований RC-генератор або імпульсна послідовність тактування міжмодульних магістралей МК. У першому випадку частота синхронізації АЦП обов'язково виявиться оптимальною, яка рекомендується в технічному описі. У другому випадку обрана в результаті інших міркувань f_{BUS} може виявитися невідповідною для модуля АЦП. На цей випадок у складі деяких модулів передбачений програмувальний дільник частоти $f_{BUS.}$

Момент завершення кожного циклу перетворення відзначається установкою тригера готовності даних. Якщо переривання від модуля АЦП дозволені, то генерується запит на переривання. Як правило, читання регістра результату скидає тригер готовності.

Більшість модулів АЦП має тільки режим програмного запуску: установка одного з бітів регістра режиму запускає черговий вимір. Найбільш універсальні модулі АЦП мають також режим автоматичного запуску, при якому після завершення одного циклу перетворення негайно починається наступний. Однак дані виміри кожного циклу повинні бути прочитані програмним способом .

Цифро-аналогові перетворювачі в складі МК є великою рідкістю. Модулі паралельних ЦАП можна зустріти лише в МК фірм Mitsubishi і Hitachi.

Функція цифро-аналогового перетворювача реалізується засобами модуля програмувального таймера. На одному з виводів МК формується високочастотна імпульсна послідовність із регульованою тривалістю імпульсу. Отриманий сигнал згладжується фільтром нижніх частот на операційному підсилювачі. Розв'язна здатність такого ЦАП визначається дискретністю регулювання коефіцієнта заповнення γ у режимі ШІМ.

2.7 Мінімізація споживання енергії в МП-системах

Одна з основних тенденцій розвитку вбудованих МП-систем – їхня мініатюризація, що вимагає не тільки підвищення ступеня інтеграції елементної бази з метою скорочення кількості ІС системи, але й зниження споживаної цими ІС потужності. Мала енергія споживання в багатьох

випадках є визначальним чинником доцільності реалізації проектованого пристрою в цілому:

- З одного боку, лавиноподібно наростає кількість пристроїв з автономним живленням, які принципово можуть бути виконані тільки на основі елементної бази з малим споживанням енергії.
- З іншого боку, навіть при роботі в складі виробів з живленням від побутової або промислової мережі мікропроцесорний контролер повинен займати жорстко обмежений обсяг, що може бути реалізовано при малій потужності розсіювання мікропроцесорного контролера й, як наслідок, малих розмірах джерела живлення.

Сучасні МК мають кілька режимів роботи, які розрізняються не тільки алгоритмами функціонування МК, але й потужністю споживання:

1 Активний режим (Run mode) – основний режим роботи МК. У цьому режимі МК виконує прикладну програму, тобто керує об'єктом. У активному режимі функціонують усі ресурси МК. У цьому режимі МК споживає максимальну потужність $P_{RUN.}$

2 Режим очікування (Walt mode, або Idle mode, або Halt mode). У цьому режимі припиняє роботу центральний процесор, але продовжують роботу периферійні модулі, які відслідковують поводження об'єкта керування. При необхідності периферійні модулі переводять МК в активний режим роботи, й програма керування обчислює коригувальні впливи для керування об'єктом. Перехід МК із режиму очікування в робочий режим здійснюється за перериваннями від зовнішніх джерел або від периферійних модулів, або при скиданні МК. У режимі очікування потужність споживання МК P_{WAIT} знижується в порівнянні з P_{RUN} у 5...10 разів.

3 Режим зупинки (Stop mode для МК фірми Motorola, або Sleep - mode для МК фірми Microchip, іноді Power Down mode). У цьому режимі припиняє роботу як центральний процесор, так і більшість периферійних модулів. Як правило, перехід МК із режиму зупинки в робочий режим можливий тільки за запитами на переривання від зовнішніх джерел або після подачі активного рівня сигналу на вхід скидання. У режимі зупнки потужність споживання МК P_{STOP} знижується в порівнянні з P_{RUN} на три рівні й становить одиниці мікровольт.

Два останніх режими звуться режимами зниженого енергоспоживання МК. Мінімізація енергії споживання вбудованих МП-систем досягається оптимізацією потужності споживання МК в активному режимі роботи, а також чергуванням у часі активного режиму роботи МК і режимів зниженого енергоспоживання. Потужність споживання МК в активному режимі роботи є однією з важливих технічних характеристик МК. Однак вона не є величиною постійною й залежить від напруги живлення МК і частоти тактування.

Незалежно від фірми-виробника 8-розрядні МК мають три групи виконання за напругою живлення. До першої групи відносяться МК із напругою живлення 5,0 В ± 10 %. Ці МК призначені для роботи в складі

пристроїв з живленням від промислової або побутової мережі. Як правило, це МК середньої або вищої групи складності, з досить високим рівнем споживаної потужності внаслідок розвинених функціональних можливостей. Друга група – МК із розширеним діапазоном напруги живлення: від 2.0...3.0 В до 5.0...7.0 В. Конкретні значення напруг верхньої й нижньої границь визначаються моделлю МК. МК другої групи можуть працювати в складі пристроїв як з мережним, так і з автономним живленням. Нерідкі випадки їхнього використання у виробах з убудованим джерелом безперебійного живлення, які автоматично переходять на живлення з акумуляторів при зниженні напруги мережі. До третьої групи відносяться МК зі зниженою напругою живлення: від 1,8 до 3,0 В. Ці МК призначені для роботи в переносних виробах з автономним живленням, тому що мають струм споживання в 3 рази менший, чим аналогічні МК першої групи, й забезпечують ощадливу витрату енергії елемента живлення.

З огляду на те, що для вторинних джерел живлення звичайно наводиться значення максимального струму навантаження при заданому рівні напруги на виході, а запас енергії автономних джерел оцінюється в А×час, потужність споживання МК прийнято побічно характеризувати струмом споживання при заданих значеннях напруги живлення й частоти синхронізації..

Залежність струму споживання від напруги живлення МК у першому наближенні можна розглядати як прямо пропорційну. Тому зниження напруги живлення досить істотно знижує потужність споживання МК. Однак варто пам'ятати, що для багатьох типів МК зі зниженням напруги живлення зменшується максимально припустима частота тактування f_{BUS}, тобто виграш у споживаній потужності обертається зниженням продуктивності системи. Так, для МК сімейства HC08 фірми Motorola перехід від напруги живлення U_{DD} = 5,0 В до U_{DD}= 3,0 В супроводжується зниженням припустимої частоти тактування від f_{BUS} = 8 МГц до f_{BUS}= 4 МГц. А для МК сімейства HC05 при напрузі живлення U_{DD}= 5,0 В максимальна частота тактування становить f_{BUS}= 4 МГц, при U_{DD} = 3,0 В – f_{BUS} = 2 МГц, а при U_{DD} = 1,8 В – усього 500 кГц. Однак обмеження f_{BUS}, якщо таке є, має східчастий характер, і для багатьох моделей МК можливий режим роботи: $U_{DD} < U_{DDMAX}$, $f_{BUS}= f_{BUSmax}$.

При тій самій напрузі живлення струм споживання МК значною мірою залежить від частоти тактування f_{BUS}. Тому, вибираючи частоту тактування, треба в тому числі мати на меті зниження потужності споживання. Не слід прагнути до гранично високої швидкодії МК у завданнях, які цього не вимагають. Вибір частоти тактування МК у багатьох випадках визначається не тільки міркуваннями обчислювальної продуктивності. Часто визначальним фактором виявляється розв'язна здатність вимірників часових інтервалів на основі таймера або швидкість передачі послідовного інтерфейсу. Але в кожному разі варто детально оцінити необхідну частоту тактування, а потім вибрати f_{BUS} з деяким

запасом. При такому підході технічні умови проекту будуть виконані і пристрій буде економно витрачати енергію.

Сучасні МК не мають нижньої границі частоти тактування. Бажаючи підкреслити дану особливість, у довідкових даних указують, що мінімальна частота тактування дорівнює dc (direct current), тобто як завгодно низька. Це в тому числі означає, що при налагодженні системи користувач може тактувати МК від кнопки з антидріб'язковим тригером, виконуючи програму по кроках.

У більшості моделей МК частота тактування визначається зовнішнім часозадавальним елементом: кварцовим або керамічним резонатором, RC-ланцюгом. Причому частота часозадавального елемента й частота тактування $\mathbf{f_{BUS}}$ жорстко зв'язані коефіцієнтом розподілу убудованого дільника частоти. Тому зміна частоти в процесі виконання програми керування не є можливою. Однак низка останніх сімейств МК деяких фірм (HC08 Motorola, DS87C530) має у своєму складі систему тактування, засновану на принципі синтезатора частоти з контуром фазового автопідстроювання (PLL – phase loop lock). Така система працює як множувач частоти й дозволяє мати в якості часозадавального елемента низькочастотний кварцовий резонатор, що знижує рівень електромагнітного випромінювання при його роботі. Коефіцієнти розподілу контуру PLL підлягають програмуванню й, отже, можуть бути змінені під управлінням програми. Останнє створює можливість зміни частоти тактування з метою зниження потужності споживання МК у проміжки часу, коли висока швидкодія не потрібна.

Крім динамічного керування частотою тактування, можливо також відключення деяких периферійних модулів МК у процесі виконання програми. Так, якщо інформація про стан об'єкта керування знімається рідко, то модуль АЦП варто підключати тільки на час його роботи. Аналогічне рішення може бути застосовано й до інших модулів, у тому числі до модуля енергонезалежної пам'яті EEPROM .

Багато приладів на основі 8-розрядних МК улаштовані таким чином, що протягом тривалого часу виконання програми МК не потрібно зовсім. Для ілюстрації наведемо два приклади.

Приклад 1. Пульт дистанційного керування побутових апаратур. Програмне забезпечення пульта повинно функціонувати тільки в тому випадку, якщо змінюється стан органів керування. В інший час МК перебуває ніби в стані готовності, причому цей стан може тривати цілодобово, наприклад, поки не дивляться телевізор.

Приклад 2. Лічильник теплової енергії. Вимірює температуру й витрату води в трубопроводах опалення один раз за хвилину. Робить за цими показниками розрахунок спожитої теплової енергії й чекає початку чергової хвилини.

Саме для таких випадків передбачені режими зниженого енергоспоживання МК, які, на відміну від розглянутих вище мір зниження потужності споживання, обов'язково припускають зупинку виконання

програми центральним процесором. Так, у прикладі 1 після виконання чергової дії варто перевести МК у режим зупинки (Stop mode). Вихід із цього режиму буде здійснений за запитом на переривання при натисканні будь-якої клавіші пульта керування. У прикладі 2 програма керування повинна виконуватися на початку кожної хвилини. Якщо для відліку інтервалу часу, рівного хвилині, використовується внутрішній таймер, то в проміжках між обчисленнями МК може бути переведений тільки в режим очікування (Wait mode), тому що в цьому режимі залишаються працездатними периферійні модулі МК. Для досягнення більше істотної економії енергії варто перевести МК у режим зупинки, але при цьому прийдеться скористатися зовнішньою часозадавальною схемою.

Для зниження потужності споживання в сучасних МК передбачена можливість програмного відключення периферійних модулів у режимі очікування. Комбінація модулів, що залишилися в роботі, визначає струм споживання в режимі очікування. Тому при описі деяких МК (наприклад, MSP430 фірми Texas Instruments) мова йде про безліч режимів зниженого енергоспоживання.

У деяких МК для режиму зупинки є опція, що дозволяє залишити в роботі генератор кварцового резонатора й один часозадавальний модуль. Так, у МК сімейства HC08 фірми Motorola залишається в роботі модуль базового таймера TBM08, а в МК сімейства PIC16 фірми Microchip – модуль сторожового таймера. Перше зі згаданих рішень трохи більш зручно, тому що дозволяє не поєднувати алгоритм обслуговування скидання із зависання системи (спрацював сторожовий таймер) і алгоритм обслуговування периферії через рівні проміжки часу.

Крім очевидного розходження у функціонуванні периферійних модулів, режими очікування й зупинки відрізняються також характером процесу переходу з режиму зниженого енергоспоживання в активний режим роботи. Вихід з режиму очікування відбувається протягом 3...5 періодів синхронізації МК у той час, як затримка виходу з режиму зупинки становить кілька тисяч періодів синхронізації МК. Якщо тактування МК здійснюється від низькочастотного кварцового резонатора з наступним множенням частоти внутрішніми засобами МК, то під періодом синхронізації варто розуміти період коливання кварцового резонатора. Крім очевидної незручності поганої динаміки системи, що перебуває в стані зупинки, такий великий час переходу в активний режим роботи є причиною додаткової витрати енергії. У МК MSP430 з ультранизьким споживанням енергії додаткова економія енергії досягається за рахунок скорочення часу переходу з режиму низького енергоспоживання в активний режим роботи. Це дозволяє МК MSP430 працювати в складі лічильника теплової енергії при автономному живленні протягом 10 років без заміни батарейок.

Не всі МК мають саме два режими зниженого енергоспоживання. Так, у МК сімейства PIC16 реалізується тільки один режим Sleep (дослівно – «сплячий режим»), а в МК MSP430 – п'ять режимів низького

енергоспоживання LP0...LP4. Але хоча б один режим зниженого енергоспоживання мають усі сучасні МК.

2.8 Моніторинг напруги живлення МК

Джерело живлення МП-системи не може бути ідеальним. Напруга джерела живлення піддається як прогнозованим, так і випадковим змінам, які повинні враховуватися розроблювачем системи. Властивість МП-системи відновлювати працездатність при короткочасних відключеннях напруги живлення або при її «просіданнях» нижче припустимого значення є обов'язковою для сучасних систем керування. Залежно від характеру можливих змін напруги живлення МП-системи можна поділити на наступні групи:

1 Системи з імпульсними джерелами вторинного електроживлення, які стабілізують вихідну напругу в заданому діапазоні (U_{DDMIN} … U_{DDMAX}), у противному випадку втримують вихідну напругу, яка дорівнює нулю.

2 Системи з так званими «гладкими» джерелами вторинного електроживлення, які при значному зниженні напруги мережі пропорційно знижують вихідну напругу.

3 Системи з автономним живленням від батарейок або акумуляторів.

4 Системи з комбінованим живленням. Такі системи автоматично переходять на живлення від автономного джерела у випадку, якщо напруга вторинного джерела живлення перестала задовольняти вимозі:

$$U_{DDMIN} < U_{DD} < U_{DDMAX}.$$

У системах типу 3 і 4 варто обов'язково застосовувати МК із розширеним діапазоном напруги живлення. У системах типу 1 і 2 можуть використатися МК із фіксованою напругою живлення (5 В ± 10 %, 3 В ± 10 %), але при цьому варто уважно проаналізувати реакцію МП-системи на можливі зміни напруги живлення й ужити заходів щодо усунення відмов.

При включенні напруги живлення МК повинен почати виконувати прикладну програму. На етапі наростання напруги живлення МК примусово переводиться в початковий стан, що називають ***станом скидання.*** При цьому встановлюються у вихідний стан внутрішні магістралі МК, сигнали керування й регістри спеціальних функцій. Останні визначають початковий стан периферійних модулів МК. Звичайно цей стан неактивний. Відразу після виходу зі стану скидання МК виконує наступні дії:

1 Запускає генератор синхронізації МК. Для стабілізації частоти тактування внутрішніми засобами МК формується затримка часу $t_{ЗАТ}$.

2 Зчитує енергонезалежні регістри конфігурації у відповідні регістри ОЗП (за необхідністю).

3 Завантажує в програмний лічильник PC адресу початку прикладної програми.

4 Робить вибірку першої команди з пам'яті й починає виконання прикладної програми.

Адресу комірки пам'яті, у якій зберігається код першої команди прикладної програми, називають вектором початкового запуску або вектором скидання. У деяких МК цю адресу однозначно визначають у технічному описі. Про таких МК говорять, що вони мають фіксований вектор скидання. Розроблювач зобов'язаний указати цю фіксовану адресу на етапі формування файла кодів прикладної програми (завантажувальний модуль програми). В інших МК вектор скидання може бути довільно визначений користувачем. На етапі програмування МК бажаний вектор початкового запуску записується в комірки пам'яті з фіксованими адресами й при виході МК зі стану скидання автоматично завантажується в програмний лічильник. Про таких МК говорять, що вони мають завантажувальний вектор скидання. Завантажувальний вектор скидання мають усі 8-розрядні МК фірми Motorola.

Для переходу МК у стан скидання досить подати напругу високого або низького логічного рівня (зазначено в специфікації) на вхід RESET. Традиційно для формування сигналу скидання при включенні напруги живлення використовують RC-ланцюг.

У сучасних МК лінія RESET звичайно виконана двонаправленою і має низький активний рівень. При подачі напруги активного рівня буфер лінії встановлюється в режим уведення й реалізується так зване ***зовнішнє скидання.*** МК може перейти в стан скидання також за сигналами пристроїв контролю стану, які втримуються в самому МК. У цьому випадку говорять, що МК перебуває в стані ***внутрішнього скидання.*** Порядок виходу МК зі станів зовнішнього й внутрішнього скидання в цілому однаковий. При знаходженні в стані внутрішнього скидання буфер лінії RESET установлюється в стан виводу з низьким логічним рівнем на виході й може бути використаний для установки в початковий стан периферійних ІС.

З метою мінімізації кількості допоміжних ІС плати МП-контролера більшість сучасних МК мають у своєму складі блок детектування напруги живлення (схема POR — Power-On-Reset), що формує сигнал внутрішнього скидання при наростанні напруги живлення (рис. 2.12). Схема POR має два пороги спрацьовування, тобто, по суті, є компаратором з гістерезисом. Поріг спрацьовування V_{POR} значно нижче мінімально допустимого напруження живлення МК і дорівнює -1 В. При досягненні напругою живлення значення V_{POR} схема POR фіксує подію включення живлення МК, формує затримку часу t_{POR}, після чого знімає сигнал внутрішнього скидання. Передбачається, що напруга живлення МК протягом t_{POR} встигне досягти номінального значення. Тому швидкість наростання напруги живлення обмежена знизу. У випадку, якщо швидкість наростання недостатня, схема POR не може бути використана для початкового запуску

МК і варто застосувати спеціальну ІС для формування сигналу зовнішнього скидання з нормованим фронтом.

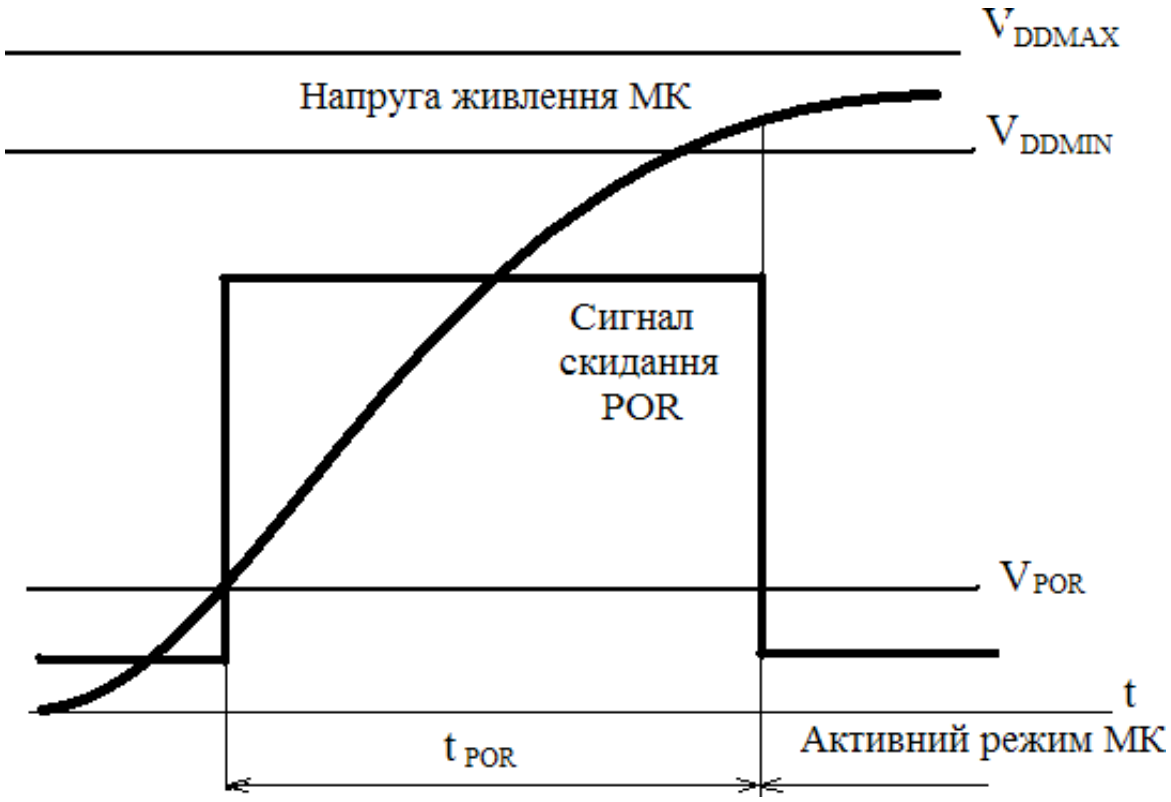


Рисунок 2.12 – Діаграма роботи схеми POR

Порог відпускання V_{POR} не перевищує 200 мв. Тому при використанні гладких джерел живлення може скластися ситуація, при якій напруга живлення сталася менше $V_{DDMIN,}$ але не досягла напруги відпускання схеми POR. Тоді МК може «зависнути». При відновленні напруги живлення до номінального значення МК не запрацює. При використанні імпульсних джерел живлення така ситуація виключається, якщо діапазон стабілізації напруги джерела обраний рівним діапазону допустимих напружень живлення МК.

Для відновлення працездатності МП-системи після «просідання» напруги живлення МК необхідно знову скинути. Із цією метою в сучасних МК реалізований додатковий блок детектування зниженої напруги живлення. У МК сімейства HC08 фірми Motorola такий модуль зветься LVI, аналогічний модуль є в складі МК PIC16 Microchip, Z8 фірми Zilog. Розглянутий модуль генерує сигнал внутрішнього скидання при зниженні напруги живлення до рівня трохи менше V_{DDMIN} (рис. 2.13). Так, для МК MC68HC908JL3 з напругою живлення V_{DDN} = 5 В ± 10 % рівень спрацьовування модуля зниженої напруги живлення лежить у межах 3.6... 4.4 В. Отже, зона нечутливості все-таки залишається. Тому наявність у складі МК модуля зниженої напруги живлення значно знижує

ймовірність зависання МК, але не усуває її повністю. Слід також зазначити, що рівень спрацьовування модуля зниженої напруги живлення значно перевищує напругу збереження даних в ОЗП МК. Подія скидання за сигналом модуля зниженої напруги відзначається спеціальним бітом в одному з регістрів. Отже, аналізуючи цей біт програмно після скидання МК, можна встановити, що руйнування даних не відбулося, і продовжити виконання програми.

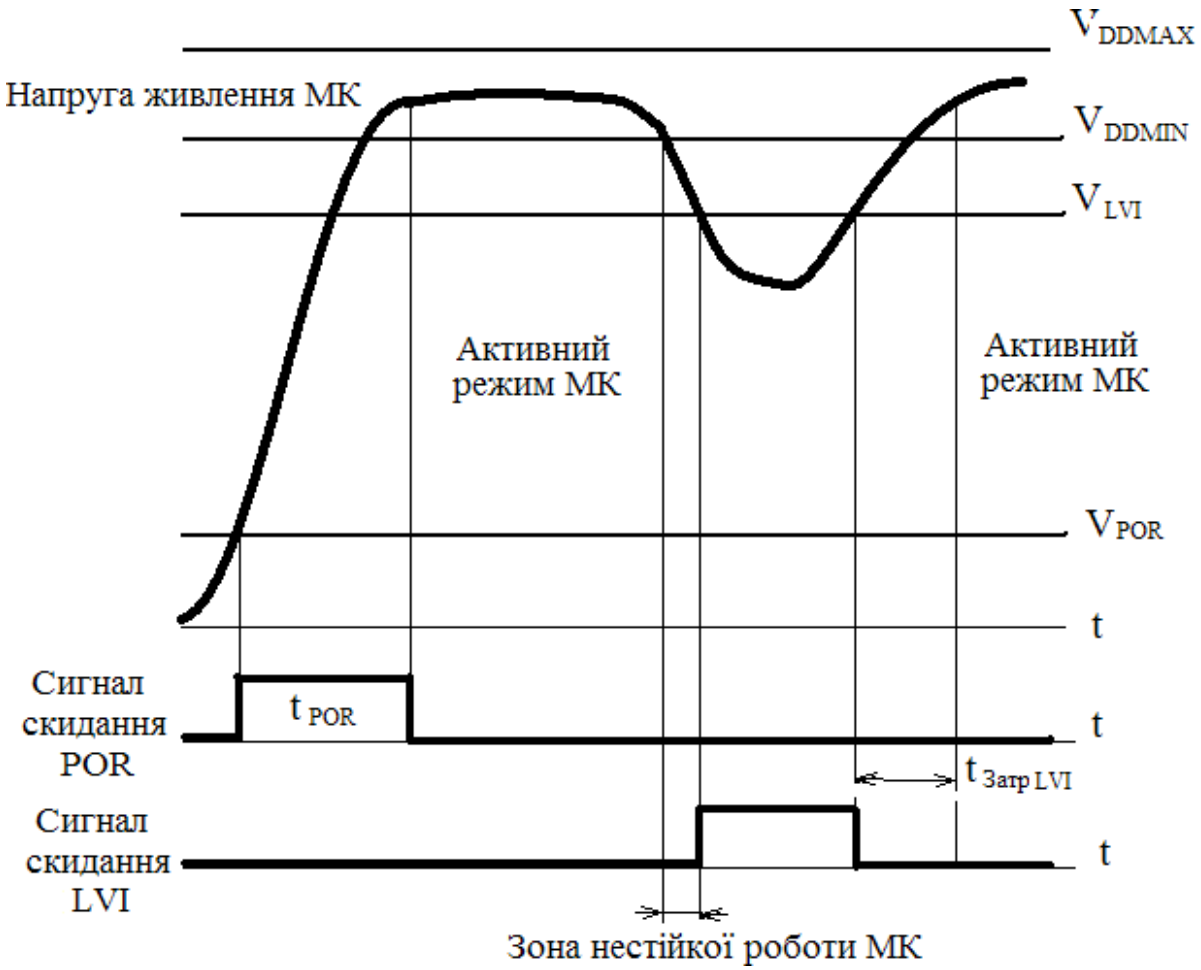


Рисунок 2.13 – Скидання МК за сигналам POR та LVI

2.9 Апаратні і програмні рішення по підвищенню надійності роботи МК

На відміну від персональних комп'ютерів, які при збігу певних обставин можуть виконати позаштатні операції й «зависнути», вбудовані МП-системи керування позбавлені цього недоліку. Прикладна програма керування, записана в пам’ять МК, повинна забезпечувати формування адекватних вихідних впливів при будь-яких комбінаціях вхідних сигналів. Однак у результаті електромагнітних перешкод передбачений

розроблювачем хід виконання прикладної програми може бути порушений. Саме в цьому випадку неможливо гарантувати правильну роботу МП-системи. Усі сучасні МК передбачають можливість відновлення правильного ходу обчислювального процесу при подібного роду відмовах. Для цього використається модуль сторожового таймера (COP або Watchdog). Основним елементом модуля сторожового таймера є багаторозрядний лічильник. При скиданні МК лічильник обнуляется. Після переходу МК в активний режим роботи лічильник починає збільшувати код незалежно від виконуваної програми. Якщо код лічильника досягає максимального зазначеного в технічному описі МК значення, то генерується сигнал внутрішнього скидання й МК починає виконання програми керування спочатку. Для виключення події скидання з переповнення сторожового таймера прикладна програма керування повинна періодично скидати лічильник. Операція скидання лічильника сторожового таймера звичайно виконується за допомогою запису зазначеного коду в один з регістрів спеціальних функцій. Тоді при нормальному, передбаченому розроблювачем порядку виконання прикладної програми переповнення лічильника сторожового таймера не настає й він не впливає на роботу МП-системи. Однак якщо хід виконання прикладної програми був порушений, то велика ймовірність, що лічильник не буде скинутий вчасно. Тоді відбудеться скидання з переповнення сторожового таймера, й нормальний хід обчислювального процесу буде відновлений.

Модулі сторожових таймерів різних МК мають наступні особливості роботи:

- У деяких МК вектори зовнішнього скидання й скидання з переповнення сторожового таймера об'єднані. Це не дозволяє розрізняти причину скидання на програмному рівні й утрудняє написання прикладних програм. Більше досконалі МК мають або різні вектори скидання, або відзначають подію скидання з переповнення сторожового таймера установкою спеціального біта.
- Частина МК автоматично припиняє роботу сторожового таймера при переході МК в один з режимів зниженого енергоспоживання, коли прикладна програма не виконується. Однак у деяких МК сторожовий таймер не припиняє роботу в режимі очікування. Тоді необхідно передбачати періодичний вихід МК із режиму очікування для скидання лічильника сторожового таймера. Це, з одного боку, не дуже зручно, тому що вимагає додаткової уваги програміста. Однак, з іншого боку, надає додаткову можливість відновлення працездатності МП-системи, якщо сигнал виходу МК із режиму очікування порушений.

Крім апаратно-програмних засобів сторожового таймера деякі МК оснащені апаратними засобами генерації сигналу внутрішнього скидання у випадку, якщо через електромагнітні перешкоди в лічильнику адреси була

сформована адреса фізично не існуючої комірки пам'яті або з пам'яті прочитаний код неіснуючої операції.

3 КОНТРОЛЕРИ Intel MCS-51

Однокристальний мікроконтролер (ОМК) 8051 є представником сімейства Intel MCS-51, що у даний час займає лідуючу позицію за кількістю різновидів і кількості компаній, що випускають його модифікації.

На сьогоднішній день існує понад 200 модифікацій мікроконтролерів, що випускаються майже 20 компаніями. Основними виробниками клонів MCS-51 є фірми Philips, Siemens, Intel, Atmel, Dallas Semiconductor, Temic Semiconductor, Oki, AMD, Gold Star, Winbond і ряд інших.

Архітектуру 8051 використовують у своїх розробках провідні виробники електронного устаткування, створюючи MSC-контролери (Mixed Signal Microcontroller). Наприклад: Analog Devices, Burr-Brown, Texas Instruments й інші.

Родоначальником архітектури MCS-51 є фірма Intel, що у 1980 році випустила мікроконтролер 8051 на базі HMOS-технології. З погляду технології 8051 був для свого часу досить складним виробом - у кристалі було використано 128 тисяч транзисторів, що в 4 рази перевищувало кількість транзисторів у 16-розрядному мікропроцесорі 8086.

Основними елементами базової архітектури є:

- 8-розрядий АЛП на основі акумуляторної архітектури;
- апаратна реалізація множення;
- віконна адресація чотирьох банків регістрів по 8 байт у кожному;
- резидентний ОЗП даних 128 байт, що забезпечує гнучке керування його ресурсами за рахунок поділу на зони регістрів, бітів і вільної зони;
- простір регістрів спеціальних функцій 128 байт;
- механізм обробки бітових даних, розташованих у резидентному ОЗП даних і в зоні регістрів спеціальних функцій;
- резидентна пам'ять програм 4Кх8, виконана на OTP чи EPROM;
- розвинута система команд із роздільним звертанням до пам'яті програм і даних;
- два шестнадцятирозрядних лічильники-таймери;
- контролер послідовного порту;
- чотири 8-розрядних паралельних порти введення/виводу, кожен біт яких можна настроїти на введення чи на вивід;
- контролер обробки переривань з п'ятьма джерелами запитів, два з яких зовнішні;
- убудований тактовий генератор.

ОМК може працювати в режимах мікроконтролера і мікропроцесора. У режимі мікропроцесора можливе приєднання зовнішньої пам'яті програм до 64КБ і даних до 64КБ.

Напруга живлення 5В. Струм споживання 18 мА (Кмоп-технологія), 150…200 мА (n-Моп-технологія). Максимальна тактова частота 12 МГц, що забезпечує час виконання основних команд за 1…2 мкс; тільки множення і ділення виконується за 4 мкс.

Основні модифікації базової моделі 8051:

n-моп технологія:

8031АН (КР1816ВЕ31) - без резидентної пам'яті програм;

8051АН (КР1816ВЕ51) – OTP;

8751Н (КМ1816ВЕ751) – EPROM (з ультрафіолетовим стиранням інформації);

КМОП технологія:

80С31 (КР1830ВЕ31);

80С51 (КР1830ВЕ51);

87С571(КМ1830ВЕ75).

Інші модифікації будуть розглянуті нижче.

3.1 Структурна організація Intel 8051

Архітектура ОМК припускає логічний і структурний поділ пам'яті команд і пам'яті даних. Такий поділ є ознакою гарвардської архітектури. Основною перевагою гарвардської архітектури є **роздільні магістралі** для звертання до пам'яті програм і пам'яті даних, що підвищує швидкодії. Однак у 8051 роздільними є тільки внутрішні шини адреси команд і даних, а дані і команди передаються загальною шиною.

Застосування гарвардського принципу в 8051 дозволяє мати різну організацію пам'яті команд і даних, більш ефективно використовувати формати команд, що розрізняються при звертанні до команд і даних використовуваними способами адресації. Остання обставина призводить до скорочення розміру пам'яті, необхідного для збереження програм.

При роботі з зовнішніми пристроями (пам'яттю, периферійними БІС) звертання до пам'яті програм і даних виконується за загальною магістраллю. Однак у процесі звертання до зовнішніх пристроїв формуються різні сигнали для керування пам'ятю програм і даних, що спрощує їхню апаратну реалізацію і збільшує обсяг сумарного адресного простору.

Основу структурної схеми ОМК (рис. 3.1) утворить внутрішня двонапрямна 8-бітна шина даних, що зв'язує між собою всі основні вузли і пристрої: резидентну пам'ять, центральний процесор, що складається з

АЛП, акумулятора ACC, регістра-розширювача акумулятора B, слова стану процесора PSW і пристрою керування, порти введення/виводу.

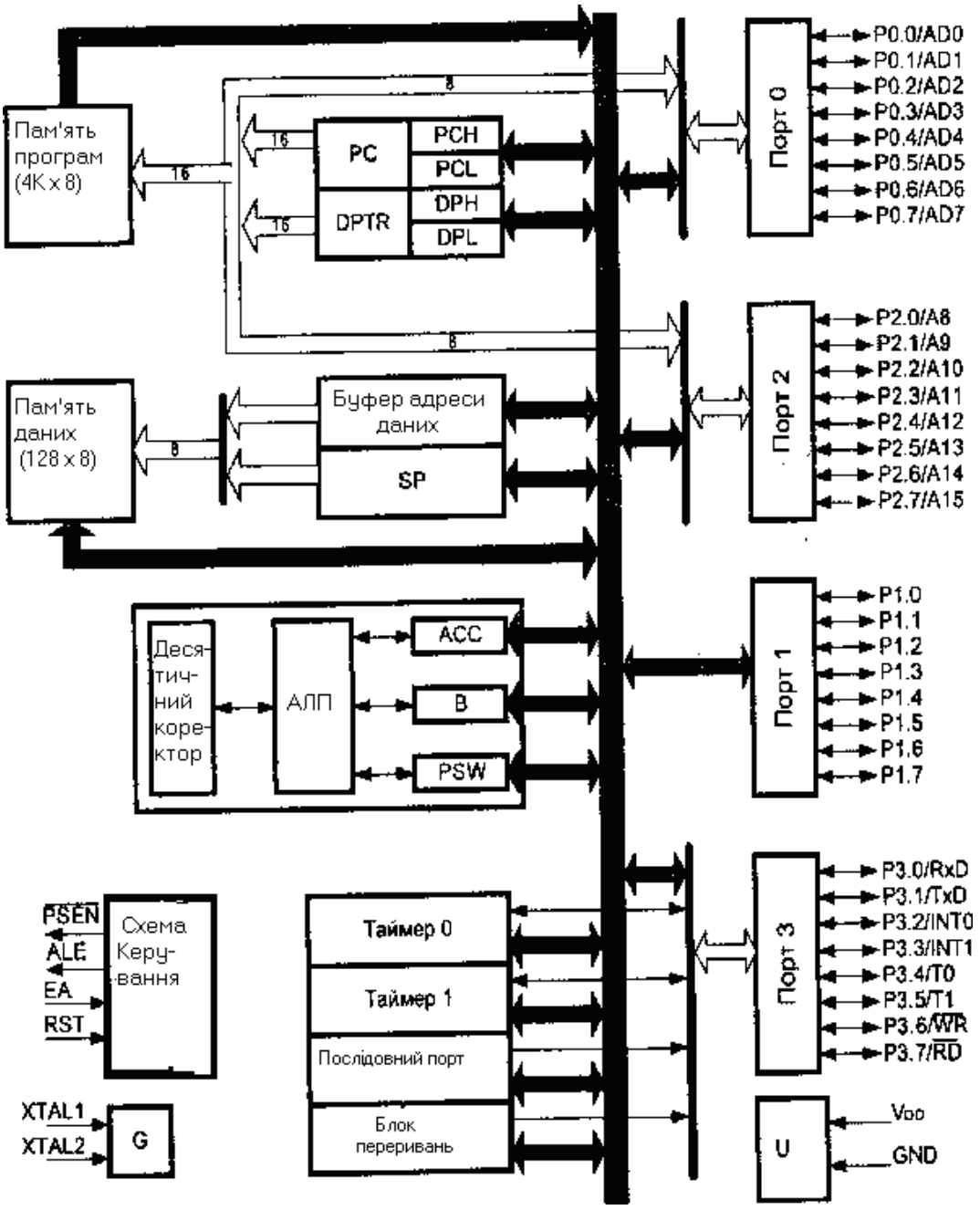


Рисунок 3.1 – Структурна схема Intel 8051

3.1.1 Арифметично-логічний пристрій

Восьмирозрядний АЛП може виконувати арифметичні операції додавання, вирахування, множення і ділення; логічні операції «І», «АБО», «виключаюче АБО», операції над бітовими даними, а також операції циклічного зсуву, скидання, інвертування і т.п. До складу АЛП входять: програмно недоступні регістри Т1 і Т2, призначені для тимчасового збереження операндів, схема десяткової корекції і схема формування ознак.

Поряд з байтовими операціями АЛП може обробляти бітові змінні. Окремі програмно-доступні біти можуть бути встановлені, скинуті, інвертовані, передані, перевірені та використані в логічних операціях.

Акумулятор ACC є джерелом і/або приймачем операндів при виконанні арифметичних і логічних операцій.

Регістр-розширювач У використовується в операціях множення і розподілу.

Регістр PSW містить ознаки результатів виконання операцій і біти (прапори), необхідні для керування ресурсами ОМК, SP-покажчик стека.

Резидентна пам'ять. Пам'ять програм і пам'ять даних, розміщені на кристалі ОМК, фізично і логічно розподілені, мають різні механізми адресації, працюють під керуванням різних сигналів і виконують різні функції (рис. 3.2).

Внутрішня (резидентна) пам'ять програм має ємність 4 Кб і призначена для збереження команд, констант, керуючих слів ініціалізації, таблиць перекодування вхідних і вихідних змінних і т. п. Резидентна пам'ять програм має 16-бітну шину адреси, керування якої виконується лічильником команд (PC) чи регістром-покажчиком даних (DPTR). Останній виконує функції базового регістра при непрямій адресації чи використовується в командах, які оперують з таблицями.

Внутрішня (резидентна) пам'ять даних (ОЗП) призначена для збереження змінних у процесі виконання прикладної програми і має організацію 128x8.

До адресного простору РПД примикають адреси регістрів спеціальних функцій (РСФ), що перелічені в таблиці 3.1.

Таблиця 3.1 – Блок регістрів спеціальних функцій

Символ	**Найменування**	**Адреса**
1	*2*	*3*
* ACC	Акумулятор	0E0H
* B	Регістр-розширювач акумулятора	0F0H
* PSW	Слово стану програми	0D0H
SP	Регістр-покажчик стека	81H
DPTR	Регістр-покажчик даних (DPH)	83H
	(DPL)	82H
* P0	Порт 0	80H
* P1	Порт 1	90H
* P2	Порт 2	0A0H
* P3	Порт 3	0B0H
* IP	Регістр пріоритетів	0B8H
* IE	Регістр маски переривань	0A8H
TMOD	Регістр режиму таймера/лічильника	89H
* TCON	Регістр керування/статус таймера	88H

TH0	Таймер 0 (старший байт)	8CH
TL0	Таймер 0 (молодший байт)	8AH

Продовження таблиці 3.1

1	*2*	*3*
TH1	Таймер 1 (старший байт)	8DH
TL1	Таймер 1 (молодший байт)	8BH
* SCON	Регістр керування прийомопередавачем	98H
SBUF	Буфер прийомопередавача	99H
PCON	Регістр керування потужністю	87H
Примітка. Регістри, імена яких відзначені знаком (*), допускають адресацію окремих біт.		

Поряд з байтовим адресним простором внутрішня пам'ять даних містить бітовий простір, що починається у бітовій зоні (адреси 20…2Fh), а продовжується у зоні регістрів спеціальних функцій (адреси 80…FFh). Звертання до нього можливе тільки з використанням бітових команд. Нульовому біту комірки 20h (20.0) відповідає адреса 0 бітового простору, а старшому біту регістра спеціальних функцій FFh- адреса 255 (FFh).

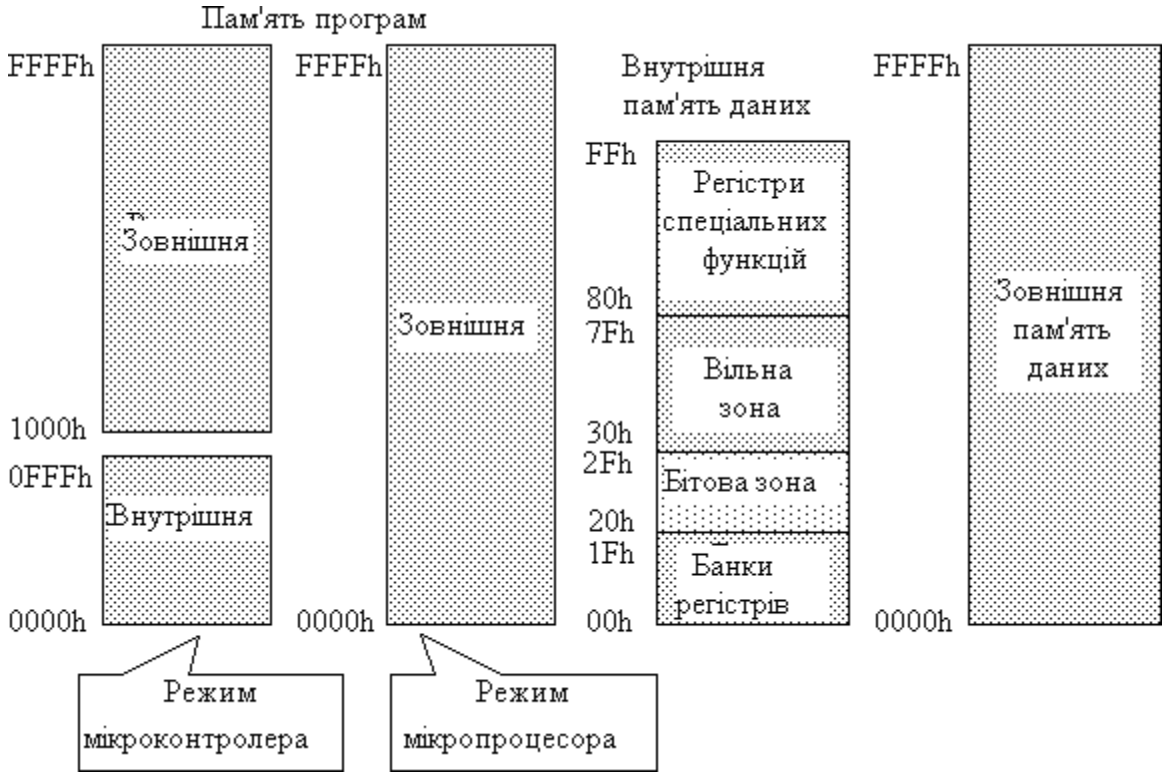


Рисунок 3.2 – Будова пам’яті мікропроцесорної системи на основі ОМК

У режимі мікроконтролера використовується тільки резидентна пам'ять програм і даних. Пам'ять програм розташовується за адресами 0000…0FFFh, а даних – з 00 до 07Fh. Регістри спеціальних функцій мають

об'єднаний з внутрішнім ОЗП адресний простір і розташовані за адресами 80…0FFh. Особливості використання регістрів спеціальних функцій розглянуті в підрозділі 3.4.

3.1.3 Зовнішня пам'ять

Пам'ять програм і пам'ять даних може бути поширена до 64 Кбайт шляхом підключення зовнішніх БІС. Варіанти організації пам'яті в МК системі пояснює рисунок 3.2. Зв'язок ОМК із зовнішньою пам'яттю забезпечують паралельні порти P0, P2, P3, що працюють у системному режимі. Особливості цього режиму будуть розглянуті далі.

Пам'ять програм повинна розташовуватися в діапазоні адрес 1000…0FFFFh, тобто доповнювати внутрішній ПЗП ОМК до 64 Кбайт (0FFFFh). Реалізація цього режиму вимагає, щоб на вході EA був установлений рівень логічної одиниці.

Можливий режим відключення резидентної пам'яті програм (сигнал на лінії EA дорівнює 0). У цьому випадку зовнішня пам'ять програм має обсяг 64 КБ, починаючи з нульової адреси. Використання цього режиму доцільно при налагодженні програмного забезпечення.

Простір зовнішньої пам'яті даних не залежить від розмірів резидентного ОЗП і може займати діапазон адрес з 0000 до 0FFFFh.

Особливості використання пам'яті програм і даних будуть розглянуті далі.

3.1.4 Пристрій керування і синхронізації

Пристрій керування **(**керуючий автомат**)** функціонує під дією зовнішніх сигналів і внутрішніх, формованих у процесі виконання команд. Пристрій керування працює за жорстким часовим циклом, обумовленим частотою внутрішнього чи зовнішнього генератора.

До зовнішніх сигналів відносяться:

XTAL1, **XTAL2** – входи зовнішнього елемента, що задає частоту;

PSEN – читання зовнішньої пам'яті програм;

ALE – сигнал дозволу фіксації адреси зовнішньої пам'яті;

EA – блокування внутрішньої пам'яті програм;

RST – сигнал загального скидання;

сигнали, формовані на виводах портів.

Частина сигналів на схемі (рис. 3.1) не показана: **PROG** – сигнал програмування, **VPP** – напруга програмування, **VPD** – вивід резервного живлення пам'яті від зовнішнього джерела.

Частота внутрішнього генератора може бути задана підключенням до його виводів XTAL1 і XTAL2 кварцового резонатора (рис. 3.3, а), LC-ланцюжка (рис. 3.3, б).

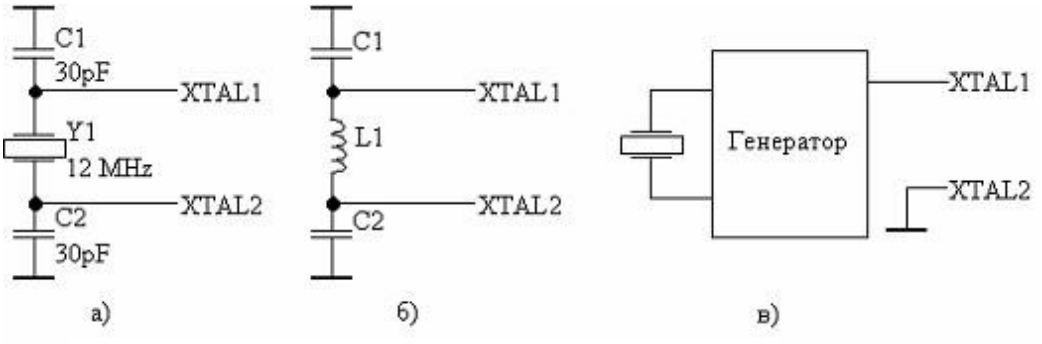


Рисунок 3.3 – Формування зовнішнього тактового сигналу

У випадку LC-ланцюжка частота синхросигналу визначається за формулою

$$f = \frac{1}{2\pi\sqrt{LC}}.$$

При використанні зовнішнього тактового генератора його вихід підключається до входу XTAL2, а вивід XTAL1 підключається до загальної шини (рис. 3.3, в).

Пристрій керування на основі сигналів синхронізації формує машинний цикл фіксованої тривалості, яка дорівнює 12 періодам тактового генератора чи 6 станам керуючого автомата S1-S6 (рис. 3.4). Кожен стан містить дві фази P1, P2. У фазі P1, як правило, виконується операція АЛП, а у фазі P2 здійснюється межрегістрова передача.

Рисунок 3.4 – Структура машинного циклу

При частоті генератора 12 МГц машинний цикл становить 1 мкс. Двічі за один машинний цикл формується сигнал ALE.

Машинний цикл служить в основному для внутрішнього мікропрограмного керування. Цикл виконання кожної команди складається з одного, двох чи чотирьох машинних циклів.

Початкове установлення (апаратне скидання) робиться з метою запуску чи перезапуску мікроконтролера після подачі на нього напруги живлення. Скидання здійснюється подачею на вхід RESET високого рівня й утримання його протягом не менш двох машинних циклів. Цей сигнал

може подаватися асинхронно стосовно внутрішньої частоти ОМК. Вхід RESET постійно опитується мікроконтролером у момент S5P2 кожного машинного циклу. Після сигналу скидання порти введення-виводу знаходяться у незмінному стані протягом 19 періодів тактування, після чого в проміжку між 19- і 31-м тактами переводяться в початковий «одиничний» стан. При цьому сигналі ALE і PSEN знаходяться в неактивному (високому) стані.

За сигналом скидання основні системні регістри мікроконтролера скидаються в нульовий стан, у покажчик стека встановлюється число 7, у регістри-засувки портів P0…P3 записується код 0FFh, налаштовуючи P0…P2 на вивід даних, а P3 - на виконання системних функцій. **На зміст внутрішньої пам'яті даних сигнал RESET не впливає**. При включенні живлення комірки резидентної пам'яті даних встановлюються в довільний стан.

Тривалість сигналу RESET повинна бути не менше часу, необхідного для запуску внутрішнього генератора плюс 2 машинних цикли. Час установлення генератора залежить від частоти синхронізації і характеристик кварцового резонатора. При частоті 12 МГц воно звичайно становить майже 1 мкс.

Для автоматичного рестарту мікросхеми після подачі напруги живлення до виводу RESET необхідно підключити RC-ланцюжок (рис. 3.5), що забезпечує необхідну затримку, яка дозволяє генерувати одиночний імпульс скидання.

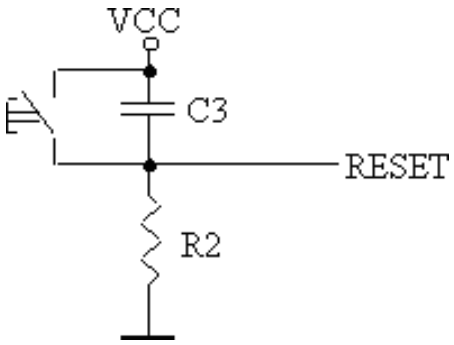


Рисунок 3.5 – Схема формування сигналу скидання

Після зняття сигналу RESET проходить від 1 до 2 тактових періодів до їхньої активізації. При цьому мікроконтролер починає виконувати програму з адреси 0000h внутрішньої або зовнішньої пам'яті програм (залежно від рівня сигналу EA). Повторне формування сигналу RESET виконується кнопкою КН.

3.2 Програмна модель ОМК

ОМК виконує дії над 8-розрядними операндами. Програмна модель центрального процесора 8051 (рис. 3.6) містить шість регістрів. Усі регістри, крім програмного лічильника (лічильника команд) PC, є частиною об'єднаного адресного простору ОЗП даних. Для звертання до них можуть бути використані як символьні імена регістрів - ACC, B, PSW, SP, DPH, DPL, так і їхні абсолютні адреси - 0E0h, 0F0h, 0D0h, 81h, 82h, 83h відповідно.

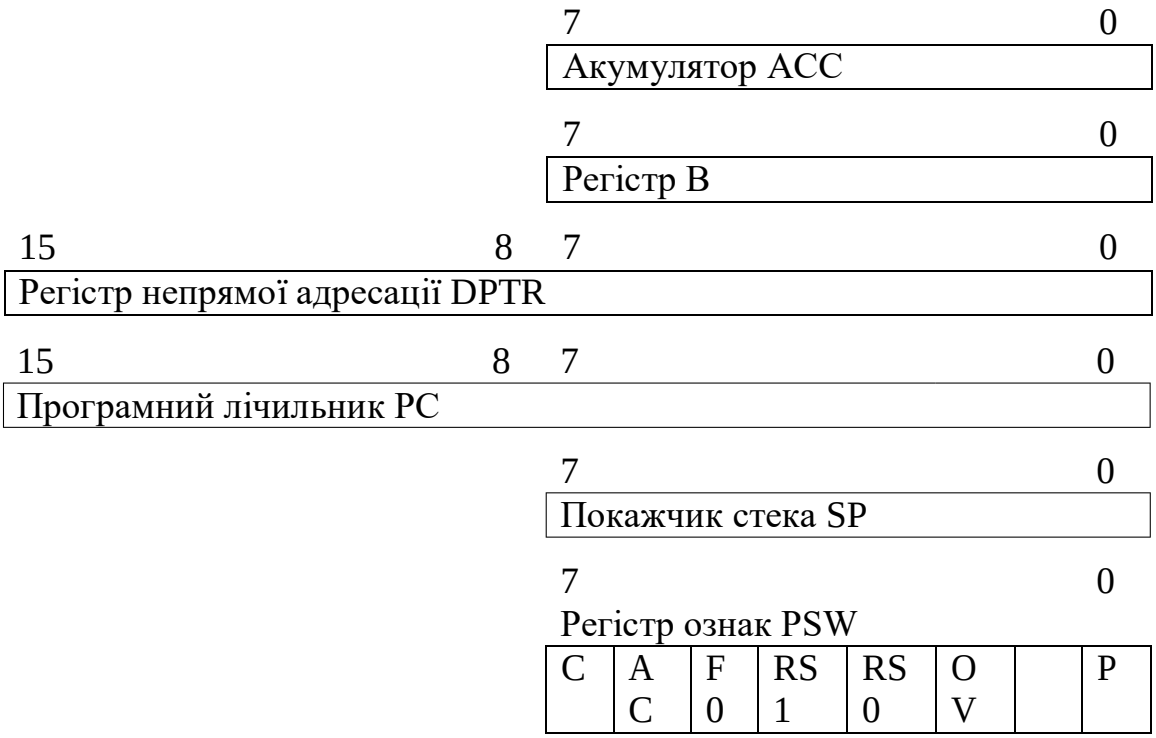


Рисунок 3.6 – Програмна модель центрального процесора

Акумулятор (ACC) є джерелом операнда і місцем фіксації результату при виконанні арифметичних, логічних операцій і низки операцій передачі даних. Крім того тільки з використанням акумулятора можуть бути виконані операції зсуву, перевірка на нуль, формування прапора паритету і т. п.

Регістр B - додатковий 8-розрядний регістр. Використовується в операціях множення і ділення. В інших командах може інтерпретуватися як регістр загального призначення.

Регістр-покажчик даних DPTR - 16-розрядний регістр (DPH - старший байт, DPL - молодший байт). Використовується як регістр непрямої адресації при звертанні до пам'яті команд і зовнішньої пам'яті даних. Система команд передбачає можливість звертання до окремих байтів (DPH чи DPL) регістра DPTR, що дозволяє використовувати ці регістри для збереження проміжних результатів чи обчислень як джерело одного з операндів.

Програмний лічильник PC - 16-розрядний регістр, містить адресу поточної команди або адресу операнда, використовуваного в поточній команді. Після скидання мікроконтролера (МК) програмний лічильник автоматично встановлюється в «0». Тому в МК із процесорним ядром MCS-51 будь-яка програма користувача повинна починатися з адреси 0000h. Команди звертання до PC відсутні. Однак він програмно доступний за допомогою спеціальних процедур.

Покажчик стека SP - 8-розрядний регістр, містить адресу вершини стека. Архітектура процесорного ядра MCS-51 припускає розміщення ділянки стекової пам'яті тільки в ділянці внутрішнього ОЗП даних. Глибина стека обмежена обсягом резидентного ОЗП. При скиданні ОМК покажчик стека встановлюється в положення 07h, призначаючи ділянку стека до першого банку регістрів. Тому при використанні стека після скидання необхідно задавати необхідне значення SP. Звичайно вершина стека розміщається в старших адресах вільної зони.

При запису в стек (команда PUSH) зміст покажчика SP збільшується на 1, а при читанні (команда POP) – зменшується на 1. Покажчик стека апаратно збільшується на 2 при виконанні команд виклику підпрограми (CALL) і звертанні до підпрограми обробки переривань і зменшується на 2 при виконанні команд повернення RET, RETI. У стек при цьому відповідно записується чи витягається вміст PC.

Слово стану процесора ССП (PSW). При виконанні більшості команд в АЛП формується низка ознак операції (прапорів), що фіксуються в регістрі ССП. У таблиці 3.2 наводиться перелік прапорів ССП, даються їхні символічні імена й описуються умови їхнього формування.

Таблиця 3.2 – Формат слова стану процесора PSW

Символ	**Позиція**	**Ім'я і призначення**
1	*2*	*3*
C	PSW.7	Прапор переносу зі старшого розряду. Установлюється і скидається при виконанні арифметичних і логічних операцій апаратно. Можлива зміна значення з програмними засобами
AC	PSW.6	Прапор допоміжного переносу з третього розряду в четвертий. Установлюється і скидається тільки апаратними засобами при виконанні команд додавання і вирахування
F0	PSW.5	Прапор 0. Може бути встановлений, скинутий чи перевірений програмою як прапор, встановлений користувачем
RS1 RS0	PSW.4 PSW.3	Вибір банку регістрів. Установлюється і скидається програмою для вибору робочого банку регістрів (див. примітку)

OV	PSW.2	Прапор арифметичного переповнення. Установлюється і скидається апаратно при виконанні арифметичних операцій
-	PSW.1	Не використовується

Продовження таблиці 3.2

1	*2*	*3*			
P	PSW.0	Прапор паритету. Установлюється і скидається в кожному циклі команди тільки апаратно. Набуває значення 1 при парному числі одиничних бітів в акумуляторі			
Примітка		RS1	RS0	Банк	Межі адрес
		0	0	0	00…07H
		0	1	1	08…0FH
		1	0	2	10…17H
		1	1	3	18…1FH

Найбільше «активним» прапором ССП є прапор переносу, що бере участь і модифікується в процесі виконання багатьох операцій, включаючи додавання, вирахування і зсув. Крім того, прапор переносу (C) виконує функції «булевого аккумулятора» у командах, що маніпулюють з бітами.

Прапор арифметичного переповнення (OV) фіксує перевищення довжини розрядної сітки при операціях над цілими числами зі знаком.

Прапор AC використовується в процедурі корекції результату при виконанні двійково-десяткового додавання.

Прапор паритету P забезпечує контроль переданої інформації. Значення біта паритету змінюється при будь-якій операції, у якій ACC є приймачем результату. При прийомі інформації виконується повторне формування P, а потім порівнюються біти паритету, обчислених при прийомі і передачі інформації.

АЛП не керує прапорами селекції банку регістрів (RS0, RS1) і F0. Їхні значення цілком визначаються прикладною програмою.

У регістрі ознак відсутні прапори нульового Z і негативного N результатів. Ознака нульового стану акумулятора формується апаратно **у всіх** командах, у яких акумулятор є приймачем результату. У регістрі ознак дорівненість нулю не відображається. Команди умовного переходу за ознакою **Z** (JZ і JNZ) варто використовувати після виконання команд, у яких результат операції записується в акумулятор.

Ознака знака результату фіксується в старшому біті ACC, який доступний за допомогою бітових команд умовних переходів (JB, JNB).

Призначення і функції регістрів спеціальних функцій докладно розглянуті в розділі 3.4.

РПД розподілена на 3 зони, кожна з якої має свої функціональні особливості.

Зона регістрів загального призначення (РЗП) (таблиця 3.3) складається з 4-х банків, розташованих з адреси 00h до 1Fh. Кожен банк складається з 8 комірок РПД. Після скидання, активним є нульовий банк. Номер банку встановлюється у слові стану процесора. Перевагою цієї зони є можливість використання прямої регістрової адресації. Команди роботи з регістрами однобайтові і виконуються за мінімальний час (1 мкс). Звичайно в цій зоні розташовують регістри тимчасового збереження інформації. Виключення складають регістри R0, R1 кожного банку, що можуть використовуватися як регістри непрямої адресації при звертанні до РПД чи як індексні регістри при звертанні до зовнішньої пам'яті даних.

Таблиця 3.3 – Структура внутрішньої пам'яті даних

Адреса	Бітові адреси								Адреса	Бітові адреси								Регістр
7Fh 30h	Комірки резидентного ОЗП (вільна зона)								0FFh	Комірки резидентного ОЗП								
2Fh	7F	7E	7D	7C	7B	7A	79	78	0F0h	F7	F6	F5	F4	F3	F2	F1	F0	B
2Eh	77	76	75	74	73	72	71	70										
2Dh	6F	6E	6D	6C	6B	6A	69	68	0E0h	E7	E6	E5	E4	E3	E2	E1	E0	A
2Ch	67	66	65	64	63	62	61	60										
2Bh	5F	5E	5D	5C	5B	5A	59	58	0D0h	D7	D6	D5	D4	D3	D2	D1	D0	PSW
2Ah	57	56	55	54	53	52	51	50										
29h	4F	4E	4D	4C	4B	4A	49	48	0B8h	-	-	-	BC	BB	BA	B9	B8	IP
28h	47	46	45	44	43	42	41	40										
27h	3F	3E	3D	3C	3B	3A	39	38	0B0h	B7	B6	B5	B4	B3	B2	B1	B0	P3
26h	37	36	35	34	33	32	31	30										
25h	2F	2E	2D	2C	2B	2A	29	28	0A8h	AF	-	-	AC	AB	AA	A9	A8	IE
24h	27	26	25	24	23	22	21	20										
23h	1F	1E	1D	1C	1B	1A	19	18	0A7h	A7	A6	A5	A4	A3	A2	A1	A0	P2
22h	17	16	15	14	13	12	11	10	99h									SBUF
21h	0F	0E	0D	0C	0B	0A	09	08	98h	9F	9E	9D	9C	9B	9A	99	98	SCON
20h	07	06	05	04	03	02	01	00										
1Fh	Банк 3 (R7-R0)								90h	97	96	95	94	93	92	91	90	P1
									8Dh									TH1
18h									8Ch									TH0
17h	Банк 2 (R7-R0)								8Bh									TL1
									8Ah									TL0
10h									89h									TMOD

0Fh	Банк 1 (R7-R0)	88h	8F	8E	8D	8C	8B	8A	89	88	TCON
		87h									PCON
08h		83h									DPH
07h	Банк 0 (R7-R0)	82h									DPL
		81h									SP
00h		80h	87	86	85	84	83	82	81	80	P0

Зона регістрів спеціальних функцій (РСФ) розташована з адреси 80h до 0FFh. У ній знаходяться основні регістри даних і керування.

Звертання до РСФ можливо тільки з указівкою прямої адреси. У 8051 використовується тільки частина адрес зони РСФ. Невикористані адреси зарезервовані для розвитку ОМК даної серії.

8051 надає великі можливості для роботи з бітовими змінними, котрі мають бути розташовані в **бітовій зоні.** Бітовий простір починається у бітовій зоні РПД (адреси 20…2Fh), а закінчується в зоні регістрів спеціальних функцій. Звертання до бітів можливо тільки з використанням команд прямої бітової адресації. Адресний простір бітової зони і РПД мають різну метрику.

Вільна зона - особливостей за способами адресації не має.

Структура адресного простору РПД подана в таблиці 3.3. РПД являє собою єдиний адресний простір, кожна зона якого має визначені особливості. У той час є можливість звертатися до будь-якої комірки пам'яті РПД, використовуючи пряму чи непряму адресацію.

Наявність зон, що мають різні функціональні можливості, при грамотному їхньому використанні дозволяє мінімізувати як довжину коду програм, так і час їхнього виконання.

3.3 Система команд Intel 8051

8051 має гнучку систему команд, орієнтовану, головним чином, на реалізацію контролерних функцій. Характеристики системи команд визначаються форматами даних і команд.

Формати даних. Тип подання даних: числові, логічні, бітові. Числові дані можуть бути подані в двійковому і двійково-десятковому кодах у цілочисленному форматі. Обробка двійкових даних може виконуватися в додатковому коді. Довжина формату байт, два байти, біт. В операціях обміну тетрадами використовується додаткове подання інформації - напівбайтами (4 біти).

Формати команд. Команди одно-, дво-, безадресні. Довжина команди – байт, два байти, три байти. При звертанні до РПД можуть використовуватися прямий регістровий, прямий, непрямий, безпосередній, стековий способи адресації. Зовнішня пам'ять даних допускає застосування тільки непрямої й індексної адресації. При роботі з пам'яттю команд

можливе застосування непрямого, індексного і відносного способів адресації.

Система команд МК51 містить 111 базових команд, що зручно поділити за функціональною ознакою на п'ять груп: команди передачі даних, арифметичних операцій, логічних операцій, передачі керування, операцій з бітами. Система команд наведена в додатку А, де використовуються наступні позначення:

#d - безпосередній операнд;

ad - адреса РПД;

ad16 - адреса пам'яті чи команд зовнішньої пам'яті даних;

Rn - регістр поточного банку;

Ri - нульовий або перший регістр поточного банку

bit – прямої адреса біта,

rel - 8-розрядний зсув у додатковому коді для виконання відносних переходів. Величина зсуву дозволяє передавати керування в межах –128…+127 байт щодо адреси наступної команди;

@ - ознака непрямої адресації.

Розглянемо узагальнені характеристики основних команд.

До **команд передачі даних** (пересилань) відносяться:

MOV – пересилання при роботі з РПД;

MOVX – пересилання із зовнішньою пам'яттю даних (ВПД);

MOVC – пересилання із пам'яттю команд;

MOV A, (Rn, ad, @Ri, #d);

MOV Rn, (A, ad, #d);

MOV ad, (Rn, A, @Ri, #d);

MOV (A, Rn, ad), #d;

MOV @Ri, (ad, Rn, #d);

MOV ad1, ad2;

MOV DPTR,#d16.

Команди цієї групи не модифікують ознаки результату за винятком команди завантаження PSW і пересилань, у яких приймачем результату є акумулятор А. У цьому випадку встановлюється біт паритету й апаратно формується ознака рівності «0» – Z=1, яку можна використовувати для виконання команд умовного переходу JZ (JNZ).

XCH A, (Rn, ad, @Ri) - обмін умістом А і (Rn, @Ri, ad);

XCHD A, @Ri - обмін умістом молодших тетрад байтових операндів;

SWAP - обмін умістом тетрад в акумуляторі;

PUSH ad – запис у стек;

POP ad – читання зі стека;

Недоліком цієї групи є **відсутність** команд типу MOV Rn1, Rn2 і MOV Rn, @Ri.

Звертання до зовнішньої пам'яті команд ЗПК і ЗПД здійснюється через регістр-покажчик DPTR. Звертання до регістра можливе за допомогою

команди MOV DPTR, #d16, старший (DPH) і молодший (DPL) байти цього регістра доступні через зону РСФ.

MOVX A, @DPTR;

MOVX @DPTR, A;

MOVX @Ri, A;

MOVX A, @Ri - сторінково-непрямий метод доступу: Зсув усередині сторінок задає вміст Ri, а номер сторінки формується на виході порту Р2. На основі цього способу можна організувати індексну адресацію.

Для читання пам'яті команд використовується:

MOVC A, @A+DPTR; (A):=@(A+DPTR),

де в А знаходиться цілий беззнаковий зсув.

Для верифікації пам'яті команда передбачена:

MOVC A, @A+PC; {PC:=PC+1; (A):=((A+PC))},

де РС – програмний лічильник.

Арифметичні операції:

ADD A, (Rn, ad, @Ri, #d) – додавання;

ADDC A, (Rn, ad, @Ri, #d) - додавання з переносом;

DA A - команда двійково-десяткової корекції при додаванні;

SUBB A, (Rn, ad, @Ri, #d) - вирахування із заємом. Це єдина команда вирахування. Перед її використанням необхідно контролювати значення біта С.

Команди додавання, вирахування формують прапори З, АС, OV, Р.

INC(DEC) (A, Rn, @Ri, ad) - інкремент (декремент);

INC DPTR.

Якщо необхідний декремент, то його реалізація можлива тільки з використанням команд декремента регістрів DPL (молодший байт DPTR), і DPH (старший байт DPTR).

Команди інкремента і декремента прапори не встановлюють.

MUL AB – беззнакове множення 8х8 (B)(A):=(A)*(B);

DIV AB – беззнакове ділення 8/8 (A)(B):=A/B.

При множенні старший байт результату записується в регістр-розширювач В, а молодший - в А. Якщо вміст А>256, то формується прапор арифметичного переповнення OV. Біт С завжди скидається.

При діленні частка записується в А, а залишок - у В. Прапори переносу С і арифметичного переповнення OV скидаються. При діленні на 0 установлюється прапор OV.

Логічні команди*:*

ANL A, (Rn, @Ri, #d, ad); - кон’юнкція;

ANL ad, (A, #d);

Структура команд ORL (диз'юнкція), XRL (сума за модулем 2) аналогічна попередній.

CLR A - очищення акумулятора;

CPL A - інвертування акумулятора;

У цих логічних командах ознаки не формуються, за винятком біта паритету P, якщо результат операції записується в акумулятор.

RR(RL) A – правий (лівий) циклічний зсув;

RRC(RLC) A – правий (лівий) циклічний зсув через біт C;

При зсувах формуються ознаки P і C.

Бітові команди:

При виконанні бітових команд біт C виконує функції акумулятора.

Команди пересилання біт:

MOV C, bit;

MOV bit, C;

SETB bit(C) - установка біта C (чи прямоадресованого біта);

CLR bit (C) - очищення біта C (чи прямоадресованого біта);

CPL C(bit) - інверсія біта C (чи прямоадресованого біта);

ANL C, bit - кон'юнкція;

ORL C, bit – диз'юнкція;

ANL C, /bit;

ORL C, /bit.

В останніх двох командах знак «/» указує на те, що як значення використовується логічне заперечення адресованого біта, однак сам біт джерела при цьому не змінюється.

При роботі з бітами використовується тільки пряма адресація.

До **команд передачі керування** відносяться команди, що забезпечують умовні і безумовні переходи, виклик підпрограм і повернення з них, команда порожньої операції NOP.

Перехід за всим адресним простором забезпечує довгий безумовний перехід LJMP ad16 і звертання до підпрограми LCALL ad16.

Перехід у межах пам'яті програм розміром 2048 байт виконується за допомогою безумовного абсолютного переходу AJMP ad11 і звертання до підпрограми ACALL ad11. Такі команди містять тільки 11 молодших бітів адреси переходу і мають довжину два байти.

Команда JMP @A+DPTR використовує непряму адресацію, при якій адреса переходу визначається підсумовуванням умісту DPTR і беззнакового зсуву, що знаходиться в A. Ця команда дозволяє виконувати множинне розгалуження за адресою, невідомою в момент написання програми, модифікуючи вміст A чи DPTR.

Команда SJMP rel виконує безумовний короткий відносний перехід.

При програмуванні на асемблері досить указати тільки тип переходу JMP чи CALL. У залежності від довжини переходу конкретна команда безумовного переходу вибирається компілятором автоматично.

RET - повернення з підпрограми;

RETI - повернення з підпрограми обробки переривання.

Умовні переходи подані двома групами команд байтовими і бітовими. До байтових команд відносяться:

JZ(JNZ) rel – перехід при нульовому (ненульовому) результаті в A. Може бути реалізований після будь-якої операції, результат якої записується в акумулятор;

JC(JNC) rel – перехід за значенням біта C;

DJNZ Rn(ad), rel. Виконується декремент умісту регістра чи комірки РПД. Якщо результат операції не дорівнює «0», виконується перехід, інакше - наступна команда.

CJNE A, (ad, #d), rel;

CJNE (Rn, @Ri), #d, rel. Виконується вирахування з першого операнда другого; якщо результат операції не дорівнює нулю, виконується перехід, інакше - наступна команда. Уміст регістрів, що беруть участь в операціях, не змінюється.

При порівнянні формується біт C = 0, якщо перший операнд більше другого і C = 1 при протилежному результаті.

Бітові переходи виконуються за значенням біта, адреса якого визначена у форматі команди JB(JNB) bit, rel.

У команді JBC bit, rel перехід виконується при одиничному значенні біта, адреса якого визначена у команді, після чого біт скидається в «0». Ця команда зручна при реалізації семафорів.

Семафор являє собою прапор, що інформує про стан зв'язаного з ним ресурсу: «1» – вільний, «0» – зайнятий. Захоплення ресурсу допускається тільки в тому випадку, якщо він вільний. Потім семафор повинний бути переведений у стан «Зайнято». Під ресурсом розуміється всякий об'єкт, що розподіляється системою.

Наприклад, дані від різних джерел повинні бути записані в роздільну пам'ять.

```
WAIT:
        JBC bit, OK; Перевірка прапора
        SJMP WAIT; Ресурс зайнятий
OK:
        ;Ресурс вільний
```

При програмуванні в об'єктному коді адреса переходу обчислюється як:

rel := адреса переходу - адреса команди переходу + довжина команди переходу;

В ОМК відсутні команди введення-виводу. Звертання до зовнішніх пристроїв здійснюється як до комірок зовнішньої пам'яті даних. Можливі формати команд наведені на рисунку 3.7.

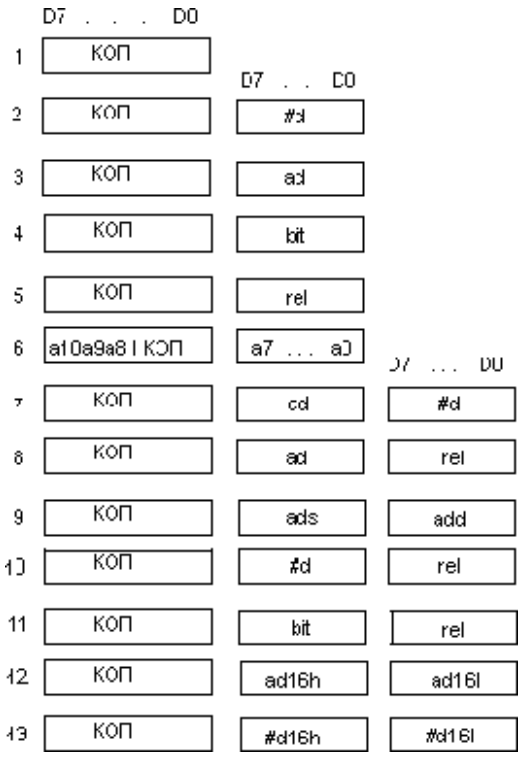


Рисунок 3.7 – Можливі формати команд

3.4 Периферійні пристрої ОМК

3.4.1 Порти введення/виводу

Порти P0…P3 призначені для уведення або виводу інформації і забезпечують обмін із зовнішніми пристроями: пам'яттю програм і даних, контролерами різного призначення, периферійними пристроями.

З боку центрального процесора ці порти подаються у виді відповідних регістрів спеціальних функцій P0…P3. У ці регістри процесор може записувати вихідну інформацію або зчитувати з них вхідні дані.

Кожний з портів складається з 8-розрядного регістра-засувки (Ргз), вихідних транзисторів Т1 і Т2, вхідних ланцюгів і схем D1, D2 з відкритим стоком. Схемотехніка портів трохи відрізняється, тому що вони виконують різні функції. Як приклад розглянемо структуру одного біта порту P0 (рис. 3.8).

При читанні входу порту P0.X дані через кон'юнктор D1 передаються на внутрішню шину даних (ВШД), що організована як «монтажне АБО». Якщо у відповідний розряд Ргз записана «1», то дані з входу порту без пошкодження передаються до приймача. Якщо біт РгЗ=0, то у відповідний разряд приймача буде записаний «0» незалежно від значення вхідного сигналу.

При записі інформації в порт дані записуються в РгЗ і виводяться через транзистор Т1 на вихід порту.

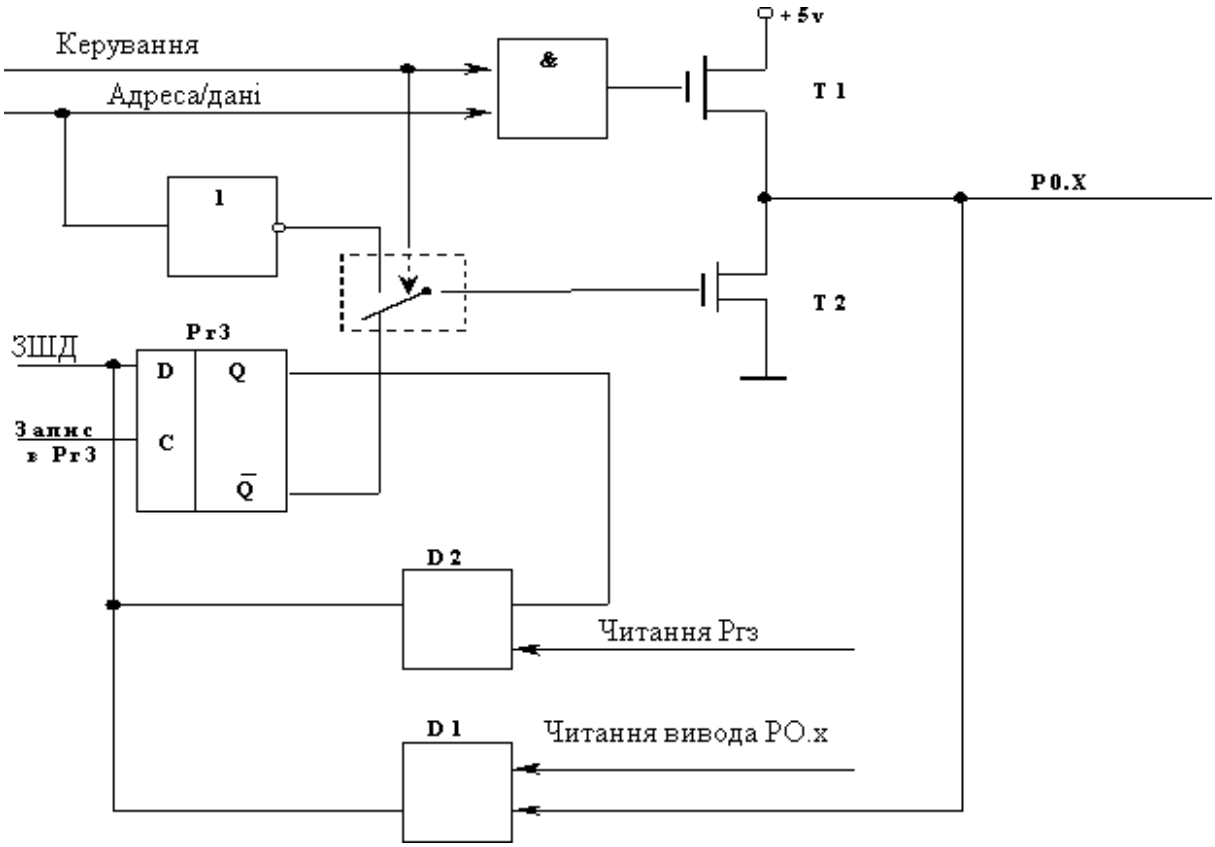


Рисунок 3.8 – Структура порта P0

Звертання до портів можливе тільки за прямою адресою. Усі розряди портів знаходяться в бітовому просторі.

При роботі в режимі мікроконтролера порти односпрямовані. Кожен біт порту може бути налагоджений як на введення (PX.Y=1), так і на вивід. За скиданням в Ргз записується код 0FFh, тобто порти P0…P2 налаштовуються на введення, а P3 – на виконання системних функцій. У режимі мікропроцесора порт P0 – двонаправлений.

Для читання даних з порту або регістра засувки використовуються різні команди. Читання Ргз здійснюється командами «читання – модифікація – запис», при виконанні яких команда зчитує стан Ргз, при необхідності модифікує отримане значення і записує результат назад у Ргз:

ORL Px, A ;
XRL Px, A
ANL Px, A
INC Px
DEC Px
DJNZ Px, rel

Для читання окремих бітів використовуються команди:

JBC Px.y, SETB Px.y
CPL Px.y, MOV Px.y, C

В усіх випадках, коли операндом і регістром призначення є порт чи біт порту, команди зчитують інформацію з виходів РгЗ, а не з зовнішніх

контактів виводів порту. При читанні регістра-засувки дані передаються через D2.

Якщо порт є операндом-джерелом, то дані зчитуються із зовнішніх виводів порту. Читання із зовнішніх виводів порту може бути виконано командами:

MOV (A, @Ri, Rn, ad), Px

ADD A, Px

Читання інформації з виходів Ргз, а не з зовнішніх контактів, дозволяє виключити в ряді випадків неправильну інтерпретацію рівня напруги на виводах порту. Наприклад, при з'єднанні виводу порту з базою n-p-n транзистора і подачі лог. 1, транзистор відкривається і читання стану зовнішнього виводу покаже лог. 0, тому що на контакті порту присутнія напруга база-емітер транзистора (0,7 В). Читання ж Ргз підтвердить подачу лог. 1.

Запис даних у порт здійснюється командами:

MOV Px, (A, Rn, @R0, #d).

Порти P0, P1, P2, P3 залежно від особливості застосування можуть реалізувати різні функції (табл. 3.4).

Таблиця 3.4 – Специфікація ліній портів

Ім'я порту	**Альтернативна функція**	**Адреса регістрів даних**
P0	Мультиплексування шини адреса/дані (при використанні зовнішньої пам'яті або даних програм)	80h
P1	-	90h
P2	Шина адреси – старший байт (при використанні зовнішньої пам'яті або даних програм)	A0h
P3	Кожна лінія специфікована індивідуально: P3.0 – вхід приймача послідовного порту RXD; P3.1 – вихід передавача послідовного порту TXD; P3.2 – вхід зовнішнього запиту INT0; P3.3 – вхід зовнішнього запиту INT1; P3.4 – зовнішній вхід таймера T0; P3.5 - зовнішній вхід таймера T1; P3.6 – сигнал шини керування WR; P3.7 – сигнал шини керування RD. Примітка. Активним рівнем сигналів WR, RD є «0». Використовуються при адресації зовнішньої пам'яті даних	B0h

При роботі з зовнішньою пам'яттю порт P0 є системним портом, через який у режимі з поділом часу передаються молодший байт адреси і

дані. Поява молодшого байта адреси супроводжується сигналом ALE, за яким він повинний бути зафіксований у зовнішньому регістрі.

У системному режимі керування обміном через порт P0 виконується командами MOVX, MOVC.

Якщо P0 використовується як порт загального призначення, то до виходу порту повинний бути приєднаний зовнішній «підтягуючий» резистор від джерела живлення +5 В, тому що в цьому режимі транзистор T1 закривається.

На відміну від P0 порти P1…P3 мають убудоване навантаження, що виключає необхідність у зовнішніх резисторах. Порт P2 виводить старший байт адреси при роботі в системному режимі. Особливістю порту P2 є можливість мультиплексування на вихід умісту Ргз або старшого байта адреси. При роботі в режимі адресної шини вміст Ргз зберігається і надходить на вивод порту в тих машинних циклах, коли немає звертання до зовнішньої пам'яті. При звертанні до зовнішньої пам'яті на вихід порту виводиться інформація з регістра адреси DPTR чи з програмного лічильника PC за командами MOVC і MOVX. Виключення складають команди MOVX A, @Ri, і MOVX @Ri, A, при виконанні яких на виході знаходиться вміст регістра-засувки.

Навантажувальна здатність P0 - два входи ТТЛ, в інших - один.

P1 - порт загального призначення й особливостей не має.

P3 - при записі в Ргз «1» виконує системні функції. Специфікація P3 наведена в таблиці 3.4. Якщо в біт Ргз порту P3 записаний «0», то на виході порту фіксується «0».

При використанні порту P3 необхідно враховувати функціональні особливості кожного входу. T1, T0, INT1, INT0 – вхідні лінії. Якщо не використовуються зовнішні переривання і/або лічильники, то призначення цих ліній визначає користувач. RD, WR - вихідні лінії, сигнали на яких формуються тільки при виконанні команд MOVX, і так далі.

Для організації взаємодії з зовнішньою пам'яттю і пристроями уведення/виводу використовуються **три типи магістральних циклів**: читання з зовнішньої пам'яті даних, запис у зовнішню пам'ять даних, читання із зовнішньої пам'яті програм.

Звертання до зовнішньої пам'яті даних вимагає двох машинних циклів. Тому другий імпульс ALE усередині машинного циклу при звертанні до зовнішньої пам'яті даних не генерується.

Часова діаграма циклу читання із зовнішньої пам’яті даних показана на рисунку 3.9.

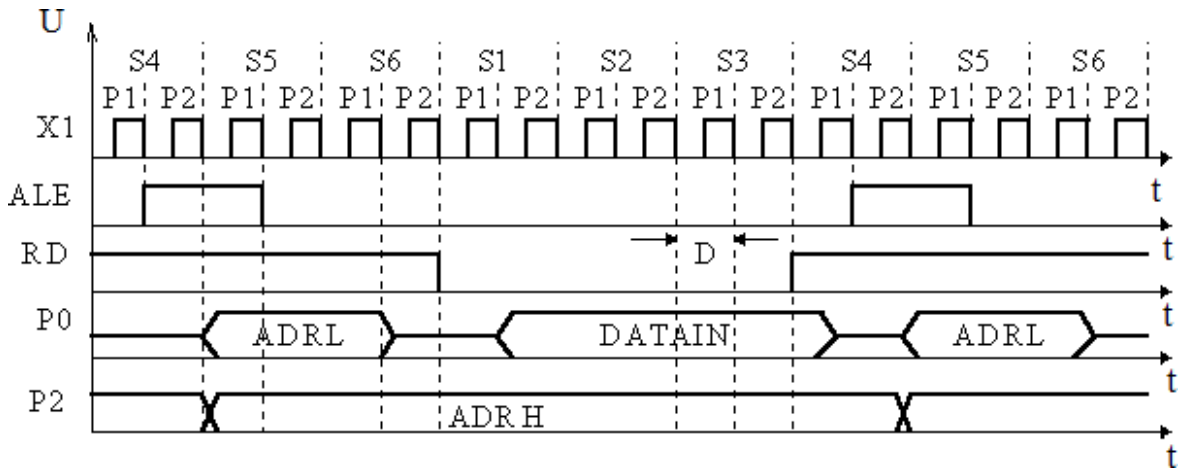


Рисунок 3.9 – Цикл читання із зовнішньої пам’яті даних

За зрізом сигналу ALE молодший байт адреси, що надходить з виводів порту P0, повинний бути зафіксований у зовнішньому регістрі-засувці. Старший байт адреси зберігається на виводах порту P2 протягом усього циклу і не вимагає зовнішньої фіксації. Дані зчитуються у фазі P1 стану S3 поточного машинного циклу. До цього моменту вони повинні бути свідомо встановлені на шині даних.

Часова діаграма циклу запису у зовнішню пам’ять даних показана на рисунку 3.10.

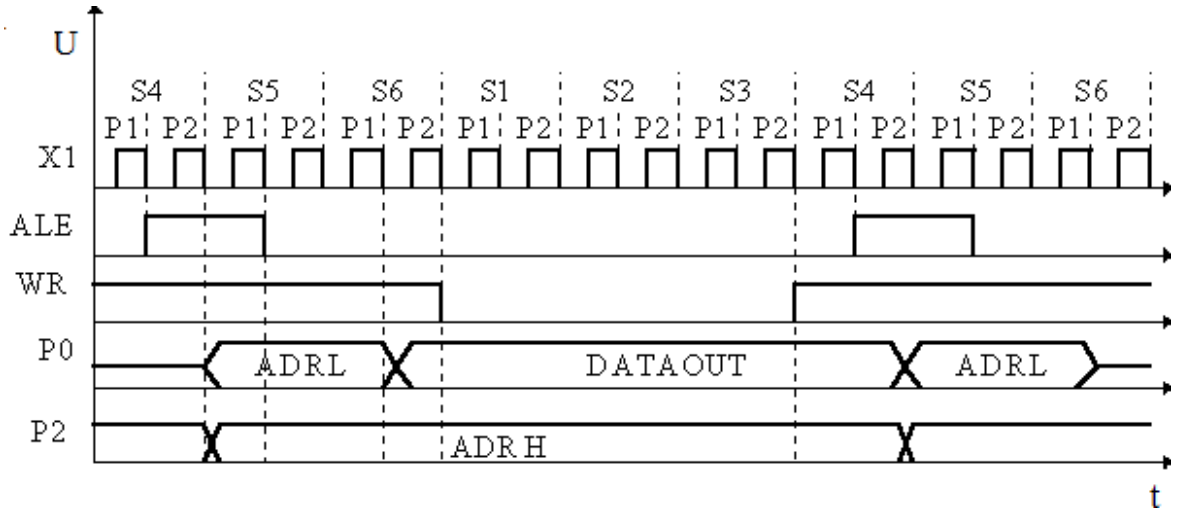


Рисунок 3.10 – Цикл запису у зовнішню пам’ять даних

При звертанні до внутрішньої пам'яті даних сигнали RD і WR не виробляються.

Для звертання до зовнішньої пам'яті програм використовується механізм, відмінний від механізму звертання до пам'яті даних. Для читання пам'яті програм використовується сигнал PSEN. При цьому сигнали ALE і PSEN виробляються двічі за машинний цикл.

Часова діаграма циклу читання із зовнішньої пам’яті програм показана на рисунку 3.11.

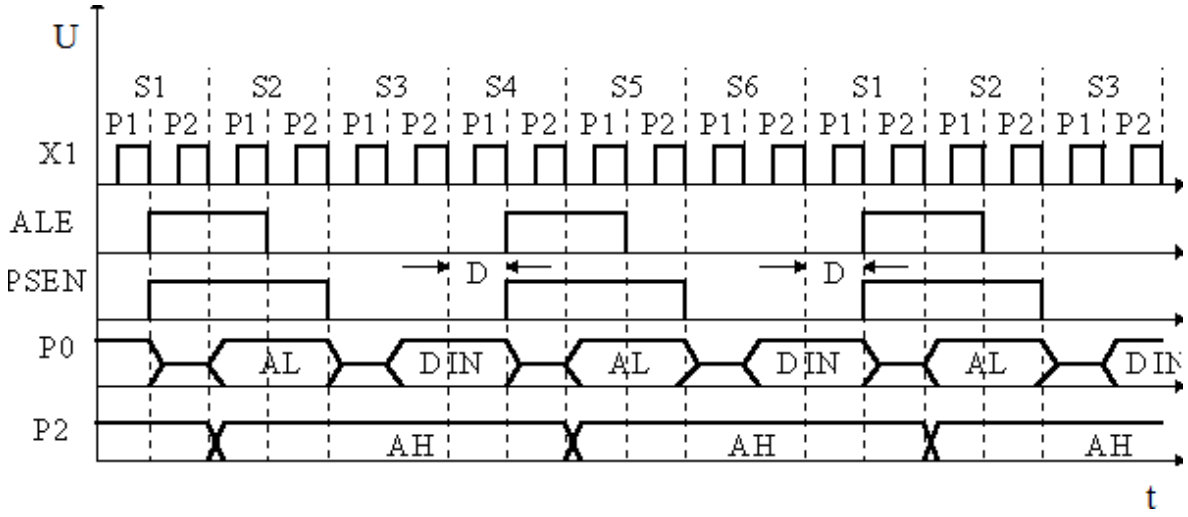


Рисунок 3.11 – Цикл читання із зовнішньої пам’яті програм

Так само, як і у випадку пам'яті даних, молодший байт адреси повинний бути зафіксований у зовнішньому регістрі за зрізом сигналу ALE. Байти програми зчитуються у фазі P1 станів S4 і S1 поточного машинного циклу. При звертанні до внутрішньої пам'яті програм сигнал PSEN не генерується.

При використанні сигналу ALE як джерела синхросигналів необхідно враховувати особливості його формування для різних машинних циклів.

Робота із зовнішньою пам'яттю програм дозволена лише при низькому рівні на вході EA. При цьому мікроконтролер вважає, що вся пам'ять програм розташована в зовнішній пам'яті. При високому рівні сигналу на цьому вході мікроконтролер реалізує цикл звертання до зовнішньої пам'яті програм тільки у випадку, якщо необхідна адреса виходить за межі внутрішнього сегмента пам'яті програм. Якщо в мікросхемі «прошитий» біт таємності, то рівень сигналу на вході EA фіксується при скиданні мікроконтролера у внутрішній засувці і подальша його зміна не впливає на роботу системи.

3.4.2 Лічильники/таймери

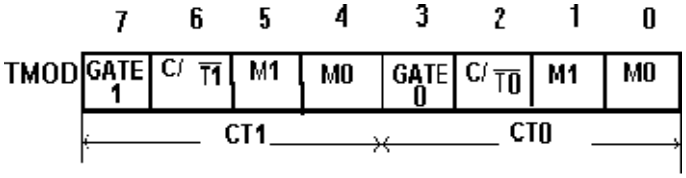
Лічильник/таймер (CT) призначений для обробки зовнішніх і внутрішніх подій, формування програмно-керованих тимчасових затримок, виконання часозавдавальних функцій.

До складу ОМК входять два 16-розрядних підсумовуючих CT. Лічильник складається з регістрів лічильника TLi (молодший байт), THi (старший байт), логіки керування вхідними сигналами і тригера переповнення TFi. Біт TFi установлюється при переході лічильника зі стану

«усі 1» у стан «усі 0». Біт TFi розташовується в бітовому просторі і доступний за прямою адресою. Звертання до TLi, THi здійснюється роздільно за прямими адресами.

Керування роботою лічильника виконується за допомогою регістра режиму роботи TMOD і регістра керування-статусу TCON.

Формат регістра TMOD показаний на рисунку 3.12.



M1	M0	режим
0	0	0
0	1	1
1	0	2
1	1	3

GATE1, GATE0 – керування способом запуску СТ;
$C/\overline{T0}$, $C/\overline{T1}$ – визначають роботу СТ як лічильника ($C/\overline{T}$ =1) або таймера ($C/\overline{T}$ =0);
M1, M0 - задають режими роботи

Рисунок 3.12 – Формат регістра режиму лічильника/таймера TMOD

Спосіб запуску СТ може бути апаратний чи програмний. Якщо GATEi =1, то реалізується апаратний запуск, при якому дозвіл рахування подається на вхід INTi порту P3. Попередньо повинний бути встановлений біт запуску лічильника TRi у регістрі TCON. (i – номер лічильника-таймера). При програмному запуску GATEi =0, початок рахунка задається установкою біта TRi =1.

У режимі таймера СТ працює від внутрішнього генератора з частотою OSC/12.

При роботі в режимі лічильника вміст СТ інкрементується під впливом переходу з «1» у «0» зовнішнього сигналу, який подається на відповідні входи T0, T1 порту P3. Інкремент виконується після аналізу стану «0» і «1» на вході Ti, тому накладаються певні обмеження на параметри преутвореного сигналу: рівень «0» і «1» повинний продовжуватися не менше OSC/12. Максимальна преутворена зовнішня частота - OSC/24.

Лічильник/таймер може бути налаштований на один з 4 режимів. Режими 0,1,2 однакові для обох лічильників, і в цих режимах вони цілком незалежні. Робота CT0 і CT1 у режимі 3 різна. При цьому установка CT0 у режим 3 впливає на режими роботи CT1.

Режим 0: режим 13-розрядного регістра, що складається з THi і 5 молодших розрядів TLi. Структура показана на рисунку 3.13.

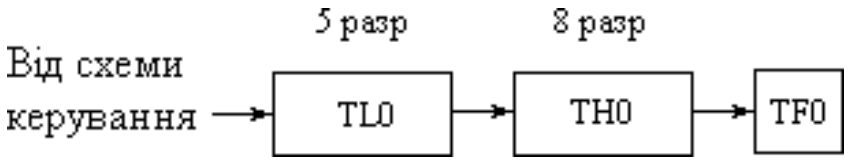


*Рисунок 3.13 – Структура лічильника/таймера у режимі **0***

Режим 1: режим 16-розрядного регістра, що складається з THi і TLi. Структура показана на рисунку 3.14.

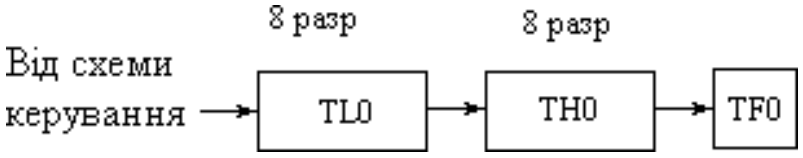


Рисунок 3.14 – Структура лічильника/таймера у режимі 1

Режим 2: у цьому режимі СТ являє собою пристрій на основі 8-розрядного регістра TLi. При кожному переповненні Tli, крім установки прапора Tfi, відбувається автоматичне завантаження вмісту THi у TLi. Структура показана на рисунку 3.15.

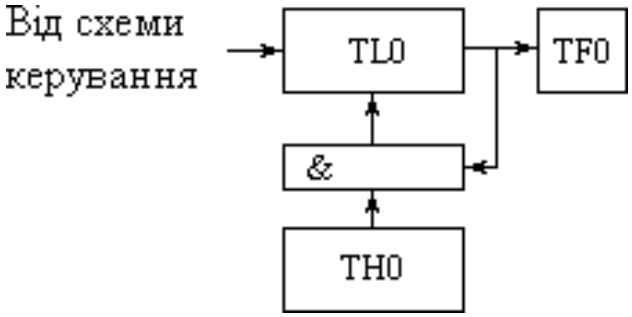


*Рисунок 3.15 – Структура лічильника/таймера у режимі **2***

Необхідний коефіцієнт розподілу повинний записуватися одночасно в THi і TLi.

Принцип роботи СТ у режимах 0,1,2 пояснює рисунок 3.16.

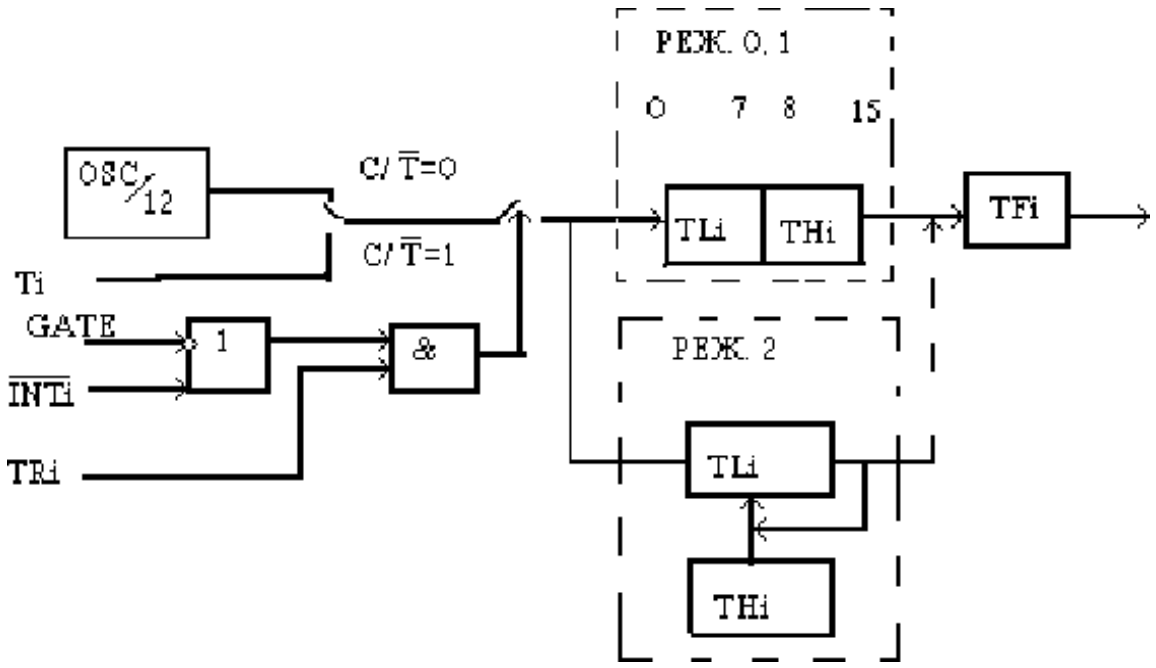


Рисунок 3.16 – Принцип роботи лічильника/таймера в режимах «0, 1, 2»

Режим 3: у цьому режимі лічильник 0 функціонує як 2 незалежних лічильники, а лічильник 1 заблокований і просто зберігає свій код (виконує функції регістра). При цьому можна настроїти лічильник 1 на інші режими.

Принцип роботи СТ у режимі 3 пояснює рисунок 3.17.

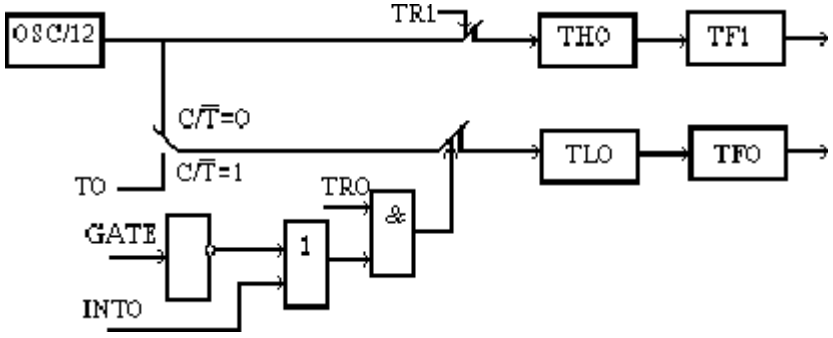


Рисунок 3.17 – Принцип роботи СТ у режимі 3

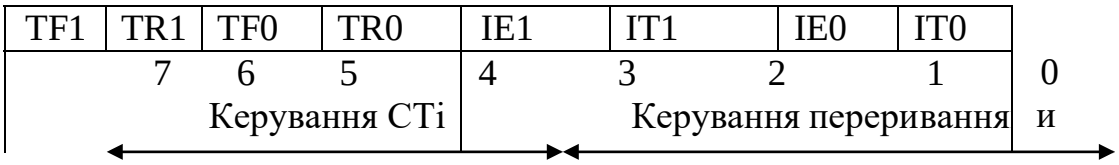
Режим 3 використовується, якщо необхідно збільшити кількість діючих лічильників до трьох. При роботі в цьому режимі СТ0 розподіляється на два 8-розрядних лічильники, сформованих на базі регістрів TH0, TL0. Лічильник на базі TH0 може працювати тільки в режимі таймера, керується бітом TR1 і формує сигнал переповнення TF1. Лічильник на базі TL0 особливостей у керуванні не має .

Установка СТ0 у режим 3 позбавляє СТ1 біта включення TR1. Тому СТ1 у режимах 0, 1, 2 при GATE=0 завжди включений і при переповненні в

режимах 0 і 1 СТ1 обнуляється, а в режимі 2 перезавантажується, не встановлюючи прапора, якщо СТ0 знаходиться в режимі 3.

СТ1 апаратно зв'язаний із блоком синхронізації послідовного порту. Тому в режимах 0, 1, 2 при переповненні СТ1 завжди виробляє імпульс синхронізації послідовного порту.

Регістр керування і стану TCON (рис. 3.18) призначений для формування сигналів запуску СТ і збереження прапорів їхнього переповнення, завдання типу сприйманого запиту і збереження прапорів зовнішніх переривань.



TFi – *прапор переповнення СТ. Установлюється апаратно при переповненні лічильника/таймера, скидається програмно. При обслуговуванні переривання скидання виконується апаратно.*
TRi – біти керування СТ. Установлюється/скидається програмно для пуску/зупинки;
ITi – біт керування видом сигналу запиту переривання. При Imi=1 програмується переривання за зрізом імпульсу запиту, при ITi=0 – за низьким рівнем (i – номер входу запиту зовнішнього переривання);
IEi – прапор запиту переривання. Установлюються апаратно від зовнішніх запитів на входах INTi. Скидаються апаратно, якщо переривання викликане зрізом імпульсу запиту чи програмно. Можливе програмне керування цим прапором.

Рисунок 3.18 – Формат регістра TCON

Для настроювання лічильника/таймера на необхідний режим треба:

1) задати необхідний коефіцієнт перерахування К в регістри Тні, TLi. Якщо лічильник підсумовуючий, то К= 65535 – С +1, де С – необхідний коефіцієнт розподілу;
2) задати режим роботи в слові TMOD;
3) при програмному введенні-виводу замаскувати відповідні переривання від лічильника, а при використанні переривання, їх дозволити;
4) при програмному запуску біт дозволу TRi в слові TCON встановлюється в момент початку роботи СТ, а при апаратному – на стадії ініціалізації.

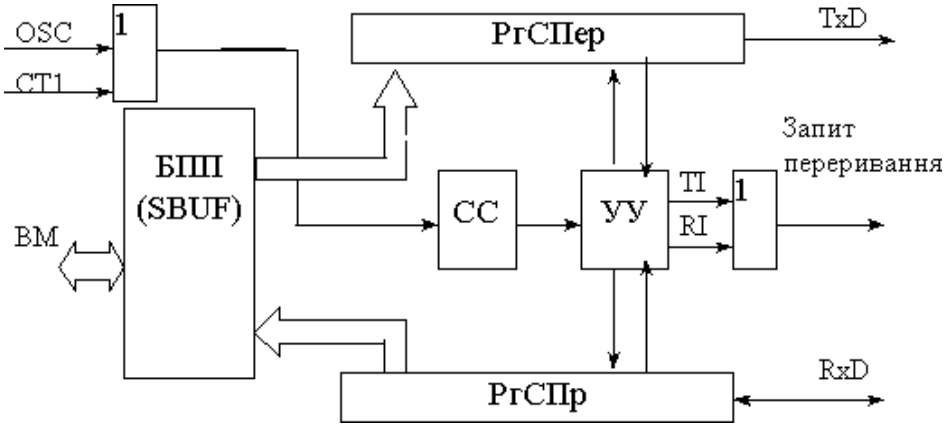
На базі СТ можна організувати перетворювачі частоти, часового інтервалу, періоду в код, генератори та формувачі сигналів. Однак необхідно враховувати, що TFi апаратно недоступно, тому вихідні сигнали варто формувати на виходах паралельних портів.

3.4.3 Послідовний порт

Послідовний порт здійснює прийом/передачу інформації в послідовному коді, молодшими бітами вперед, у дуплексному (одночасний прийом і передача інформації) чи напівдуплексному режимах.

До складу послідовного каналу входять приймаючі і передавальні зсувні регістри, спеціальний програмно-доступний буфер SBUF, регістр керування SCON і логіка керування каналом.

Спрощена структура порту подана на рисунку 3.19.



РгСПер – регістр зсуву передавача;
СС – схема синхронізації; УУ – пристрій керування;
РгСПр – регістр зсуву приймача; БПП – буфер прийомопередавача;
TI, RI – прапор готовності, відповідно передавача і приймача;
ВМ – внутрішня магістраль; Tx, Rx – вхід приймача, вихід передавача;
OSC – вихід системного генератора;
CT1 – сигнал переповнення лічильника-таймера 1

Рисунок 3.19 – Структура послідовного порта

Програмна модель порту включає регістр керування і стану послідовного порту SCON, буфер прийомопередавача SBUF і біт SMOD регістра PCON.

Послідовний канал може працювати в наступних чотирьох режимах:

Режим «0» - синхронний обмін у напівдуплексному режимі з частотою OSC/12. Формат посилки – 8 біт. Дані приймаються і передаються через вхід Rx, а частота синхронізації формується на виході Tx. У цьому режимі порт працює як 8-розрядний зсувний регістр.

Режим «1» - асинхронний обмін, десятибітовий кадр, що складається зі стартового (нуль), стопового (одиниця) бітів і 8-розрядного символу. Швидкість прийому і передачі визначається частотою переповнення лічильника CT1. У залежності від стану біта SMOD регістра PCON частота, що надходить на вхід схеми синхронізації послідовного каналу з виходу

СТ1, може змінюватися в два рази. При прийомі стоп-біт заноситься у біт RB8 регістра SCON.

Режим «2» - асинхронний 11-бітовий кадр. У порівнянні з режимом «1» доданий програмно встановлюваний дев'ятий біт. Переданий дев'ятий біт даних набуває значення біта ТВ8 з регістра керування SCON. Цей біт може бути програмно встановлений у «0» чи «1». Зокрема ТВ8 можна привласнити значення біта паритету Р з регістра PSW для підвищення вірогідності переданої інформації. При прийомі дев'ятий біт даних надходить у біт RB8 регістра SCON. Швидкість передачі фіксована і визначається значенням біта SMOD регістра PCON: OSC/32 або OSC/64.

Режим «3» - аналогічний режиму «2», але швидкість обміну задається лічильником СТ1, як у режимі «1».

Основне настроювання послідовного каналу на необхідний режим роботи виконується у регістрі SCON (табл. 3.5), у якому задається режим роботи, значення 11-го біта, дозвіл контролю 11-го біта (у режимах «2» і «3»), прапори готовності приймача і передавача.

Таблиця 3.5 – Регістр керування/статусу SCON

Символ	**Біт**	**Ім'я і призначення**
1	*2*	*3*
SM0	7	Біти керування режимом роботи УАПП. Установлюються/скидаються програмно (див. примітку)
SM1	6	
SM2	5	Заборона прийому кадрів з нульовим восьмим бітом. У режимі «0» повинний бути скинутий. У режимі «1» при SM2=1 біт RI не встановлюється, якщо прийнятий стоп-біт дорівнює 0. У режимах «2» і «3» при SM2=1 біт RI не встановлюється, якщо прийнятий дев'ятий біт даних RB8=0. Установлюється програмно
REN	4	Біт дозволу прийому. Установлюється/скидається програмно для дозволу/заборони прийому послідовних даних

TB8	3	Передача біта 8. Установлюється/скидається програмно для завдання дев'ятого переданого біта в режимах «2», «3»

Продовження таблиці 3.5

1	*2*	*3*
RB8	2	Прийом біта 8. Установлюється/скидається апаратно для фіксації дев'ятого прийнятого біта в режимах «2», «3».У режимі «1» при SM2=0 у нього записується прийнятий стоп-біт. У режимі «0» не використовується
TI	1	Прапор готовності передавача. Установлюється апаратно по закінченні передачі байта. Скидається програмно
RI	0	Прапор готовності приймача. Установлюється апаратно по закінченні прийому байта. Скидається програмно

Примітка

SM0	SM1	Режим роботи УАПП
0	0	Режим 0
0	1	Режим 1
1	0	Режим 2
1	1	Режим 3

Запис байта в передавач здійснюється автоматично після того, як інформація записана в регістр прийомопередавача SBUF. Повторний запис можливий тільки після установлення прапора TI. Читання інформації виконується з цього ж регістра після установлення прапора готовності послідовного каналу RI.

Наявність регістра РгСПр дозволяє приймати наступний байт, коли попередній ще знаходиться у регістрі SBUF. Однак якщо підпрограма обслуговування не встигла прочитати байт даних з регістра РПП (SBUF) до моменту завершення прийому наступного байта, то черговий прийнятий байт буде записаний на місці попереднього.

На рисунку 3.20 показана часова діаграма роботи послідовного порту в режимі «0».

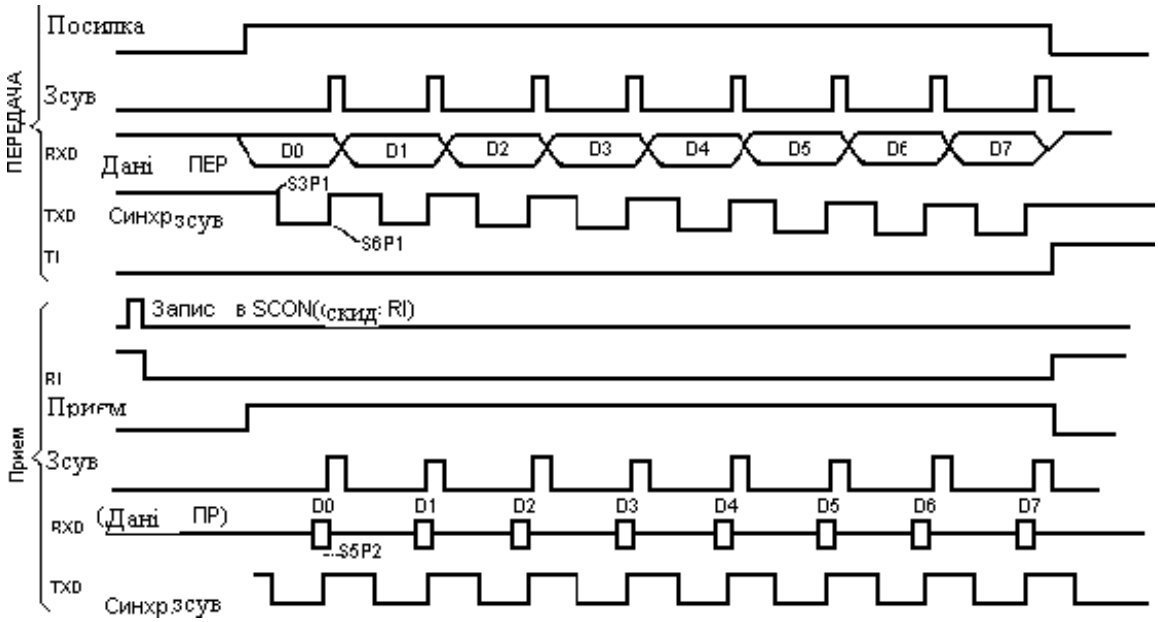


Рисунок 3.20 – Часова діаграма роботи послідовного порта в режимі 0

Передача починається будь-якою командою, за якою у SBUF надходить байт даних. У момент часу S6P2 пристрій керування ОМК за сигналом «Запис у буфер» записує байт у РгСПер і запускає блок керування передачею, що через один машинний цикл виробляє дозволений сигнал «Посилка». При цьому в момент S6P2 кожного машинного циклу вміст регістра зсувається праворуч (молодшими бітами вперед) і надходить на вивід RXD. У вивільнювані старші біти зсувного регістра передавача записуються нулі. При одержанні сигналу «Передавач порожній» блок керування передавачем знімає сигнал «Посилка» і встановлює прапор TI (момент S1P1 десятого машинного циклу після надходження сигналу «Запис у буфер»).

Прийом починається за умови REN = 1 і RI = 0. У момент S6P2 наступного машинного цикла блок керування приймачем формує дозволяючий сигнал «Прийом», за яким на вихід Tx передаються синхросигнали зсуву, й у РгСПр формуються значення біт даних, що зчитуються з входу RXD у моменти S5P2 кожного машинного циклу. У момент S1P1 десятого машинного циклу після сигналу «Запис у SCON» блок керування приймачем переписує вміст зсувного регістра у РПП, знімає дозволений сигнал «Прийом» і встановлює прапор RI.

На рисунку 3.21 показані часові діаграми роботи при прийомі та передачі даних у режимі «1». Через вивід Tx передає, а з виводу Rx приймає 10 біт: старт-біт (0), 8 біт даних і стоп-біт (1). При прийомі стоп-біт надходить у біт RB8 регістра SCON.

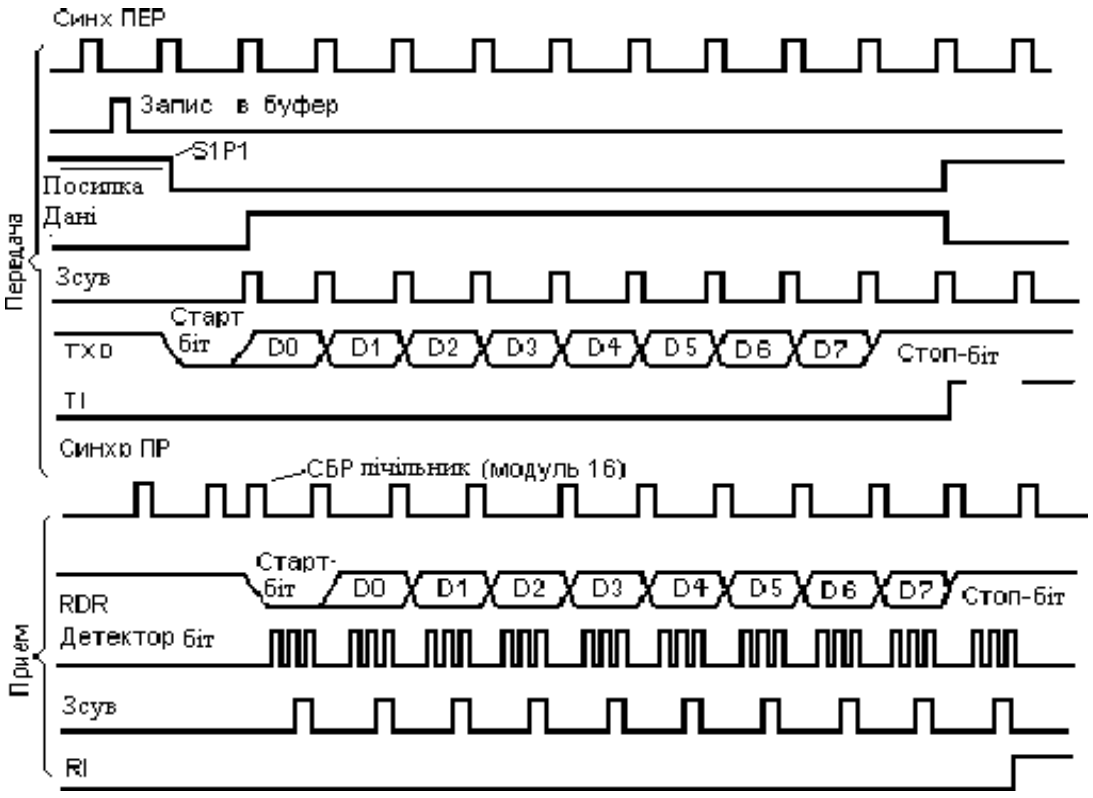


Рисунок 3.21 – Часова діаграма роботи послідовного порта в режимі 1

На рисунку 3.22 показані часові діаграми роботи при прийомі і передачі даних у режимах «2, 3» .

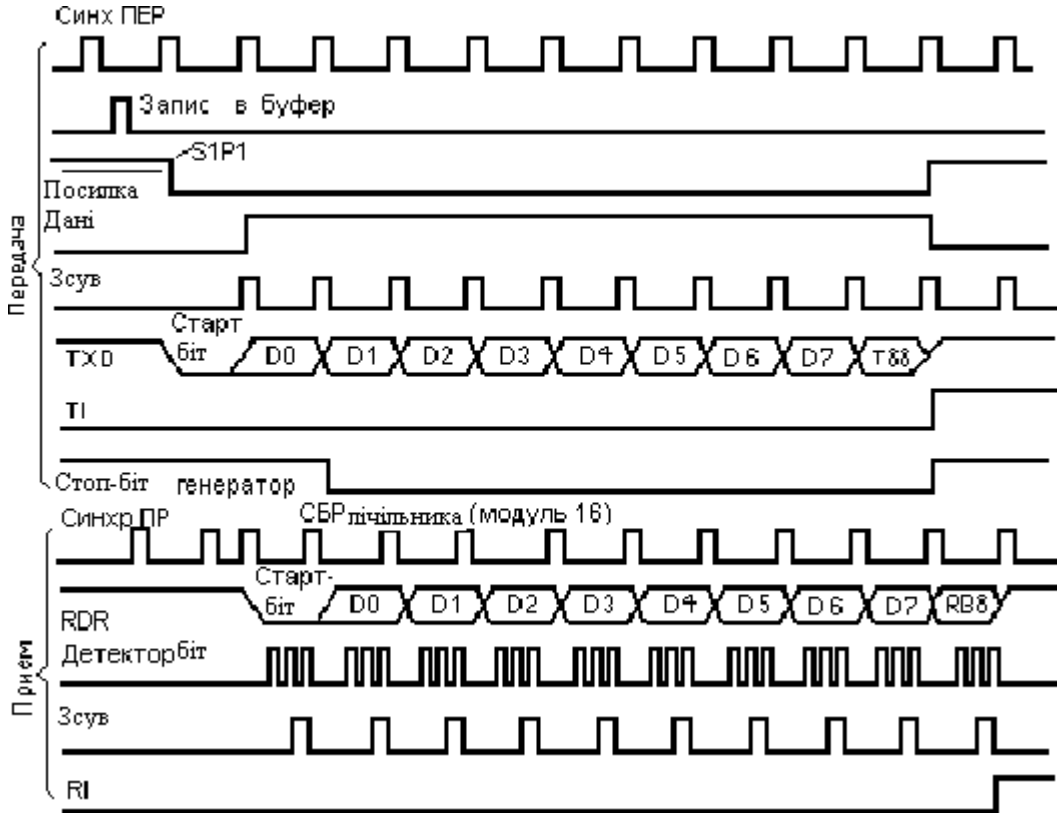


Рисунок 3.22 – Часова діаграма роботи послідовного порта в режимах 2, 3

Передача ініціюється будь-якою командою, у якій одержувачем байта є регістр SBUF. Генерований при цьому керуючий сигнал «Запис у буфер» завантажує «1» у дев'ятий біт зсувного регістра передавача, запускає блок керування передачею й у момент часу S1P1 формує дозволений сигнал «Посилка». За цим сигналом на вивід TXD спочатку надходить старт-біт, а потім (за дозволяючим сигналом «Дані») біти даних. Кожен період передачі біта дорівнює 16 тактам внутрішнього лічильника.

Прийом починається при виявленні переходу сигналу на вході RXD зі стану «1» до стану «0». Якщо значення, прийняте в першому такті, не дорівнює «0», то блок керування прийомом знову повертається до пошуку переходу з «1» у «0». Цей механізм забезпечує придушення помилкових (збійних) «старт-бітів». Справжній «старт-біт» зсувається у регістрі приймача, і продовжується прийом інших біт посилки. У процесі прийому значення біта перевіряється тричі. Справжнє значення визначається на підставі мажоритарного голосування (два з трьох). Блок керування прийомом сформує сигнал «Завантаження буфера», установить RB8 і прапор RI тільки в тому випадку, якщо в останньому такті зсуву виконуються дві умови: біт RI = 0 і SM2 = 0, або прийнятий «стоп-біт» дорівнює «1». Якщо одна з цих двох умов не виконується, то прийнята послідовність біт губиться. У цей час поза залежністю від того, виконуються зазначені умови чи ні, блок керування прийомом знову починає відшукувати перехід з «1» у «0» на вході RXD.

На часовій діаграмі (рис. 22) показана робота послідовного порту при передачі і прийомі даних у режимах «2» і «3». Як видно, режими «2» і «3» відрізняються від режиму «1» тільки наявністю дев'ятого програмованого біта. Унаслідок цього трохи змінюються умови закінчення циклу прийому: блок керування приймачем сформує керуючий сигнал «Завантаження буфера», завантажить RB8 і установить прапор RI тільки в тому випадку, якщо в останньому такті зсуву виконуються дві умови: біт RI = 0 і SM2 = 0, або значення прийнятого дев'ятого біта даних дорівнює «1».

Швидкість послідовного обміну в залежності від режиму роботи визначається або частотою роботи ОМК (режими «0» і «2») або частотою переповнення С/Т1 (режими «1» і «3»).

У режимах 1-3 імпульси переповнення лічильника С/Т1 надходять на дільник частоти, керований бітом SMOD регістра PCON, забезпечуючи зміну частоти передачі інформації в 2 рази, а потім ця частота поділяється на 16 для одержання сигналів синхронізації послідовного каналу. Структура схеми синхронізації має вид показаний на рисунку 3.23.

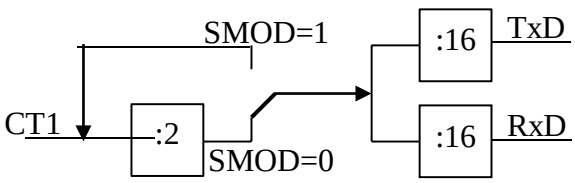


Рисунок 3.23 – Схема синхронізації послідовного каналу

У режимі «2» швидкість задається частотою $f = \frac{osc}{64} \cdot 2^{smo}$.

У режимах «1» і «3» - частотою: $f = \frac{2^{mod}}{32} \cdot f_{cn}$, $f_{C/T1}$ – частота переповнення С/Т1.

Частота переповнення лічильника в режимі «1» визначається як $\frac{osc}{12} \cdot 6553$ (65536 - коефіцієнт розподілу), а для режиму «2» - $\frac{osc}{12} \cdot 256$ (256 - коефіцієнт розподілу).

Для використання С/Т1 як джерела швидкості обміну необхідно:

- заборонити переривання від С/Т1;
- запрограмувати роботу С/Т1;
- запустити С/Т1.

Найбільш зручний для використання в послідовному каналі режим «2» лічильника/таймера (С/Т1), якщо з його допомогою можна забезпечити необхідну швидкість передачі, тому що в цьому режимі не потрібно перезавантаження коефіцієнта перерахування.

У таблиці 3.6 наводиться опис способів настроювання С/Т1 для одержання типових частот передачі даних через УАПП.

Таблиця 3.6 – Настроювання таймера 1 для керування частотою роботи УАПП

Частота прийому/передачі (BAUD RATE)	**Частота резонатора, МГц**	**SMOD**	**Таймер/лічильник 1**		
			С/Т	**Режим (MODE)**	**Число, що перезаван-тажується**
Режим 0, макс: 1 МГц	12		X	X	X
Режим 2, макс: 375 кГц	12	1	X	X	X
Режими 1, 3: 62,5 кГц	12	1	0	2	0FFh
19,2 кГц	11.059	1	0	2	0FDh
9,6 кГц	11.059	0	0	2	0FDh
4,8 кГц	11.059	0	0	2	0FAh
2,4 кГц	11.059	0	0	2	0F4h
1,2 кГц	11.059	0	0	2	0E8h
137,5 Гц	11.059	0	0	2	1Dh
110 Гц	6	0	0	2	72h
110 Гц	12	0	0	1	0FEEBh

Формат регістра PCON наведений у таблиці 3.7.

Таблиця 3.7 – Регістр керування потужністю PCON

Символ	Біт	Ім'я і призначення
SMOD	7	Подвоєна швидкість передачі. Якщо біт встановлений у 1, то швидкість передачі вдвічі більше ніж при SMOD = 0
-	6	Не використовуються
-	5	
-	4	
GF1	3	Прапори, які встановлюються користувачем (прапори загального призначення)
GF0	2	
PD	1	Біт зниженої потужності. При установці біта в 1 МК переходить у режим зниженої споживаної потужності
IDL	0	Біт холостого ходу. Якщо біт встановлений у 1, то МК переходить у режим холостого ходу

Примітка. При одночасному записі 1 у PD і IDL біт PD має перевагу. Скидання вмісту регістра виконується шляхом завантаження в нього коду 0XXX0000

Підготовка до роботи послідовного порту полягає в наступному:

- вибрати режим роботи, задавши відповідне значення регістра SCON, у якому прапори готовності мають бути скинуті;
- визначити частоту роботи. Установити необхідне значення біта SMOD. Для режимів «1» і «3» послідовного порту вибрати режим роботи СТ1. У тих випадках, коли це можливо, перевагу варто віддавати режимові «2». При роботі в інших режимах СТ необхідно забезпечити регенерацію коефіцієнта розподілу;
- вибрати спосіб уведення/виводу даних порту. У залежності від обраного способу чи заборонити, чи дозволити переривання від прапорів готовності порту;
- при прийомі інформації установити біт дозволу прийому REN.

У системах децентралізованого керування, що використовуються для керування і регулювання у розподілених об'єктах (наприклад, прокатних станах, електрорухомому складі залізниць і метрополітену, складальних конвеєрах і лініях гнучких автоматизованих виробництв), виникає задача обміну інформацією між безліччю мікроконтролерів, об'єднаних у локальну обчислювально-керуючу мережу. Як правило, локальні мережі на основі 8051 мають магістральну архітектуру з поділюваним моноканалом (коаксіальний кабель, кручена пара, оптичне волокно), за яким здійснюється обмін інформацією між ОМК. При цьому, передавальний ОМК відіграє роль ведучого, а приймаючі – роль ведених. Механізм такої передачі апаратно підтримується бітом SM2 у регістрі SCON. Ведучий мікроконтролер спочатку посилає кадр, що містить адресу одного з ведених мікроконтролерів. Адреса відрізняється від даних тим, що його дев'ятий біт

даних встановлений у «1», у той час як у кадрі даних дев'ятий біт дорівнює «0». При SM2=1 кадр адреси викликає переривання, а кадр даних – ні. Процедура обробки переривань усіх відомих аналізує прийняту адресу, при ідентифікації своєї адреси мікроконтролер скидає біт SM2 і читає наступні дані. Інші відомі залишають біт SM2 встановленим і продовжують виконувати поточну програму.

Подібний механізм дозволяє виконувати передачу в адресному, широкомовному чи груповому режимах.

При організації послідовного каналу необхідно враховувати, що передавач може працювати на довгу лінію. У цьому випадку варто використовувати зовнішні буферні схеми, які мають підвищену навантажувальну здатність.

Реалізація протоколів стандартних інтерфейсів RS-232, RS-485 вимагає застосування спеціальних БІС, які забезпечують формування відповідних сигналів: RS-232 (-12В – «лог. 1», +12В – «лог. 0»), RS-485 (диференціальний сигнал). У якості формувачів RS-232 можна використовувати передавачі - ДО559ИП19, ДО170АП2, приймачі – ДО559ИП20, ДО170УП2. Недоліком передавачів цих схем є необхідність використання джерел напруги +/-12В. Кращі характеристики мають формувачі MAX235, MAX221, 242, що живляться тільки від напруги +5В і містять в одному корпусі як приймачі, так і передавачі.

4 МІКРОКОНТРОЛЕР PIC16F877

4.1 Характеристика мікроконтролера

- Високошвидкісна архітектура RISC.
- 35 інструкцій.
- Усі команди виконуються за один цикл, окрім інструкцій переходів, що виконуються за два цикли.
- Тактова частота:
  DC – 20 МГц, тактовий сигнал DC – 200 нс, один машинний цикл.
- 8к x 14 слів FLASH пам'яті програм.
- 368 байтів пам'яті даних (ОЗП).
- 256 EEPROM пам'яті даних.
- Сумісність по виводах з PIC16C73B/74B/76/77.
- Система переривань (до 14 джерел).
- 8-рівневий апаратний стек.
- Прямій, непрямий і відносний режими адресації.
- Скидання по включенні живлення (POR).

- Таймер скидання (PWRT) і таймер очікування запуску генератора (OST) після включення живлення.
- Сторожовий таймер WDT з власним генератором RC.
- Програмований захист пам'яті програм.
- Режим енергозбереження SLEEP.
- 8 каналів 10-розрядного АЦП.
- Вибір параметрів тактового генератора.
- Високошвидкісна, енергозберігаюча CMOS FLASH/EEPROM технологія.
- Повністю статична архітектура.
- Програмування в готовому пристрої (використовується два виведення мікроконтролера).
- Низьковольтний режим програмування.
- Режим внутрішньосхемного відлагодження (використовується два виведення мікроконтролера).
- Широкий діапазон напруги живлення від 2,0 В до 5,5 В.
- Підвищена здатність навантаження портів введення/виводу (25 мА).
- Мале енергоспоживання: < 0,6 мА при 3,0 В, 4,0 МГц; 20 мкА при 3,0 В, 32 кГц ; < 1 мкА в режимі енергозбереження.

4.2 Структурна схема мікроконтролера PIC16F877

Структурна схема мікроконтролера PIC16F877 наведена на рисунку 4.1.

Розглядаючи дану структурну схему, видно, що фізичні і логічні компоненти, з яких складається PIC 16FXX, аналогічні будь-якому іншому мікроконтролерові. Тому писати програми для PIC не складніше, ніж для будь-якого іншого процесора. Звичайно, Гарвардська архітектура і велика розрядність команди дозволяє зробити код для PIC значно більш компактним, чим для інших мікроконтролерів й істотно підвищити швидкість виконання програми.

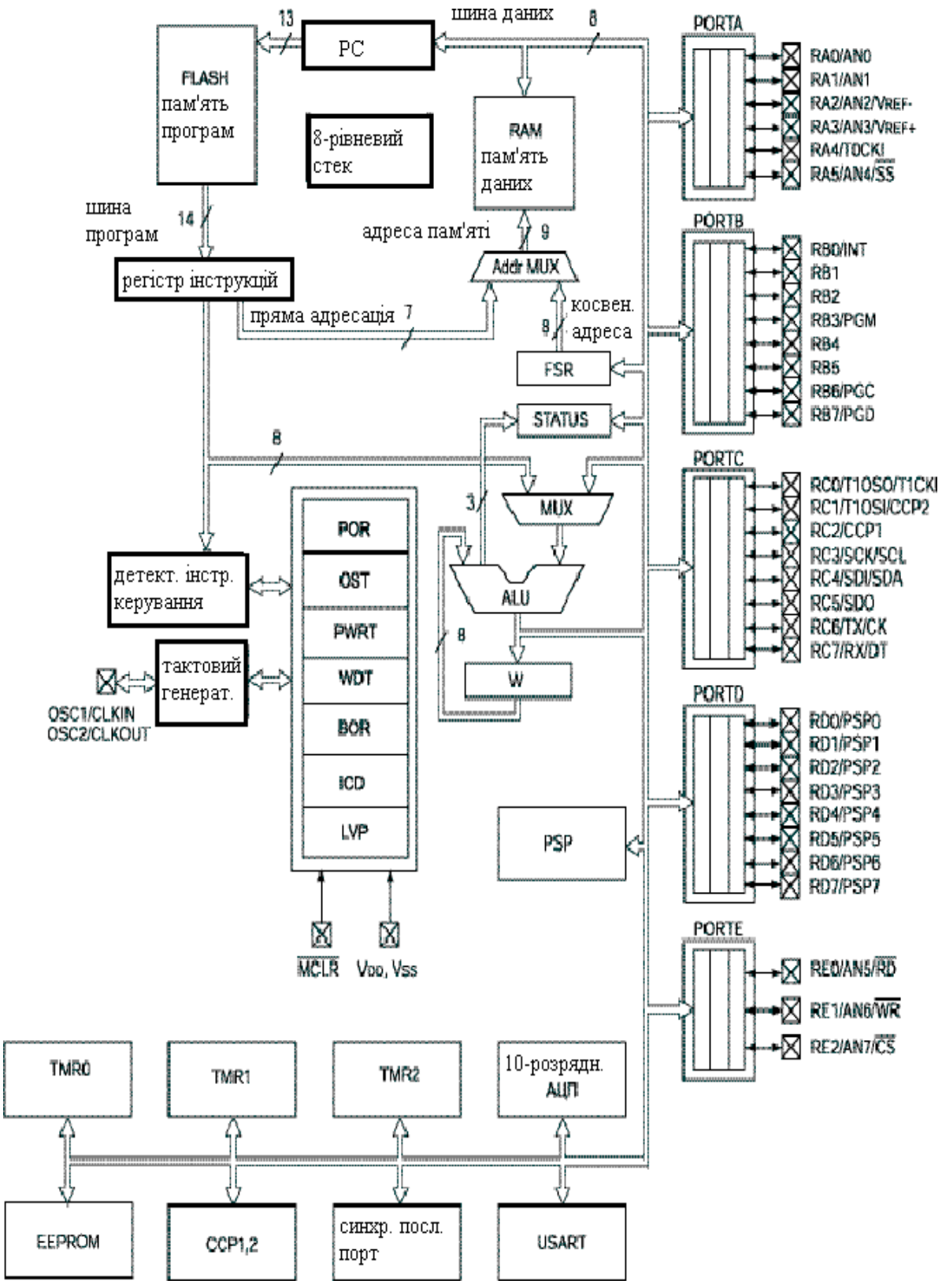


Рисунок 4.1 – Структура мікроконтролера PIC16F877

Основу структури даного мікроконтролера складають дві внутрішні шини: двонаправлена 8-бітова шина даних і 14-бітова шина команд. Це відповідає, як вже згадувалося раніше, Гарвардській архітектурі, заснованій на концепції роздільних шин і областей пам'яті для даних і команд. Шина даних зв'язує між собою всі основні функціональні блоки МК: пам'ять даних (RAM); арифметико-логічний пристрій (ALU); порти введення/виводу (PORT A,B,C,D,E); регістри стану (STATUS), непрямої адресації (FSR), таймерів-лічильників (TMR 0,1,2), програмного лічильника (PC).

У мікроконтролерах сімейства PIC існують пряма і непряма адресація всіх регістрів і елементів пам'яті. Усі спеціальні регістри і лічильник команд також відображаються на пам'ять даних.

Мікроконтролери PIC16FXX мають ортогональну (симетричну) систему команд, що дозволяє виконувати будь-яку операцію з будь-яким регістром, використовуючи будь-який метод адресації. Це полегшує програмування для них і значно зменшує час, необхідний на навчання роботі з ними.

Арифметико-логічний пристрій (ALU) 8-розрядний і виконує складання, віднімання, зрушення, бітові і логічні операції. У командах, що мають два операнди, одним з операндів є робочий регістр W. Другий операнд може бути константою або вмістом будь-якого регістра ОЗП. У командах з одним операндом операнд може бути вмістом робочого регістра або вмістом будь-якого регістра. Для виконання всіх операцій ALU використовує робочий регістр W, який не може бути прямо адресований. Залежно від результату виконання операції, можуть змінитися значення бітів перенесення C, десяткового перенесення DC і нуля Z в регістрі стану STATUS. При відніманні біти C і DC працюють як біти позики і десяткової позики відповідно.

Вхідна тактова частота, що надходить з виведення OSC1/CLKIN, усередині ділиться на чотири і з неї формуються чотири циклічні тактові послідовності, що не перекриваються Q1, Q2, Q3 і Q4.

Вибірка команди і її виконання суміщені за часом таким чином, що вибірка команди займає один цикл, а виконання – наступний цикл. Ефективний час виконання команди складає один цикл. Якщо команда змінює лічильник команд (наприклад, команда GOTO), то для виконання цієї команди буде потрібно два цикли. Цикл вибірки починається із збільшення лічильника команд в такті Q1. У циклі виконання команди вибрана команда защіпується в регістр команд в такті Q1. Протягом тактів Q2, Q3 і Q4 відбувається декодування і виконання команди. У такті Q3 прочитується пам'ять даних (читання операнда), а запис відбувається в такті Q4. Таким чином, цикл виконання команди складається з 4-х тактів Q1-Q4, в кожному з яких проводяться різні, заздалегідь визначені дії:

Q1 – Вибірка певної команди з пам'яті програм і її декодування або вимушений NOP.

Q2 – Вибірка даних або NOP.

Q3 – Виконання команди і обробка даних.

Q4 – Запис даних або NOP.

Характеристика периферійних модулів:

– Таймер 0: 8-розрядний таймер/лічильник з 8-розрядним програмованим перед дільником.

– Таймер 1: 16-розрядний таймер/лічильник з можливістю підключення зовнішнього резонатора.

– Таймер 2: 8-розрядний таймер/лічильник з 8-розрядним програмованим переддільником і вихідним дільником.

– Два модулі порівняння/захоплення/ШІМ (PCP).

– 16-розрядне захоплення (максимальна роздільна здатність 12,5 нс); 16-розрядне порівняння (максимальна роздільна здатність 200 нс); 10-розрядний ШІМ.

– Багатоканальний 10-розрядний АЦП.

– Послідовний синхронний порт MSSP; ведучий/ведений режим SPI; ведучий/ведений режим I2C.

– Послідовний синхронно-асинхронний приймач USART з підтримкою детектування адреси.

– Ведений 8-розрядний паралельний порт PSP з підтримкою зовнішніх сигналів: -RD,-WR, -CS.

– Детектор зниженої напруги (BOD) для скидання по зниженні напруги живлення (BOR).

4.3 Організація пам'яті

У мікроконтролерах PIC16F87X існує два види пам'яті. Пам'ять програм і пам'ять даних мають роздільні шини даних і адреси, що дозволяє виконувати паралельний доступ.

4.3.1 Пам’ять програм

Мікроконтролери PIC16F87X мають 13-розрядний лічильник команд PC, здатний адресувати 8К x 14 слів пам'яті програм. Фізично реалізовано FLASH пам'яті програм 8К x 14 у PIC16F877. Адреса вектора скидання – 0000h. Адреса вектора переривань – 0004h.

Пам'ять програм (ППЗП) має сторінкову організацію (рис. 4.2). Об'єм однієї сторінки 2048 14-розрядних слів. Мікроконтролер PIC 16F877 має чотири сторінки пам'яті програм для зберігання 14-розрядних кодів команд.

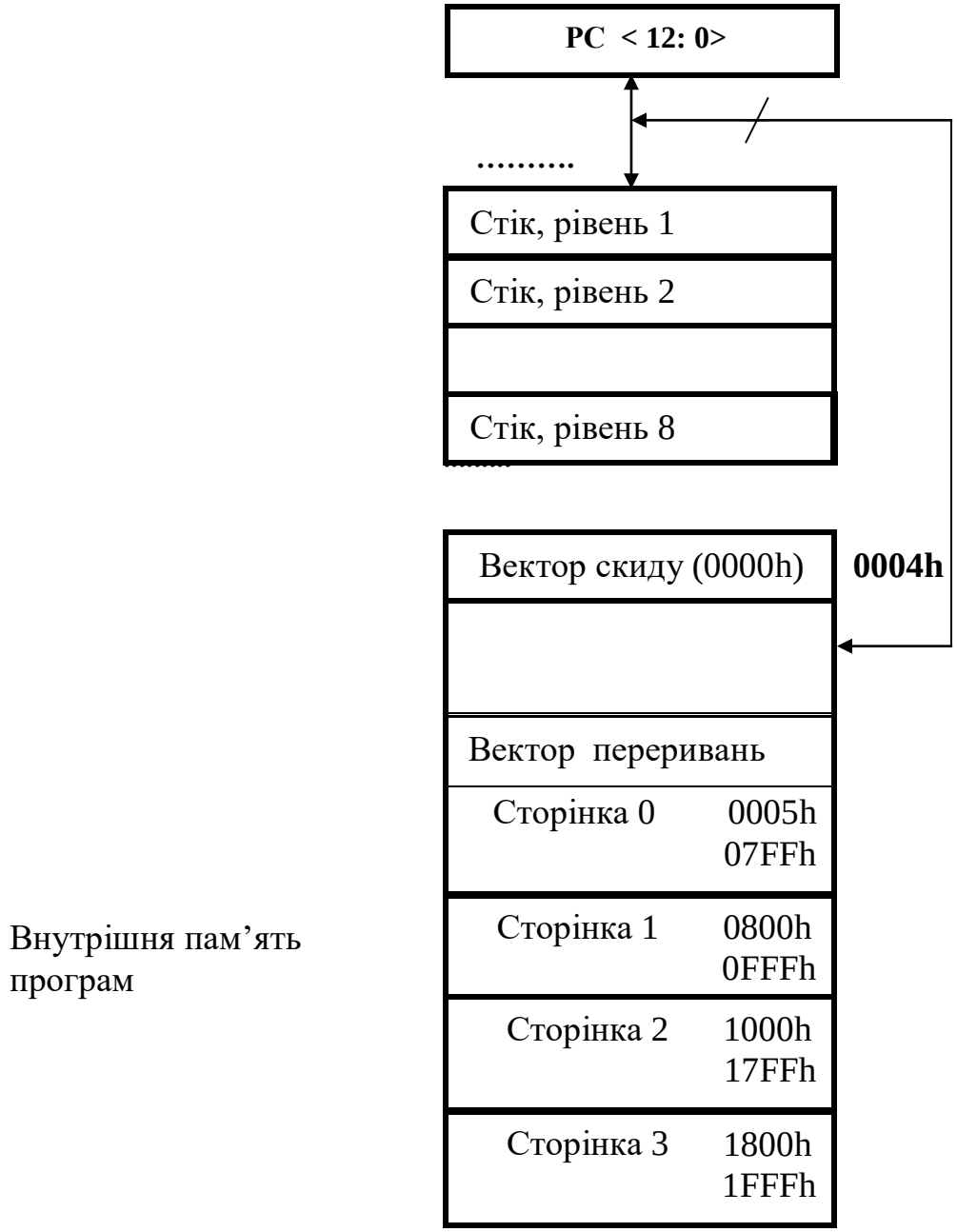


Рисунок 4.2 – Структура пам’яті програм

Усі мікроконтролери PIC16F87X здатні адресувати 8К слів пам'яті програм. Інструкції переходів (CALL і GOTO) мають 11-разрядне поле для вказівки адреси, що дозволяє безпосередньо адресувати 2К слів пам'яті програм. Для адресації верхніх сторінок пам'яті програм використовуються 2 біта в регістрі PCLATH<4:3>. Перед виконанням команди переходу (CALL або GOTO) необхідно запрограмувати біти регістра PCLATH<4:3> для адресації необхідної сторінки.

При виконанні інструкцій повернення з підпрограми, 13-розрядне значення для лічильника програм PC береться з вершины стека, тому маніпуляція бітами регістра PCLATH<3:4> не потрібна.

4.3.2 Організація пам'яті даних

Пам'ять даних поділена на чотири банки, які містять регістри загального і спеціального (SFR) призначення. Біти RP1 (STATUS<6>) і RP0 (STATUS<5>) призначені для управління банками даних. У таблиці 4.1 показаний стан керуючих бітів при зверненні до банків пам'яті даних. Карта пам'яті даних показана на рисунку 4.3.

Таблиця 4.1 – Адресація банків даних

RP1:RPO	Банк
00	**0**
01	**1**
10	**2**
11	**3**

Об'єм банків пам'яті даних до 128 байтів (7Fh). На початку банку розміщуються регістри спеціального призначення, потім регістри загального призначення виконані як статичний ОЗП. Усі реалізовані банки містять регістри спеціального призначення. Деякі, часто використовувані регістри спеціального призначення, можуть відображатися і в інших банках пам'яті.

Регістри можуть бути адресовані прямо або побічно з використанням регістра непрямої адресації FSR. Безпосередня адресація підтримується спеціальними командами, що завантажують дані з пам'яті програми в робочий регістр W.

Регістри спеціального призначення використовуються для керування функціями мікроконтролера і можуть бути розділені на два набори: регістри базових функцій і регістри периферійних пристроїв. Регістри базових функцій містять у собі регістр-перемикач непряму адресації (INDF), програмний лічильник (PC), представлений двома регістрами PCL і PCLATH, регістр слова стану (STATUS), регістр-покажчик непряму адресації (FSR), робочий регістр (W), регістр переривань (INTCON), а також регістр режимів роботи та конфігурації попереднього дільника і таймера (OPTION). Регістри периферійних пристроїв містять у собі регістри (RA,RB,RC,RD,RE-порти A,B,C,D,E), регістри даних (EEDATA) і адреси (EEADR) пам'яті даних-констант, регістри таймерів-лічильників (TMR0,1,2) і регістри керування конфігурацією портів вводу/виводу (TRISA, TRISB, TRISC, TRISD, TRISE).

Регістри загального призначення використовуються для зберігання даних по розсуду користувача.

	Адреса		Адреса		Адреса		Адреса
Регістр * непрямої адресації	00h	Регістр * непрямої адресації	80h	Регістр * непрямої адресації	100h	Регістр * непрямої адресації	180h
TMR0	01h	OPTION	81h	TMR0	101h	OPTION REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h	x	185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h	x	107h	x	187h
PORTD(1)	08h	TRISD(1)	88h	x	108h	x	188h
PORTE(1)	09h	TRISE(1)	89h	x	109h	x	189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Резерв^'	18Eh
TMR1H	0Fh	x	8Fh	EEADRH	10Fh	Резерв^'	18Fh
T1CON	10h	x	90h	Регістри загального призна-чення 16 байтів	110h	Регістри загального призна-чення 16 байтів	19Fh
TMR2	11h	SSPCON2	91h				
T2CON	12h	PR2	92h				
SSPBUF	13h	SSPADD	93h				
SSPCON	14h	SSPSTAT	94h				
CCPR1L	15h	x	95h				
CCPR1 H	16h	x	96h				
CCP1CON	17h	x	97h				
RCSTA	18h	TXSTA	98h				
TXREG	19h	SPBRG	99h				
RCREG	1Ah	x	9Ah				
CCPR2L	1Bh	x	9Bh				
CCPR2H	1Ch	x	9Ch				
CCP2CON	1Dh	x	9Dh				
ADRESH	1Eh	ADRESL	9Eh				
ADCONO	1Fh	ADCON1	9Fh		11Fh		
Регістри загаль-ного призна-чення 96 байтів	20h	Регістри загаль-ного призна-чення 80 байтів	A0h	Регістри загаль-ного призна-чення 80 байтів	120h	Регістри загаль-ного призна-чення 80 байтів	1A0h
	7Fh	Доступ до 70h- 7Fh	EFh F0h FFh	Доступ до 70h- 7Fh	16Fh 170h 17Fh	Доступ до 70h- 7Fh	1EFh 1F0h
Банк 0		Банк 1		Банк 2		Банк3	

*- нефізичний регистр

x – не реалізовані, значення при читанні 00h

Рисунок 4.3 – Карта пам'яті даних мікроконтролерів PIC16F876/877

Для виконання непрямої адресації необхідно звернутися до фізично не реалізованого регістра INDF. Звернення до регістра INDF фактично викличе дію з регістром, адреса якого вказана в FSR. Непряме читання регістра INDF (FSR=0) дасть результат 00h. Непрямий запис в регістр INDF не викличе ніяких дій (викликає дії на прапори АЛП в регістрі STATUS). 9-й біт непрямої адреси IRP зберігається в регістрі STATUS<7>. Приклад 9-розрядної непрямої адресації показаний на рисунку 4.4.

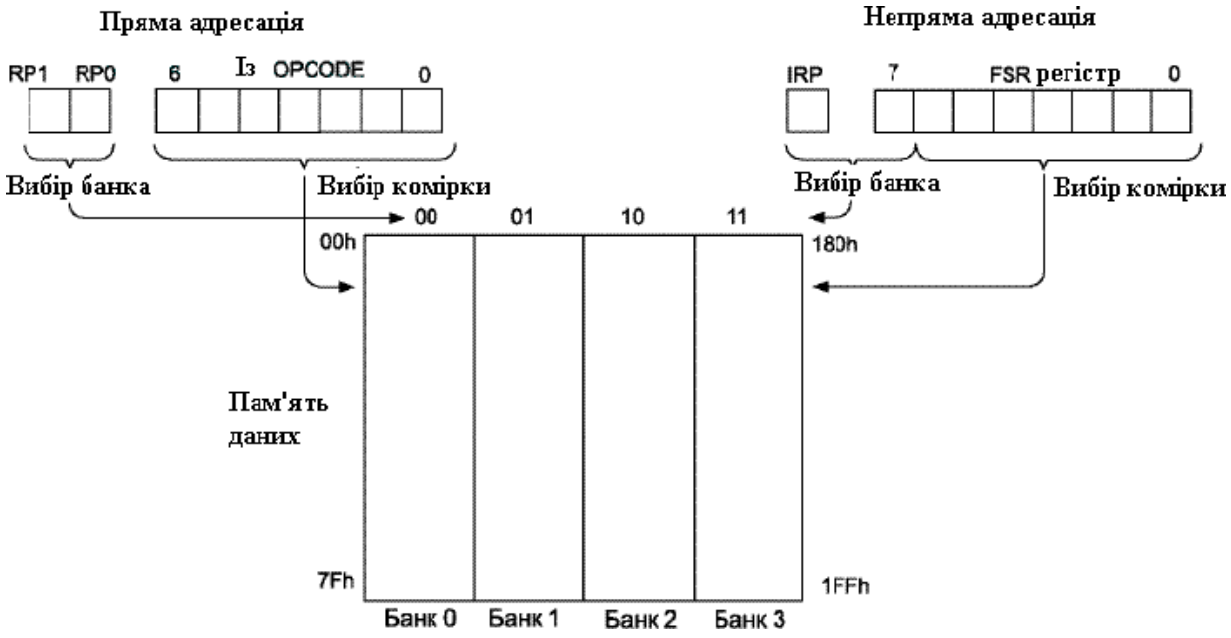


Рисунок 4.4 – Приклад 9-розрядної непрямої адресації

4.4 Регістр стану STATUS

Регістр стану (адреса 03h, 83h, 103h або 183h) містить арифметичні прапори АЛП, біти стану контролера при скиданні і біти вибору сторінок пам’яті. Регістр STATUS доступний для будь-якої команди так само, як будь-який інший регістр. Проте біти TO і PD встановлюються апаратно і не можуть бути записані в регістр статусу програмно. Це слід мати на увазі при виконанні команди з використанням регістра статусу. Наприклад, команда CLRF обнулить усі біти, окрім бітів TO і PD, а потім встановить біт Z=1. Після виконання цієї команди регістр статусу може і не мати нульового значення (із-за бітів TO і PD) 000??100. Тому рекомендується для зміни регістра статусу використовувати тільки команди бітової установки BCF, BSF, MOVWF, які не змінюють решту біт статусу.

Розміщення прапорів в регістрі STATUS наступне:

b7	b6	b5	b4	b3	b2	b1	b0

IRP	RP1	RP0	-TO	- PD	Z	DC	C

Біт 7: IRP: Біт вибору банку при непрямій адресації
1 – банк 2, 3(100…1FFh)
0 – банк 0, 1 (000… 0FFh)

Біти 6-5: RP1:RPO: Біти вибору банку при безпосередній адресації
11 – банк 3 (180…1FFh)
10 – банк 2 (100…17Fh)
01 – банк 1 (080… 0FFh)
00 – банк 0 (000… 07Fh)

Біт 4: -TO: Прапор переповнювання сторожового таймера
1 – після POR або виконань команд CLRWDT, SLEEP
0 – після переповнювання WDT

Біт 3: -PD: Прапор включення живлення
1 – після POR або виконань команди CLRWDT
0 – після виконання команди SLEEP

Біт 2: Z: Прапор нульового результату
1 – нульовий результат виконання арифметичної або логічної операції
0 – не нульовий результат виконання арифметичної або логічної операції

Біт 1: DC: Прапор десяткового перенесення/позики (для команд ADDWF, ADDWL, SUBWF, SUBWL), позика має інверсне значення
1 – було перенесення з молодшого півбайта
0 – не було перенесення з молодшого півбайта

Біт 0: C: Прапор перенесення/позики (для команд ADDWF, ADDWL, SUBWF, SUBWL), позика має інверсне значення
1 – було перенесення із старшого біта
0 – не було перенесення із старшого біта

Примітка. Прапор позики має інверсне значення. Віднімання виконується шляхом збільшення додаткового коду другого операнда. При виконанні команд зрушення (RRF, RLF) біт C завантажується старшим або молодшим бітом зрушуваного регістра.

4.5 Регістр OPTION

Регістр OPTION доступний для читання і запису, містить біти управління:

- попереднім дільником TMR0/WDT;
- активним фронтом зовнішнього переривання RB0/INT;

- підтягаючими резисторами на входах PORTB.

Примітка. Якщо попередній дільник включений перед WDT, то коефіцієнт ділення тактового сигналу для TMR0 дорівнює 1:1.

Розміщення бітів керування в регістрі OPTION наступне:

b7	b6	b5	b4	b3	b2	b1	b0
-RBPU	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0

біт 7: -RBPU: Включення підтягаючих резисторів на входах PORTB:

1 = підтягаючі резистори відключені, 0 = підтягаючі резистори включені;

біт 6: INTEDG: Вибір активного фронту сигналу на вході зовнішнього переривання: INT 1 = переривання за переднім фронтом сигналу,

0 = переривання за заднім фронтом сигналу;

біт 5:TOCS: Вибір тактового сигналу для TMR0: 1 = зовнішній тактовий сигнал з виведення RA4/TOCKI, 0 = внутрішній тактовий сигнал CLKOUT;

біт 4: TOSE: Вибір фронту приросту TMR0 при зовнішньому тактовому сигналі: 1 = приріст за заднім фронтом сигналу (з високого до низького рівня) на виведенні RA4/TOCKI, 0 = приріст за переднім фронтом сигналу (з низького до високого рівня) на виведенні RA4/TOCKI;

біт 3: PSA: Вибір включення переддільника: 1 = переддільник включений перед WDT, 0 = переддільник включений перед TMR0;

біти 2-0: PS2…PS0: Установка коефіцієнта ділення переддільника, коефіцієнти ділення наведені в таблиці 4.2.

Таблиця 4.2 – Коефіцієнти ділення переддільника

Значення PS2 – PS0	Для TMR0	Для WDT
000	1 :2	1:1
001	1 :4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1 :256	1:128

Примітка. При використанні режиму низьковольтного програмування і включених підтягаючих резисторах на PORTB необхідно скинути в '0' 3-й біт регістра TRISB для виключення підтягаючого резистора на виведенні RB3.

4.6 Регістр INTCON

Регістр INTCON доступний для читання і запису, містить біти дозволів і прапори переривань: переповнювання TMR0; зміни рівня сигналу на виводах PORTB; зовнішнє джерело переривань RBO/INT.

Розміщення бітів в регістрі INTCON наступне:

b7	b6	b5	b4	b3	b2	b1	b0
GIE	|PEIE	TOIE	INTE	RBI	TOIF	INTF	RBIF

біт 7: **GIE:** Глобальний дозвіл переривань:
1 = дозволені всі немасковані переривання,
0= усі переривання заборонені;

біт **6:** **PEIE:** Дозвіл переривань від периферійних модулів:
1= дозволені всі немасковані переривання периферійних модулів,
0= переривання від периферійних модулів заборонені;

біт **5:** **TOIE:** Дозвіл переривання по переповнюванні TMR0:
1= переривання дозволене,
0= переривання заборонене;

біт **4:** **INTE:** Дозвіл зовнішнього переривання INT:
1= переривання дозволене,
0= переривання заборонене;

біт **3:** **RBIE:** Дозвіл переривання по зміні сигналу на входах RB7:RB4 PORTB:
1= переривання дозволене,
0= переривання заборонене;

біт **2:** **TOIF:** Прапор переривання по переповнюванні TMR0:
1= відбулося переповнення TMR0 (скидається програмно),
0= переповнювання TMR0 не було;

біт **1:** **INTF:** Прапор зовнішнього переривання INT:
1=виконана умова зовнішнього переривання на виведенні RBO/INT (скидається програмно),
0= зовнішнього переривання не було;

біт **0:** **RBIF:** Прапор переривання по зміні рівня сигналу на входах RB7:RB4 PORTB:
1 = зафіксована зміна рівня сигналу на одному з входів RB7:RB4 (скидається програмно),
0 = не було зміни рівня сигналу ні на одному з входів RB7:RB4.

Примітка. Прапори переривань встановлюються при виникненні умов переривань незалежно від відповідних бітів дозволу і біта загального

дозволу переривань GIE (INTCON<7>).

4.7 Лічильник команд

13-розрядний регістр лічильника команд PC указує адресу виконуваної інструкції. Молодший байт лічильника команд PCL доступний для читання і запису. Старший байт PCH, що містить <12:8> біти лічильника команд PC, не доступний для читання і запису. Усі операції з регістром PCH відбуваються через додатковий регістр PCLATH. При будь-якому виді скидання мікроконтролера лічильник команд PC очищається. На рисунку 4.5 показано дві ситуації завантаження значення в лічильник команд PC. Приклад зверху: запис в лічильник команд PC відбувається при записі значення в регістр PCL (PCLATH <4:0> —> PCH). Приклад знизу: запис значення в лічильник команд PC відбувається при виконанні команди CALL або GOTO (PCLATH <4:3> -» PCH).

Обчислюваний перехід може бути виконаний командою приросту до регістра PCL (наприклад, ADDWF PCL). При виконанні табличного читання обчислюваним переходом слід піклуватися про те, щоб значення PCL не перетнуло межу блоку пам'яті (кожен блок – 256 байтів).

Усі мікроконтролери PIC16F87X здатні адресувати 8 К слів пам'яті програм. Інструкції переходів (CALL і GOTO) мають 11-разрядне поле для вказівки адреси, що дозволяє безпосередньо адресувати 2 К слів пам'яті програм. Для адресації верхніх сторінок пам'яті програм використовуються 2 біта в регістрі PCLATH<4:3>. Перед виконанням команди переходу (CALL або GOTO) необхідно запрограмувати біти регістра PCLATH<4:3> для адресації необхідної сторінки.

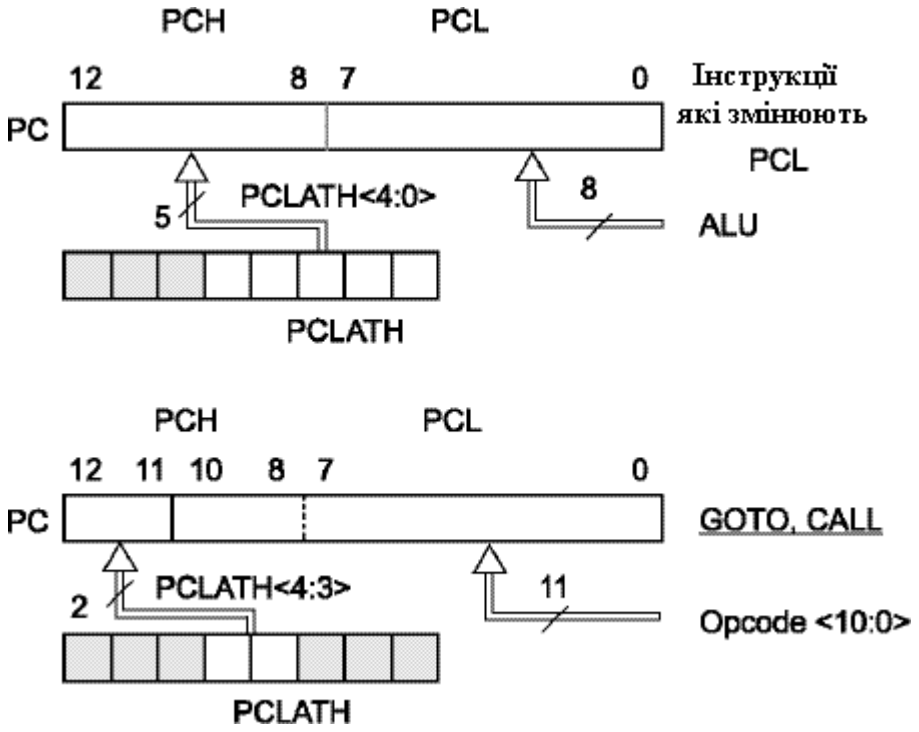


Рисунок 4.5 – Приклади завантаження лічильника команд

При виконанні інструкцій повернення з підпрограми 13-розрядне значення для лічильника програм PC береться з вершини стека, тому маніпуляція бітами регістра PCLATH<3:4> не потрібна.

Примітка. Уміст регістра PCLATH не змінюється після виконання інструкції RETURN або RETFIE. Користувач має сам змінити значення регістра PCLATH для подальшого виконання команд GOTO і CALL.

4.8 Стік

PIC16F87X мають 8-рівневий 13-розрядний апаратний стік. Стік не має відображення на пам'ять програм і пам'ять даних, не можна записати або прочитати дані зі стека. Значення лічильника команд заноситься у вершину стека при виконанні інструкцій переходу на підпрограму (CALL) або обробки переривань. Читання зі стека і запис в лічильник команд PC відбувається при виконанні інструкцій повернення з підпрограми або обробки переривань (RETURN, RETLW, RETFIE), при цьому значення регістра PCLATH не змінюється.

Стек працює як циклічний буфер. Після 8 записів в стек, дев'ятий запис запишеться на місце першого, а десятий запис замінить другий і так далі.

Примітки:

1 У мікроконтролерах не існує ніяких покажчиків про переповнювання стека.

2 У мікроконтролерах не передбачено команд запису/читання із стека, окрім команд виклику/повернення з підпрограм (CALL, RETURN, RETLW і RETFIE) або умов переходу за вектором переривань.

4.9 Порти введення/виводу

Регістри введення/виводу можуть керуватися як будь-які інші регістри. Проте команда «читання» (наприклад, MOVF 6,W) завжди прочитує фактичний рівень сигналу на виводі порту, незалежно від того, запрограмований цей розряд порту на введення або на вивід. Після сигналу «Скидання» всі порти введення/виводу встановлюються на «введення» (електрично еквівалентно третьому стану), а регістри керування введенням/виводом (TRISA, TRISB, TRISC, TRISD, TRISE) встановлюються в одиниці (конфігурація на введення). Для того щоб конфігурувати деякі лінії порту на вивід, необхідно встановити відповідні біти в потрібному регістрі TRIS в «0». Це можна робити командою "TRIS f".

При операціях введення порти не защіпуються. Вхідний сигнал повинен бути присутнім поки йде процес читання (наприклад, MOVF 6, W). При операціях виводу порти защіпуються і зберігають значення до тих пір, поки не будуть перезаписані. На рисунку 2.6 не показані діоди, які захищають ніжку порту від зовнішніх імпульсів великої напруги. Вони обмежують імпульсну напругу на ніжці значеннями від Vss - 0,6 до Vdd + 0,6 В. Якщо статична напруга вийде поза вказані межі, то виникнуть великі статичні струми, здатні вивести мікроконтролер з ладу.

Деякі канали портів введення/виводу мультиплікові з периферійними модулями мікроконтролера. Коли периферійний модуль включений, вивід не може використовуватися як універсальний канал введення/виводу.

4.9.1 Регістри PORTA і TRISA

PORTA – 6-розрядний порт уведення/виводу. Усі канали PORTA мають відповідні біти напряму в регістрі TRISA, що дозволяють настроювати канал як вхід або вихід. Запис «1» в TRISA переводить відповідний вихідний буфер у 3-й стан. Запис «0» в регістр TRISA визначає відповідний канал як вихід, уміст защіпки PORTA передається на виведення мікроконтролера (якщо вихідна защіпка підключена до виведення мікроконтролера).

Читання регістра PORTA повертає стан на виводах порту, а запис проводиться в защіпку PORTA. Всі операції запису в порт виконуються за

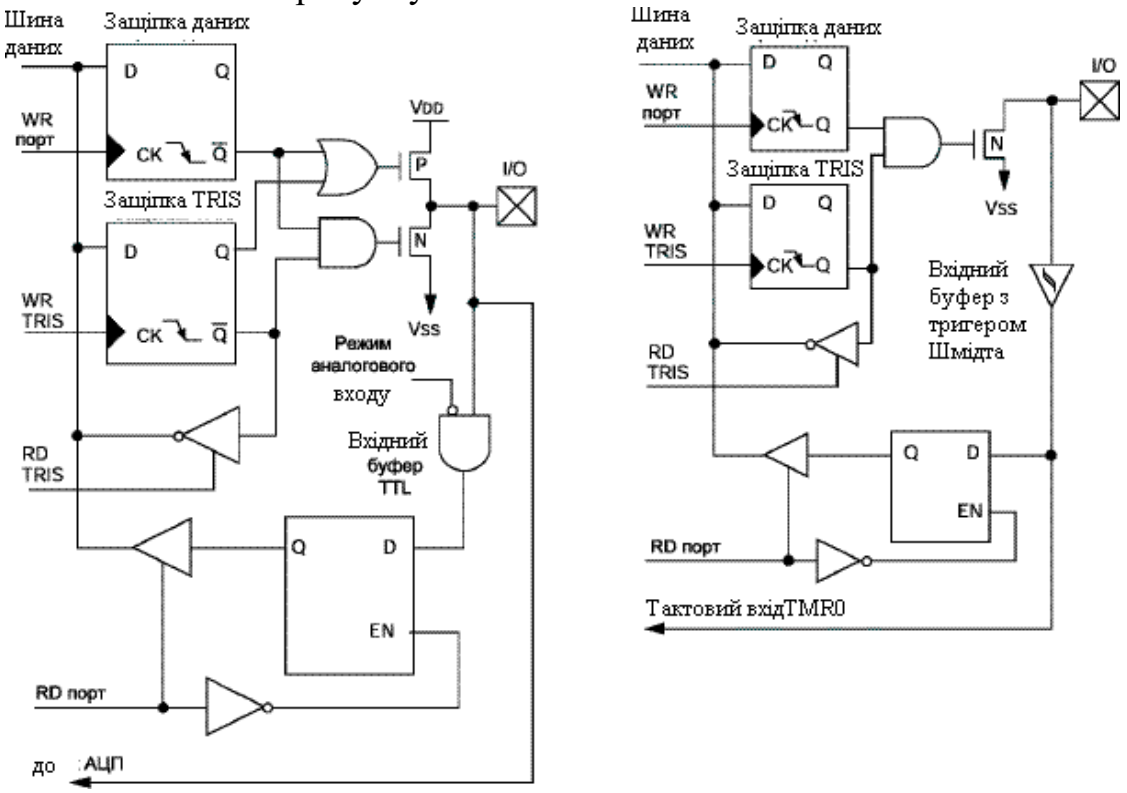
принципом «читання – модифікація – запис», тобто спочатку проводиться читання стану виводів порту, потім зміна і запис у защіпку.

RA4 – має трігер Шмідта на вході і відкритий стік на виході, мультиплікований з тактовим входом TOCKI. Решта всіх каналів PORTA має TTL буфер на вході і повнофункціональні вихідні КМОП буфери.

Канали PORTA мультиплікований з аналоговими входами АЦП і аналоговим входом джерела опорної напруги VREF. Біти управління режимів роботи каналів порту введення/виводу PORTA знаходяться в регістрі ADCON1.

Примітка. Після скидання по включенні живлення виводи настроюються як аналогові входи, а читання дає результат «0».

Біти регістра TRISA управляють напрямом каналів PORTA, навіть коли вони використовуються як аналогові входи. Користувач повинен упевнитися, що відповідні канали PORTA налаштовані на вхід при використанні їх як аналогових входів. Структурні схеми виводів порта PORTA показані на рисунку 4.6.



а *б*

а – виводи RA3-RA0 та RA5; б – вивод RA4

Рисунок 4.6 – Структурна схема виводів порта PORTA

4.9.2 Регістри PORTB і TRISB

PORTB – 8-розрядний двонаправлений порт введення/виводу. Біти регістра TRISB визначають напрям каналів порту. Установка біта в 1 регістра TRISB переводить вихідний буфер у 3-й стан. Запис '0' в регістр TRISB настроює відповідний канал як вихід, уміст защіпки PORTB передається на виведення мікроконтролера (якщо вихідна защіпка підключена до виведення мікроконтролера).

Три виведення PORTB мультиплікові зі схемою низьковольтного програмування: RB3/PGM, RB6/PGC, RB7/PGD.

До кожного виведення PORTB підключений внутрішній підтягаючий резистор. Біт-RBPU (OPTION_REG <7>) визначає підключені (-RBPU=0) чи ні (-RBPU=1) підтягаючі резистори. Підтягаючі резистори автоматично відключаються, коли канали порту настроюються на вихід і після скидання по включенні живлення POR.

Чотири канали PORTB RB7:RB4, налаштовані на вхід, можуть генерувати переривання по зміні логічного рівня сигналу на вході. Якщо один з каналів RB7:RB4 налаштований на вихід, то він не може бути джерелом переривань. Сигнал на виводах RB7:RB4 порівнюється із значенням, збереженим при останньому читанні PORTB. У разі неспівпадання одного зі значень встановлюється прапор RBIF (INTCON<0>) і, якщо дозволено, генерується переривання.

Це переривання може вивести мікроконтролер з режиму SLEEP. У підпрограмі обробки переривань необхідно зробити наступні дії:

- виконати читання або запис у PORTB, виключивши невідповідність;
- скинути прапор RBIF в «0».

Невідповідність збереженого значення з сигналом на вході PORTB завжди встановлює біт RBIF в 1. Читання з PORTB перерве умову невідповідності і дозволить скинути прапор RBIF в «0».

Переривання по зміні сигналу на входах рекомендується використовувати для визначення натиснення клавіш, коли PORTB повністю задіяний для реалізації клавіатури. Не рекомендується опитувати PORTB при використанні переривань по зміні вхідного сигналу.

Переривання по зміні сигналу на входах PORTB і програма перемикання конфігурації цих каналів дозволяє реалізувати простий інтерфейс обслуговування клавіатури з виходом з режиму SLEEP по натисненню клавіш .

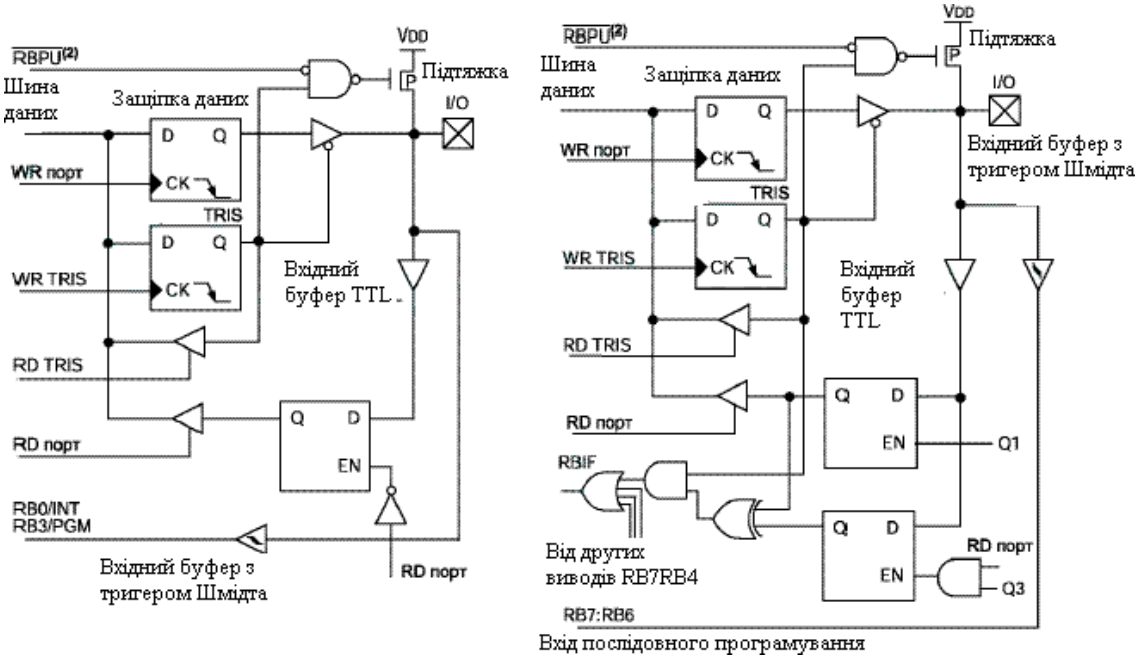
RBO/INT вхід зовнішнього джерела переривань, що настроюються бітом INTEDG (OPTION_REG<6>).

Структурна схема виводів порта PORTB показана на рисунку 4.7.

Примітки:

Виводи портів мають захисні діоди, підключені до V_{DD} і Vss-

Для включення підтягаючих резисторів необхідно встановити в «1» відповідний біт TRIS і скинути в «0» біт -RBPU (OPTION_REG<7>).



а *б*

а – виводи RB3-RB0; б – виводи RB7-RB4

Рисунок 4.7 – Структурна схема виводів порта PORTB

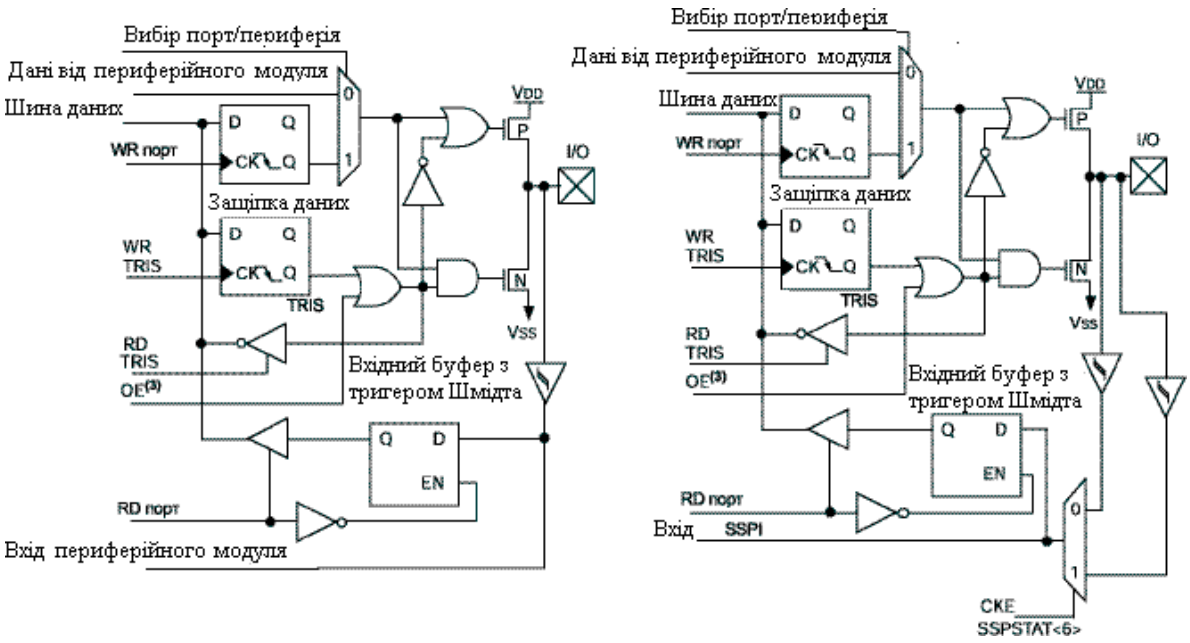
4.9.3 Регістри PORTC і TRISC

PORTC – 8-розрядний двонаправлений порт уведення/виводу. Біти регістра TRISC визначають напрям каналів порту. Установка біта в «1» регістра TRISC переводить вихідний буфер у 3-й стан. Запис «0» в регістр TRISC настроює відповідний канал як вихід, уміст защіпки PORTC передається на виведення мікроконтролера (якщо вихідна защіпка підключена до виведення мікроконтролера).

Виводи PORTC мультиплексовані з декількома периферійними модулями. На каналах PORTC присутній вхідний буфер з тригером Шмідта.

Коли модуль MSSP включений в режимі I2C, виводи PORTC<4:3> можуть підтримувати рівні вихідних сигналів за специфікацією I2C або SMBus залежно від стану біта CKE(SSPSTAT<6>).

При використанні периферійних модулів необхідно відповідним чином настроювати біти регістра TRISC для кожного виведення PORTC. Деякі периферійні модулі відміняють дію бітів TRISC, примусово настроюючи вивід на вхід або вихід. У зв'язку з чим не рекомендується використовувати команди «читання – модифікація – запис» з регістром TRISC. Структурна схема виводів порта PORTC показана на рисунку 4.8.



а *б*

а – виводи RC7-RC5, RC2-RC0; б – виводи RC4-RC3

Рисунок 4.8 – Структурна схема виводів порта PORTC

Примітки:

1 Виводи портів мають захисні діоди, підключені до VDD і Vss.

2 Сигнал режиму каналу – вивід – використовується периферійним модулем або цифровим портом введення/виводу.

3 Сигнал дозволу (OE) від периферійного модуля, настроює канал як вихід.

4.9.4 Регістри PORTD і TRISD

PORTD – 8-розрядний двонаправлений порт введення/виводу. Біти регістра TRISD визначають напрям каналів порту.

PORTD може працювати як 8-розрядний мікропроцесорний порт (ведений паралельний порт), якщо біт PSPMODE (TRISE<4>) встановлений в «1». У режимі веденого паралельного порту до входів підключені буфери TTL. Структурна схема виводів порта PORTD показана на рисунку 4.9.

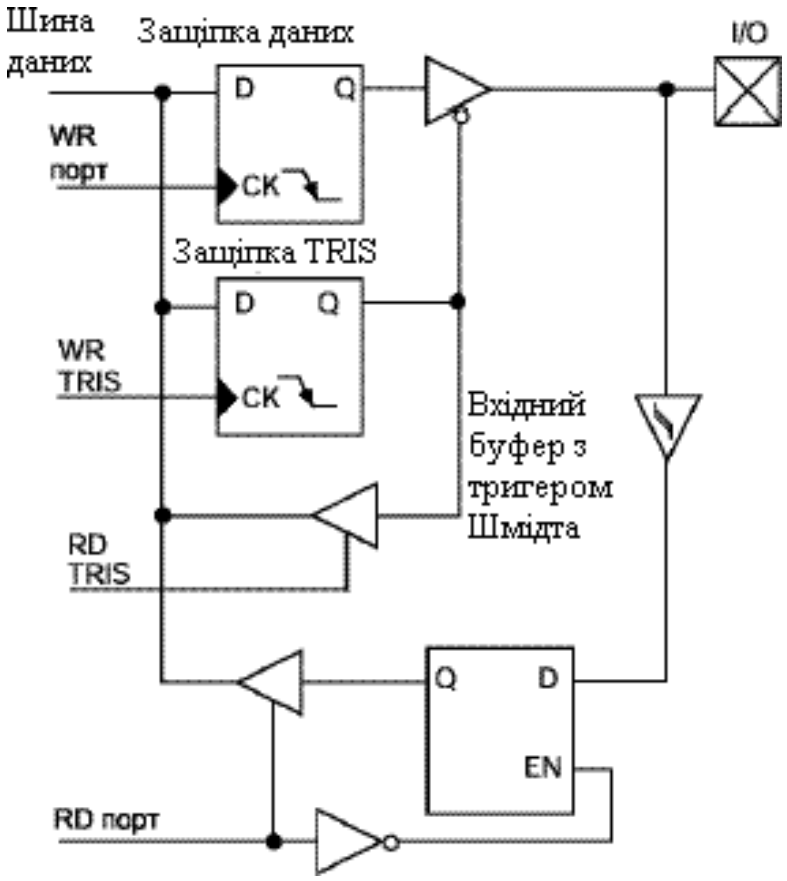


Рисунок 4.9 – Структурна схема виводів порта PORTD

4.9.5 Регістри PORTE і TRISE

PORTE має три виводи (REO/-RD/AN5, RE1/-WR/AN6, RE2/-CS/AN7), що індивідуально настроюються на вхід або вихід. Виводи PORTE мають вхідний буфер Шмідта.

Канали PORTE стануть керуючими виводами веденого паралельного порту, коли біт PSPMODE(TRISE<4>) встановлений в «1». У цьому режимі біти TRISE<2:0> повинні бути встановлені в «1». У регістрі ADCON1 необхідно також настроїти виводи PORTE як цифрові канали введення/виводу. У режимі веденого паралельного порту до виводів PORTE підключені вхідні буфери TTL.

Виводи PORTE мультиплексовані з аналоговими входами. Коли канали PORTE настроєні як аналогові входи, біти регістра TRISE керують напрямом даних PORTE (читання даватиме результат «0»). Структурна схема виводів порта PORTE аналогічна схемі порта PORTD.

4.10 Таймери

Контролер PIC16F877 має три багатофункціональні таймери. Кожен модуль таймера/лічільника може працювати окремо або входити складовою частиною до складу модулів спеціального призначення.

4.10.1 Модуль таймера TMR0

TMR0 – таймер/лічильник, має наступні особливості:

- 8-розрядний таймер/лічильник;
- можливість читання і запису поточного значення лічильника;
- 8-розрядний програмований переддільник;
- внутрішнє або зовнішнє джерело тактового сигналу;
- вибір активного фронту зовнішнього тактового сигналу;
- переривання при переповнюванні (перехід від FFh до 00h).

Блок-схема модуля TMR0 і загального з WDT переддільника показана на рисунку 4.10.

Примітка. Біти управління TOCS, TOSE, PS2, PS1, PSO, PSA розташовані в регістрі OPTION_REG.

Коли біт TOCS скинутий в «0» (OPTION_REG<5>), TMR0 працює від внутрішнього тактового сигналу. Приріст лічильника TMR0 відбувається в кожному машинному циклі (якщо переддільник відключений). Після запису в TMR0 приріст лічильника заборонений два наступні цикли. Користувач повинен скоректувати цю затримку перед записом нового значення в TMR0.

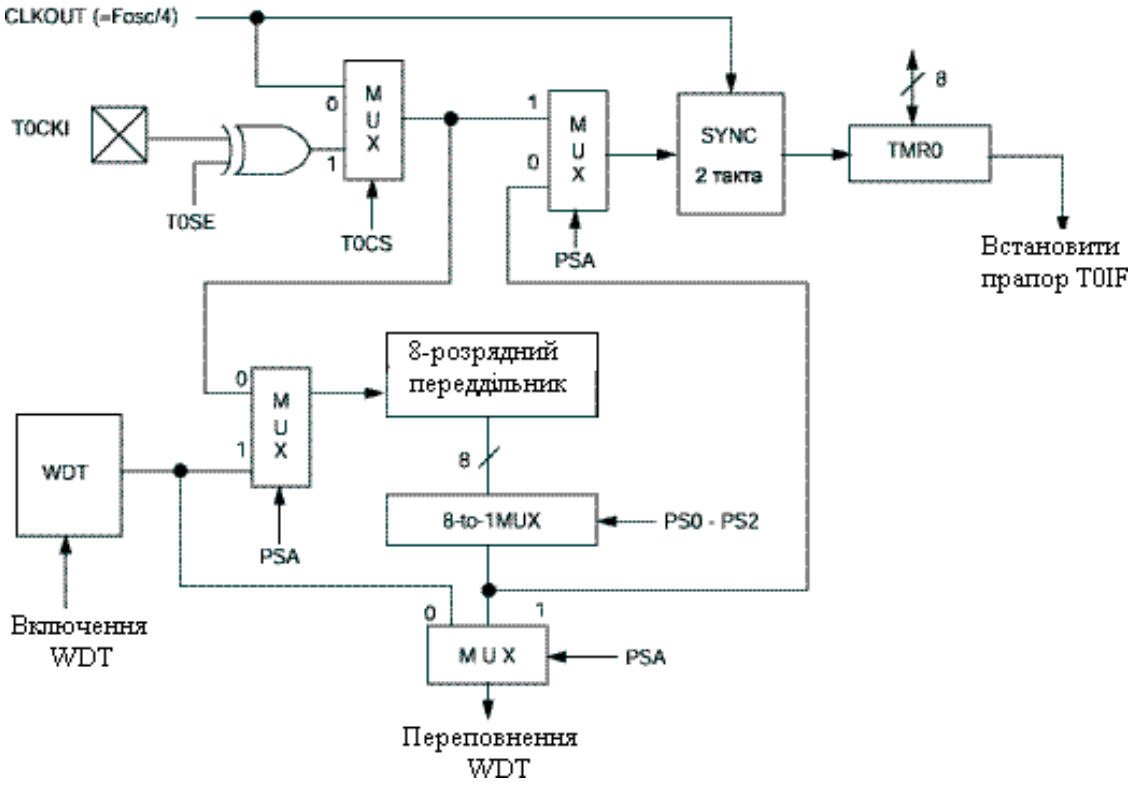


Рисунок 4.10 – Блок-схема модуля TMR0

Якщо біт TOCS встановлений в '1' (OPTION_REG<5>), TMR0 працює від зовнішнього джерела тактового сигналу з боку входу RA4/TOCKI. Активний фронт зовнішнього тактового сигналу вибирається бітом TOSE в регістрі OPTION_REG<4> (TOSE=0 – активним є передній фронт сигналу). Переддільник може бути включений перед WDT або TMR0, залежно від стану біта PSA (OPTION_REG<3>). Не можна прочитати або записати нове значення в переддільник.

Переривання від TMR0 виникають при переповнюванні лічильника, тобто під час переходу його значення від FFh до 00h. При виникненні переривання встановлюється в «1» біт TOIF(INTCON<2>). Само переривання може бути дозволено/заборонено установленням/скиданням біта TOIE (INTCON<5>). Прапор переривання від TMR0 TOIF (INTCON<2>) повинен бути скинутий в підпрограмі обробки переривань. У SLEEP режимі мікроконтролера модуль TMR0 вимкнений і не може генерувати переривання.

Якщо переддільник не використовується, зовнішній тактовий сигнал поступає безпосередньо на синхронізатор. Синхронізація TOCKI з таким сигналом мікроконтролера ускладнюється із-за опиту виходу синхронізатора в машинні цикли Q2 і Q4. Тому тривалість високого або низького логічного рівня зовнішнього сигналу повинна бути не менше 2Tosc (плюс невелика затримка внутрішнього RC ланцюга – 20 нс).

8-розрядний лічильник може працювати як переддільник TMR0 або вихідний дільник WDT. Для простоти опису цей лічильник завжди

називатимемо переддільник. Зверніть увагу, що існує тільки один переддільник, який може бути включений перед TMR0 або WDT. Використання переддільника перед TMR0 означає, що WDT працює без переддільника, і навпаки.

Коефіцієнт ділення переддільника визначається бітами PSA і PS2:PSO в регістрі OPTION_REG<3:0>.

Якщо переддільник включений перед TMR0, будь-які команди запису в TMR0 (наприклад, CLRF 1, MOVWF 1, BSF 1,x і т.д.) скидають переддільника. Коли переддільник підключений до WDT, команда CLRWDT скине переддільника разом з WDT. Переддільник також очищається при скиданні мікроконтролера. Переддільник недоступний для читання/запису.

Примітка. Запис у регістр TMR0 скине переддільника, якщо він підключений до TMR0, але не змінить його режиму роботи.

4.10.2 Модуль таймера TMR1

TMR1 – 16-розрядний таймер/лічильник, що складається з двох 8-розрядних регістрів (TMR1H і TMR1L), доступних для читання і запису. Рахунок виконується в спарених регістрах (TMR1H:TMR1L), при цьому інкрементується їх значення від 0000h до FFFFh, далі рахує з 0000h. При переповнюванні лічильника встановлюється в «1» прапор переривання TMR1IF в регістрі PIR1<0>. Само переривання можна дозволити/ заборонити установленням/скиданням біта TMR1IE в регістрі P1E1<0>. Блок-схема модуля TMR1 показана на рисунку 4.11.

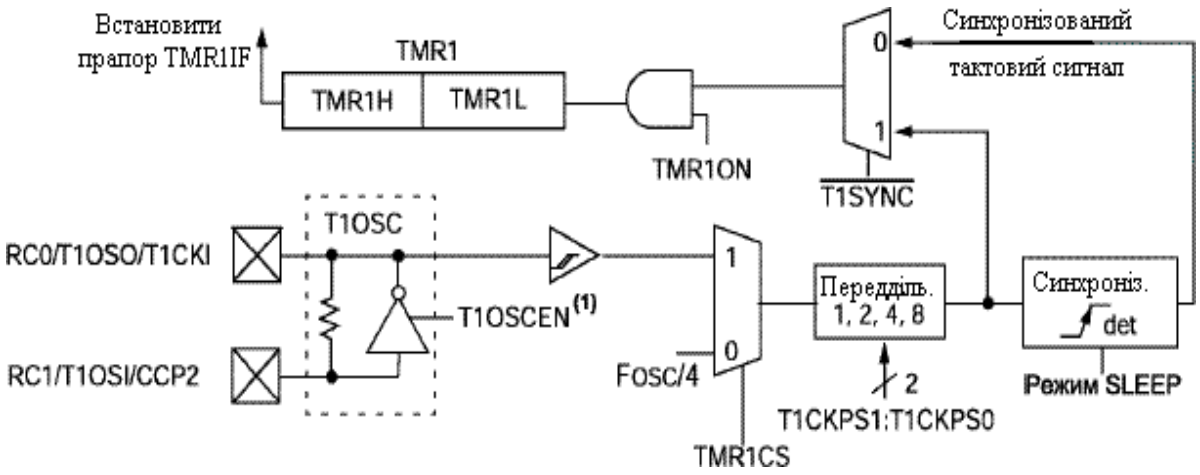


Рисунок 4.11 – Блок схема модуля TMR1

Примітка. Якщо T1OSCEN=0, то інвертуючий елемент і резистивний зворотний зв'язок вимкнені для зменшення струму споживання.

TMR1 може працювати в двох режимах:

- Режим таймера;
- Режим лічильника.

Включення модуля TMR1 здійснюється установкою бита TMR1ON в «1» (T1COM<0>).

Бітом TMR1CS (T1CON<1>) вибирається джерело тактових імпульсів. У режимі таймера TMR1 інкрементується на кожному машинному циклі. Якщо TMR1 працює із зовнішнім джерелом тактового сигналу, то приріст відбувається за кожним переднім фронтом сигналу.

TMR1 має внутрішній вхід скидання від модуля CPP .

Коли включений генератор тактових імпульсів (T1OSCEN=1), виводи RC1/T1OSI/CCP2 і RCO/T1OSO/T1CKI настроєні як входи. Значення бітів TRISC<1:0> ігнорується, а читання даних з цих виводів дає результат «0».

Керуючі биті TMR1 знаходяться в регістрі T1CON (адреса 10h):

b7	b6	b5	b4	b3	b2	b1	b0
-	-	T1CKPS1	T1CKPS0	T1OSCEN	-T1SYNC	TMR1CS	TMR1ON

біти 7-6: не реалізовані: читаються як «0»;

біти 5-4: T1CKPS1:T1CKPS0: вибір коефіцієнта ділення переддільника TMR1 11 = 1:8, 10= 1:4, 01 = 1:2, 00=1:1;

біт 3:T1OSCEN: включення тактового генератора TMR1: 1 = генератор включений, 0 = генератор вимкнений (інвертуючий елемент і резистивний зворотний зв'язок вимкнені для зменшення струму споживання);

біт 2: - T1SYNC: синхронізація зовнішнього тактового сигналу TMR1CS = 1
1= не синхронізувати зовнішній тактовий,
0= синхронізувати зовнішній тактовий,
TMR1CS = 0 : значення біта ігнорується;

біт 1: TMR1CS: вибір джерела тактового сигналу:
1= зовнішнє джерело з виведення RCO/T1OSO/T1CKI (активним є передній фронт сигналу),
0= внутрішнє джерело Fosc/4

біт 0: TMR1ON: Включення модуля TMR1: 1=включений, 0 = вимкнений.

У режимі таймера приріст походить від внутрішнього сигналу Fosc/4, коли біт TMR1CS (T1CON<1>) скинутий в «0». У цьому режимі біт синхронізації T1SYNC (T1COM<2>) ігнорується, тому що внутрішній тактовий сигнал завжди синхронізований.

У режимі лічильника TMR1 може працювати в синхронному або асинхронному режимі залежно від стану бита TMR1CS. Коли TMR1 використовує зовнішній тактовий сигнал, приріст таймера відбувається за переднім фронтом. Включивши TMR1 в режим зовнішнього тактового сигналу, рахунок почнеться тільки після появи заднього фронту.

У режимі синхронного лічильника робота TMR1 від зовнішнього

джерела тактового сигналу вибирається установкою біта TMR1CS в «1». У цьому режимі приріст таймера відбувається за кожним переднім фронтом сигналу на виведенні RC1/T1OSI/CCP2 (якщо T1OSCEN=1) або RCO/T1OSO/T1CKI (якщо T1OSCEN=0).

Якщо -T1SYNC=0, то активний фронт зовнішнього тактового сигналу синхронізується з внутрішнім тактовим сигналом на виході асинхронного переддільника.

У режимі SLEEP мікроконтролера лічильник не буде інкрементуватися (за наявності тактового сигналу), оскільки синхронізатор вимкнений (переддільник продовжує рахунок тактових імпульсів)

У режимі асинхронного лічильника, якщо біт -T1SYNC (T1CON<2>) встановлений в «1», зовнішній тактовий сигнал TMR1 не синхронізується з внутрішнім тактовим сигналом мікроконтролера, таймер продовжує працювати в режимі SLEEP. Переповнювання таймера викличе «пробудження» мікроконтролера, якщо дозволено переривання від TMR1. Проте потрібна обережність при записі/читанні TMR1. У цьому режимі TMR1 не може використовуватися для захоплення/порівняння даних модуля PCP.

Читання TMR1H або TMR1L, під час рахунку в асинхронному режимі, гарантує отримання поточного значення лічильника (реалізовано апаратно). Проте користувач повинен мати на увазі, що читання 16-розрядного значення виконується за кожним байтом. Це накладає деякі обмеження, оскільки таймер може переповнитися між читаннями байтів.

Запис в TMR1 рекомендується виконувати після зупинки таймера. Запис в регістри TMR1 під час приросту таймера може призвести до непередбачуваного значення регістра.

Якщо модуль CCP1 або CCP2 працює в режимі порівняння з трігером спеціальних функцій (CCP1M3 : CCP1 M0=1011), то сигнал трігера скине TMR1. TMR1 повинен працювати в режимі синхронізованого зовнішнього тактового сигналу або внутрішнього тактового сигналу. У асинхронному режимі ця функція не працює.

Коли запис в TMR1 співпадає із сигналом скидання від трігера спеціальних подій, пріоритет віддається запису в TMR1.

У цьому режимі модуля CCP період скидання TMR1 зберігається в регістрах CCPRxH:CCPRxL.

Регістри TMR1H і TMR1L не скидаються в 00h при скиданні по включенню живлення POR і інших видах скидання, окрім скидання по сигналу трігера спеціальних подій модуля CCP1 або CCP2.

Регістр T1CON скидається в 00h при скиданні POR і BOR (TMR1 вимикається, коефіцієнт переддільника рівний 1:1). При решті всіх видів скидання значення регістра T1CON не змінюється.

Переддільник TMR1 очищається при записі в регістр TMR1L або TMR1H.

4.10.3 Модуль таймера TMR2

TMR2 – 8-розрядний таймер з програмованими переддільником і вихідним дільником, 8-розрядним регістром періоду PR2. TMR2 може бути опорним таймером для CCP модуля в режимі ШІМ. Регістри TMR2 доступні для запису/читання і очищаються при будь-якому виді скидання.

Вхідний тактовий сигнал (Fosc/4) поступає через переддільника з програмованим коефіцієнтом ділення (1:1, 1:4 або 1:16), визначуваний бітами T2CKPS1 :T2CKPSO (T2CON<1:0>).

TMR2 рахує, інкрементуючи від 00h до значення в регістрі PR2, потім скидається в 00h на наступному машинному циклі. Регістр PR2 доступний для запису і читання. Після скидання значення регістра PR2 рівне FFh.

Блок-схема модуля TMR2 показана на рисунку 4.12.

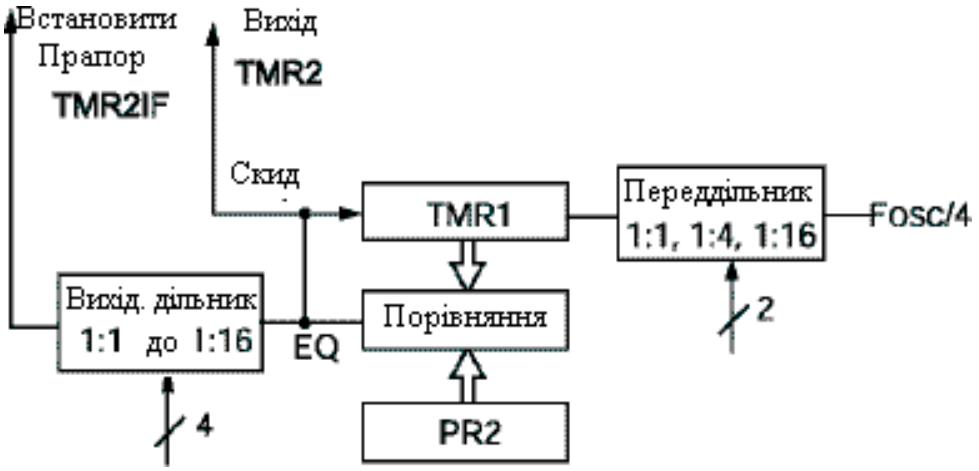


Рисунок 4.12 – Блок-схема модуля TMR2

Сигнал переповнювання TMR2 проходить через вихідний 4-розрядний дільник з програмованим коефіцієнтом ділення (від 1:1 до 1:16 включно) для установлення прапора TMR2IF в регістрі PIR1 <1 >.

Для зменшення енергоспоживання таймер TMR2 може бути вимкнений скиданням біта TMR2ON (T2COM<2>) в «0».

TMR2 може використовуватися для програмного вибору швидкості обміну даними модуля SSP.

Керуючі біти TMR2 знаходяться в регістрі T2CON (адреса 12h):

b7	b6	b5	b4	b3	b2	b1	b0
–	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0

біт 7: не реалізований: читається як «0»;

біти 6-3: TOUTPS3:TOUTPS0: вибір коефіцієнта вихідного дільника TMR2 0000=1:1

0001 = 1:2

0010 = 1:3

1111 =1:16

біт 2: TMR2ON: включення модуля TMR2 1 = включений, 0= вимкнений;

біти 1…0: T2CKPS1 :T2CKPSO: вибір коефіцієнта ділення переддільника TMR2:

00= 1:1

01=1:4

1x= 1:16

Лічильник переддільника і вихідного дільника скидаються у випадку:

- запису в регістр TMR2;
- запису в регістр T2CON;
- будь-якого виду скидання мікроконтролера (POR, BOR, скидання WDT або активний сигнал -MCLR). Регістр TMR2 не очищається при запису в T2CON.

Сигнал переповнювання TMR2 (до вихідного переддільника) надходить до модуля SSP для управління швидкістю передачі даних.

4.11 Модуль 10-розрядного АЦП

Модуль аналого-цифрового перетворення (АЦП) має п'ять каналів у 28-вивідних мікросхемах і вісім каналів у 40/44-вивідних мікросхемах.

Вхідний аналоговий сигнал через комутатор каналів заряджає внутрішній конденсатор АЦП C_{HOLD}. Модуль АЦП перетворить напругу, що утримується на конденсаторі C_{HOLD}, на відповідний 10-розрядний цифровий код методом послідовного наближення. Джерело верхньої і нижньої опорних напруг може бути програмно вибране з виводів Vdd, Vss, RA2 або RA3.

Допускається робота модуля АЦП в режимі SLEEP мікроконтролера, при цьому як джерело тактових імпульсів для АЦП повинен бути вибраний RC-генератор.

Для управління АЦП в мікроконтролері використовується 4 регістри:

- регістр результату ADRESH (старший байт);
- регістр результату ADRESL (молодший байт);
- регістр управління ADCON0;
- регістр управління ADCON1.

Регістр ADCON0 використовується для настройки роботи модуля АЦП, а за допомогою регістра ADCON1 встановлюється, які входи мікроконтролера використовуватимуться модулем АЦП і в якому режимі (аналоговий вхід або цифровий порт уведення/виводу).

ADCON0 (адреса 1Fh)

b7	b6	b5	b4	b3	b2	b1	b0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/-DONE	–	ADON

біти 7-6: ADCS1: ADCS0: Вибір джерела тактового сигналу

00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = Frc (внутрішній RC генератор модуля АЦП)

біти 5-3: CHS2:CHS0: Вибір аналогового каналу
000= канал 0 (RAO/ANO)
001= канал 1(RA1/AN1)
010= канал 2 (RA2/AN2)
011= канал 3 (RA3/AN3)
100= канал 4 (RA5/AN4)
101= канал 5 (RE0/AN5)
110= канал 6 (RE1/AN6)
111 = канал 7 (RE2/AN7)

біт 2: GO/-DONE: Біт статусу модуля АЦП
Якщо ADON=1
1 = модуль АЦП виконує перетворення (установлення біта викликає початок перетворення)
0 = стан очікування (апаратно скидається по завершенні перетворення)

біт 1: Не використовується: читається як '0'

біт 0: ADON: Біт включення модуля АЦП
1= модуль АЦП включений
0 = модуль АЦП вимкнений і не споживає струм.

ADCON1 (адреса 9Fh)

b7	b6	b5	b4	b3	b2	b1	b0
ADFM	–	–	–	PCFG3	PCFG2	PCFG1	PCFG0

біт 7: ADFM: Формат збереження 10-розрядного результату
1 = праве вирівнювання, 6 старших біт ADRESH читаються як '0'
0 = ліве вирівнювання, 6 молодших біт ADRESL читаються як '0'

біти 6-4: Не використовуються: читаються як '0';

биті 3-0: PCFG3:PCFG0: Керівні біти настройки каналів АЦП їх значення приведене в таблиці 4.3.

Таблиця 4.3 – Біти настройки каналів АЦП

PCGF3: PCGF0	AN7 RE2	AN6 RE1	AN5 RED	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AND RAO	V_{REF+}	V_{REF-}	Кан./ V_{REF}
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
0000	A	A	A	A	A	A	A	A	V_{DD}	V_{SS}	8/0
0001	A	A	A	A	V_{REF+}	A	A	A	RA3	V_{SS}	7/1
0010	D	D	D	A	A	A	A	A	V_{DD}	V_{SS}	5/0
0011	D	D	D	A	V_{REF+}	A	A	A	RA3	V_{SS}	4/1
0100	D	D	D	D	A	D	A	A	V_{DD}	V_{SS}	3/0
0101	D	D	D	D	V_{REF+}	D	A	A	RA3	V_{SS}	2/1
011x	D	D	D	D	D	D	D	D	V_{DD}	V_{SS}	0/0

Продовження таблиці 4.3

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	Vss	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = аналоговий вхід; D = цифровий канал введення/виводу.

В останньому стовпці указується кількість аналогових каналів, доступних для перетворення і кількість входів джерела опорної напруги.

У регістрі ADRESH:ADRESL зберігається 10-розрядний результат аналого-цифрового перетворення. Коли перетворення завершене, результат перетворення записується в регістр ADRESH:ADRESL, після чого скидається прапор GO/-DONE (ADCON0<2>) і встановлюється прапор переривання ADIF. Структурна схема модуля АЦП показана на рисунку 4.13.

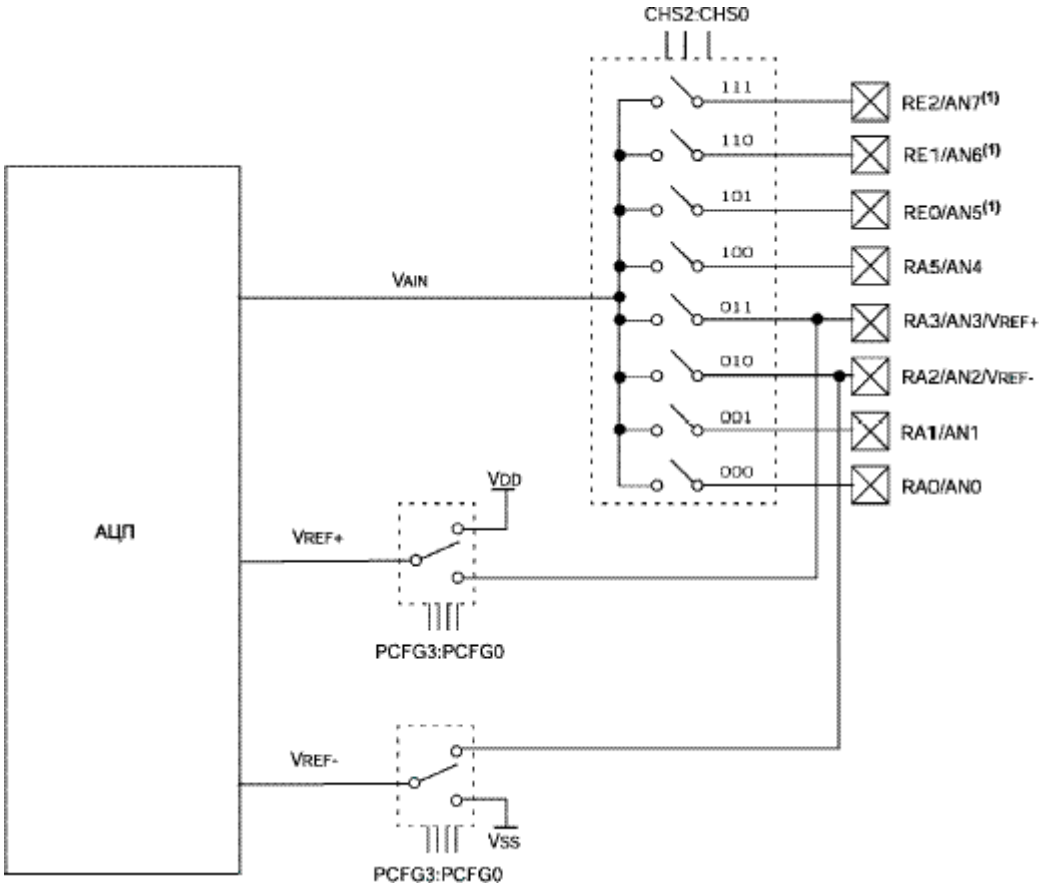


Рисунок 4.13 – Структурна схема модуля АЦП

Після включення і конфігурації АЦП вибирається робочий аналоговий канал. Відповідні біти TRIS аналогових каналів повинні настроювати порт введення/виводу на вхід.

Рекомендована послідовність дій для роботи з АЦП:

1 Настроїти модуль АЦП:

- настроїти виводи як аналогові входи, входи VREF або цифрові канали введення/виводу (ADCON1);
- вибрати вхідний канал АЦП (ADCON0);
- вибрати джерело тактових імпульсів для АЦП (ADCON0);
- включити модуль АЦП (ADCON0).

2 Настроїти переривання від модуля АЦП (якщо необхідно):

- скинути біт ADIF в «0»;
- встановити біт ADIE в «1»;
- встановити біт PEIE в «1»;
- встановити біт GIE в«1».

3 Витримати паузу, необхідну для зарядження конденсатора CHOLD.

4 Почати аналого-цифрове перетворення:

•установити біт GO/-DONE в «1» (ADCON0).

5 Чекати закінчення перетворення:

- чекати поки біт GO/-DONE не буде скинутий в «0»;
- чекати переривання по закінченні перетворення.

6 Прочитати результат перетворення з регістрів ADRESH: ADRESL, скинути біт ADIF в '0', якщо це необхідно.

7 Для наступного перетворення необхідно виконати кроки, починаючи з пункту 1 або 2. Час перетворення одного біта визначається як час TAD. Мінімальний час очікування перед наступним перетворенням повинен становити не менше 2TAD.

4.12 Переривання

PIC16F87X мають 14 джерел переривань. Регістр INTCON містить прапори окремих переривань, біти дозволу цих переривань і біт глобального дозволу переривань.

Якщо біт GIE (INTCON<7>) встановлений в «1», дозволені всі немасковані переривання. Якщо GIE=0, то всі переривання заборонені. Кожне переривання окремо може бути дозволене/заборонене установленням/скиданням відповідного біта в регістрах INTCON, PIE1 і PIE2. При скиданні мікроконтролера біт GIE скидається в «0». При поверненні з підпрограмі обробки переривання, за командою RETFIE, біт GIE апаратно встановлюється в «1», дозволяючи всі немасковані переривання.

У регістрі INTCON знаходяться прапори наступних переривань: зовнішнього сигналу INT, зміни рівня сигналу на входах RB7:RB4, переповнювання TMR0.

У регістрах PIR1, PIR2 містяться прапори переривань периферійних модулів мікроконтролера, а в регістрах PIE1, PIE2 – відповідні біти дозволу переривань. У регістрі INTCON знаходиться біт дозволу переривань від периферійних модулів.

При переході на підпрограму обробки переривань біт GIE апаратно скидається в «0», забороняючи переривання, адреса повернення з підпрограми обробки переривань поміщається в стек, а в лічильник команд PC завантажується вектор переривання 0004h. Джерело переривань може бути визначене перевіркою прапорів переривань, які повинні бути скинуті програмно перед дозволом переривань, щоб уникнути повторного виклику.

Для зовнішніх джерел переривань (сигнал INT, зміни рівня сигналу на входах RB7:RB4) час переходу на підпрограму обробки переривань становитиме 3…4 машинних цикли. Точний час переходу залежить від конкретного випадку, він однаковий для 1- і 2-циклових команд. Прапори переривань встановлюються незалежно від стану відповідних бітів маски і біта GIE.

Примітка. Індивідуальні прапори переривань встановлюються незалежно від стану відповідних бітів маски і біта GIE.

Структурна схема логіки переривань показана на рисунку 4.14.

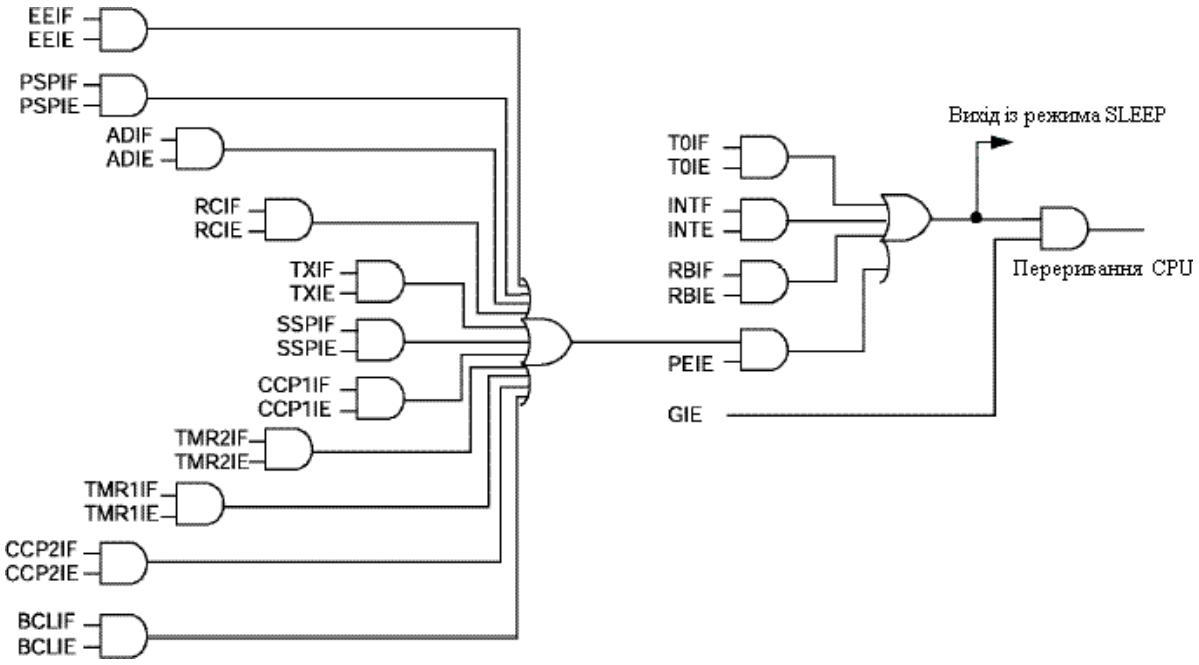


Рисунок 4.14 – Структурна схема логіки переривань

Зовнішнє переривання з входу RB0/INT відбувається: за переднім фронтом сигналу, якщо біт INTEDG (OPTION_REG<6>) встановлений в «1»; за заднім фронтом сигналу, якщо біт INTEDG скинутий в «0». Коли

активний фронт сигналу з'являється на вході RB0/INT, біт INTF (INTCON<1>) встановлюється в «1». Переривання може бути заборонене скиданням біта INTE (INTCON<4>) в «0». Прапор переривання INTF повинен бути скинутий програмно в підпрограмі обробки переривань. Переривання INT може вивести мікроконтролер з режиму SLEEP, якщо біт INTE=1 перед переходом до режиму SLEEP. Стан біта GIE визначає, переходити на підпрограму обробки переривань після виходу з режиму SLEEP чи ні.

Переповнювання таймера TMR0 (FFh -> 00h) встановлює прапор TOIF (INTCON<2>) в«1». Переривання від TMR0 можна дозволити/заборонити установленням/скиданням біта TOIE (INTCON<5>). Зміна рівня сигналу на входах RB7:RB4 викликає установлення прапора RBIF (INTCON<0>). Переривання можна дозволити/заборонити установленням/скиданням біта RBIE (INTCON<4>).

4.13 Сторожовий таймер WDT

Убудований сторожовий таймер WDT працює від окремого генератора RC, що не вимагає зовнішніх компонентів. Це дозволяє працювати сторожовому таймеру WDT при вимкненому тактовому генераторі (виводи OSC1, OSC2) в режимі SLEEP мікроконтролера. У нормальному режимі роботи при переповнюванні WDT відбувається скидання мікроконтролера. Якщо мікроконтролер знаходиться в режимі SLEEP, переповнювання WDT виводить його з режиму SLEEP з продовженням нормальної роботи. WDT вимкнений, якщо WDTE = 0 в слові конфігурації.

Час переповнювання залежить від температури, напруги живлення V_{DD} і розкиду технологічних параметрів мікроконтролера. Якщо потрібний більший час переповнювання WDT, необхідно програмно підключити переддільник в регістрі OPTION_REG з максимальним коефіцієнтом ділення 1:128.

Примітки:

1 Команди CLRWDT і SLEEP скидають сторожовий таймер і переддільника, якщо він підключений до WDT, відкладаючи скидання пристрою.

2 Команда CLRWDT скидає сторожовий таймер і переддільник, якщо він підключений до WDT, але не змінює коефіцієнт ділення переддільника.

Структурна схема сторожового таймера WDT показана на рисунку 4.15.

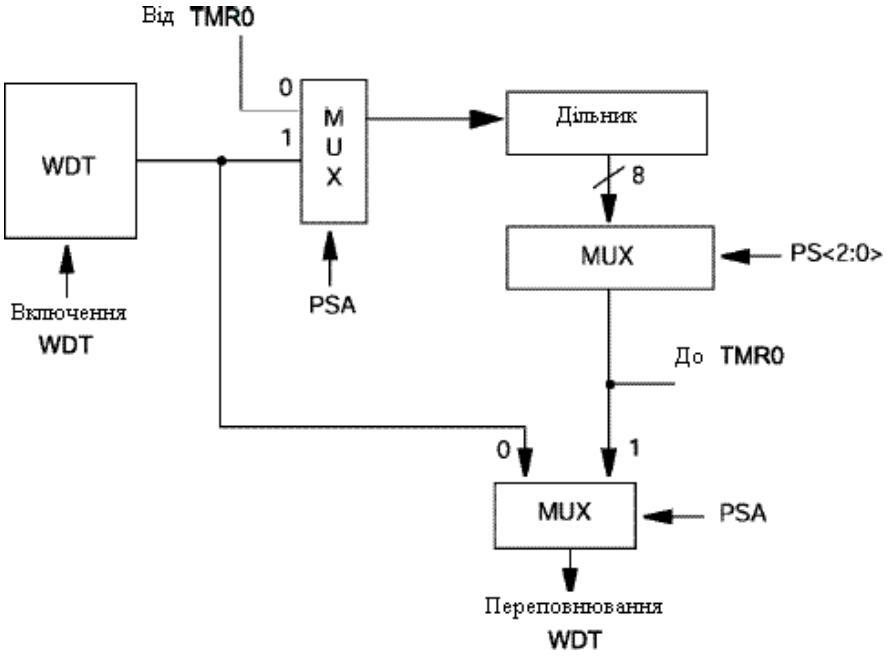


Рисунок 4.15 – Структурна схема сторожового таймера WDT

СПИСОК ВИКОПИСТАНОЇ ЛІТЕРАТУРИ

1 Локазюк, В. М. Мікропроцесори та мікроЕОМ у виробничих системах : посібник. – К. : Академія, 2002. – 368 с. – ISBN 966-580-130-9.

2 Микропроцессорные системы : учебное пособие для вузов / Е. К. Александров [и др.]; под общ. ред. Д. В. Пузанкова. – С Пб. : Политехника, 2002. – 935 с. – ISBN 5-7325-0516-4.

3 Предко, М. Справочник по PIC-микроконтроллерам : пер. с англ. – М. : ДМК Пресс, 2004. – 512 с. – ISBN 5-94076-116-

4 Оксанич А.П., Притчин С.Є., Вашерук О.В. Комп'ютерна електроніка. Ч. I-II.– К.: Вища школа, 2005, 456 с.

5 Бех І.І., Левитський С.М. Фізичні основи комп'ютерної електроніки. – К.:ТОВ "Карбон", 2010. – 233 с.

6 Електротехніка, електроніка та мікропроцесорна техніка / М. С. Будіщев; Ред.Мельников О.В. – Львів: Афіша, 2001. – 424 с.

УДК 621.382:004(076.5)
М-23

Тарас Григорович Бенько

Мікроконтролери :Навчально – методичний посібник.–Івано-Франківськ, Прикарпатський національний університет, 2023. – 151 с.

…….....
76018, м. Івано-Франківськ, вул. Шевченка, 57
Прикарпатський національний університет імені Василя Стефаника