*Kpi-best*

*Міністерство освіти та науки України*

*Національний технічний університет України «Київський політехнічний інститут ім. Ігоря Сікорського»*

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

*Курсовий проект*

*з дисципліни*

*«Архітектура комп’ютера»*

*Виконав:*

*Студент Антонов Є.А.*

*Група ІО-51*

*ФІОТ  
Залікова книжка № 5101*

*Номер технічного завдання 101 101*

*Прийняла:*

*Ткаченко В.В.*

*Київ*

*2018*

Опис альбому

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N п.п. | Формат | Позначення | | | Найменування | К. сторін | | № экз. | | Примітки | | |
|  |  |  | | | Документація загальна |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
| 1 | А4 |  | | | Завдання на  дипломний проект | 2 | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
| 2 | А4 | ІАЛЦ 466454.001 ОА | | | Опис альбому | 1 | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
| 3 | А4 | ІАЛЦ 466454.002 ПЗ | | | Пояснювальна записка |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
| 4 | А3 | ІАЛЦ 466454.003 Е1 | | | Схема електрична  структурна | 1 | |  | |  | | |
| 5 | А3 | ІАЛЦ 466454.004 Е1 | | | Схема електрична  структурна | 1 | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  | | |  |  | |  | |  | | |
|  |  |  |  |  | ДП ІАЛЦ 462617.001 ОА | | | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум. | Підпис | Дата |
| Виконав | | Антонов Є.А. |  |  | Опис альбому | | Лит. | | | | Лист | Листів |
| Перевірив | |  |  |  |  | |  |  | 1 | 1 |
|  | |  |  |  | НТУУ «КПІ» ФІОТ  ФІОТ Гр. ІО-51 | | | | | |
| *Н. контр.* | |  |  |  |
| *Зав. каф.* | |  |  |  |

*Пояснювальна записка*

*Зміст*

[***ВСТУП*** *2*](#_Toc514749353)

[***1.******Огляд МК по різним фірмам виробників*** *3*](#_Toc514749354)

[*1.1. Архітектура ARM 4*](#_Toc514749355)

[*1.2. Мікроконтролери виробництва Atmel 4*](#_Toc514749356)

[*1.3. Мікроконтролери виробництва Microchip Technology Inc. 5*](#_Toc514749357)

[*1.4. Мікроконтролери виробництва Intel 5*](#_Toc514749358)

[***2.******Архітектура створюваної системи*** *8*](#_Toc514749360)

[*2.1. Система команд 15*](#_Toc514749361)

[*2.3. Карта розподілу пам’яті 18*](#_Toc514749362)

[*2.4. Селектори адрес для зовнішніх пристроїв та ЗПД 19*](#_Toc514749363)

[*2.5. Програмований переферійний адаптер КР580ВВ55. 20*](#_Toc514749364)

[*2.5.1. Опис режимів роботи 21*](#_Toc514749365)

[*2.5.1.1. Режим 0 (просте введення / виведення) 21*](#_Toc514749366)

[*2.5.1.2. Режим 1 (періодичного введення / виведення) 21*](#_Toc514749367)

[*2.5.1.3. Режим 2 (двонаправлений тактируемого введення / виведення) 22*](#_Toc514749368)

[*2.6. Таймер 22*](#_Toc514749369)

[*2.7. Програмний режим обміну 24*](#_Toc514749370)

[*2.8. КПП. Реалізація переривань. 29*](#_Toc514749371)

[*2.9. КПДП 35*](#_Toc514749372)

[***3.******Спеціалізований обчислювач*** *40*](#_Toc514749373)

[*3.1. Аналіз рішення аглоритму заданої задачі 40*](#_Toc514749374)

[*3.2. Розробка аглоритму задачі 41*](#_Toc514749375)

[*3.3. Система команд 42*](#_Toc514749376)

[***4.******Опис функціональної та принципової схеми заданого блоку*** *42*](#_Toc514749377)

[***5.******Розробка вузла ЕОМ на ПЛІС*** *42*](#_Toc514749378)

[***6.******Побудова заданої задачі на ПЛІС*** *51*](#_Toc514749379)

[***Висновки*** *54*](#_Toc514749380)

[***Список використаної літератури*** *55*](#_Toc514749381)

# *ВСТУП*

*Мікроконтролер - мікросхема, призначена для керування електронними пристроями. Типовий мікроконтролер поєднує в собі функції процесора і периферійних пристроїв, може містити ОЗП і ПЗП. Велика частина процесорів, що випускається в сучасному світі - мікроконтролери.*

*Основною класифікаційною ознакою мікроконтролерів є розрядність даних, які оброблюються арифметико-логічним пристроєм (АЛП). За цією ознакою вони поділяються на 4 -, 8 -, 16 -, 32 - і 64-розрядні.*

*При проектуванні мікроконтролерів доводиться дотримувати баланс між розмірами і вартістю з одного боку і гнучкістю і продуктивністю з іншого. Для різних додатків оптимальне співвідношення цих та інших параметрів може розрізнятися дуже сильно. Тому існує величезна кількість типів мікроконтролерів, що відрізняються архітектурою процесорного модуля, розміром і типом вбудованої пам'яті, набором периферійних пристроїв, типом корпусу і т. д.*

*Неповний список периферії, яка може бути присутня в мікроконтролерах, включає в себе:*

* *універсальні цифрові порти*
* *різні інтерфейси введення-виведення*
* *аналого-цифрові і цифро-аналогові перетворювачі*
* *широтно-імпульсні модулятори*
* *таймери, вбудований тактовий генератор і таймер*
* *контролери без колекторних двигунів*
* *контролери дисплеїв і клавіатур*
* *радіочастотні приймачі та передавачі*
* *масиви вбудованої флеш-пам'яті*

*Програмування мікроконтролерів зазвичай здійснюється на мові асемблера або Сі. Для налагодження програм використовуються програмні симулятори (спеціальні програми для персональних комп'ютерів, що імітують роботу мікроконтролера) внутрішньосхемний емулятор і інтерфейс JTAG.*

1. ***Огляд МК по різним фірмам виробників***

*З 80-х років XX століття в мікропроцесорній техніці виділився самостійний клас інтегральних схем – однокристальні мікроконтролери, які призначені для вбудовування в прилади різного призначення. Від класу однокристальних мікропроцесорів їх відрізняє наявність внутрішньої пам’яті, розвинені засоби взаємодії з зовнішніми пристроями.*

*Основною ознакою в їх класифікації є розрядність. В даний час промисловістю випускаються 4-, 8-, 16- і 32-розрядні. Однак зустрічаються і вельми екзотичні варіанти. Так, MC14500 фірми Motorola, набір команд якого складається з 16 інструкцій, має розрядність 1 біт і здатний працювати з необмеженим обсягом пам’яті.*

*Мікроконтролери можна диференціювати також за ступенем універсальності. Випускаються як системи широкого застосування з програмованою пам’яттю, так і вузькоспеціалізовані контролери для певної області, або навіть для конкретного пристрою, часто з вже записаною в чіп програмою. Однак випуск таких мікроконтролерів економічно виправданий лише при масовому виробництві.*

*В даний час випускається цілий ряд типів мікроконтролерів. Всі ці прилади можна умовно розділити на три основні класи:*

* *8-розрядні для вбудованих додатків;*
* *16- і 32-розрядні;*
* *цифрові сигнальні процесори (DSP).*

*Найбільш поширеним представником сімейства є 8-розрядні прилади, широко використовуються в промисловості, побутовій та комп’ютерній техніці. Вони пройшли в своєму розвитку шлях від найпростіших приладів з відносно слаборозвиненою периферією до сучасних багатофункціональних контролерів, які забезпечують реалізацію складних алгоритмів керування в реальному часі. Причиною їх життєздатності є використання для управління реальними об’єктами, де застосовуються, в основному, алгоритми з переважанням логічних операцій, швидкість обробки яких практично не залежить від розрядності процесора. Популярності 8-розрядних мікроконтролерів також сприяє постійне розширення номенклатури виробів, що випускаються відомими фірмами, такими як Motorola, Microchip, Intel, Zilog, Atmel і багатьма іншими.*

* 1. *Архітектура ARM*

*Архітектура ARM (раніше Advanced RISC Machine – вдосконалена RISC-машина, попередник – Acorn RISC Machine) – 32-бітова архітектура зі скороченим набором команд, що розробляється ARM Limited. Дані процесори мають низьке енергоспоживання, тому знаходять широке застосування у вбудованих системах і домінують на ринку мобільних пристроїв, для яких це важливо.*

*У 2007 році близько 98 відсотків з більш ніж мільярда мобільних телефонів, продаваних щорічно, були оснащені принаймні одним процесором ARM. Станом на 2009 рік на процесори ARM припадає до 90% всіх вбудованих 32-розрядних RISC процесорів. Процесори ARM широко використовуються в споживчій електроніці – в тому числі КПК, мобільних телефонах, цифрових носіях і плеєрах, портативних ігрових консолях, калькуляторах і комп’ютерних периферійних пристроях, таких як жорсткі диски або маршрутизатори.*

* 1. *Мікроконтролери виробництва Atmel*

*Справжня революція в світі мікроконтролерів сталася в 1996 році, коли корпорація Atmel представила своє сімейство чіпів на новому прогресивному ядрі AVR. Мікропроцесори AVR мають більш розвинену систему команд, що налічує до 133 інструкцій, продуктивність, що наближається до 1 MIPS/МГц, Flash пам’ять програм з можливістю внутрішньосхемного перепрограмування. Багато чіпів мають функцію самопрограмування. AVR-архітектура оптимізована під мову високого рівня C. Крім того, всі кристали сімейства сумісні “знизу вгору”.*

*Величезну роль в їх поширенні зіграла доступність програмного забезпечення і засобів підтримки розробки. У Atmel багато безкоштовно розповсюджуваних програмних продуктів. Добре відомо, що розвинені засоби підтримки розробок при освоєнні і знайомстві з будь-яким мікроконтролерним сімейством грають не менш значущу роль, ніж самі кристали. Фірма Atmel приділяє цьому питанню велику увагу.*

* 1. *Мікроконтролери виробництва Microchip Technology Inc.*

*PIC – мікроконтролери гарвардської архітектури, вироблені американською компанією Microchip Technology Inc. Назва PIC є скороченням від Peripheral Interface Controller, що означає “периферійний інтерфейсний контролер”. Назва пояснюється тим, що спочатку PIC призначалися для розширення можливостей введення-виведення 16-бітних мікропроцесорів CP1600.*

**

*Рисунок 1.1 – Мікроконтролер PIC з “віконцем” для перепрограмування.*

*У номенклатурі Microchip Technology Inc. представлений широкий спектр 8-и, 16-и і 32-бітних мікроконтролерів та цифрових сигнальних контролерів під маркою PIC. Їх відмінною особливістю є хороша наступність різних сімейств. Є і програмна сумісність (єдина безкоштовне середовище розробки MPLAB IDE), сумісність по периферії, по напругам живлення, по бібліотеках. Номенклатура налічує більше 500 різних контролерів зі різними варіаціями периферії, пам’яті, продуктивністю, діапазонами живлення і температури і т. д.*

## *Мікроконтролери виробництва Intel*

*Біля витоків виробництва мікроконтролерів стоїть фірма Intel з родинами восьмирозрядних мікроконтролерів 8048 і 8051. Архітектура MCS-51 отримала свою назву від першого представника цього сімейства – мікроконтролера 8051, випущеного в 1980 році на базі технології HMOS. Вдалий набір периферійних пристроїв, можливість гнучкого вибору зовнішньої або внутрішньої програмної пам’яті і прийнятна ціна забезпечили цьому мікроконтролеру успіх на ринку. З точки зору технології МК-51 був для свого часу дуже складним виробом – у кристалі було використано 128 тис. транзисторів, що в 4 рази перевищувало кількість транзисторів в 16-розрядному мікропроцесорі 8086. Загалом фірма Intel випустила близько 50 моделей на базі операційного ядра МК-51. Одночасно багато інших фірм, такі як Atmel, Philips, почали виробництво своїх мікроконтролерів, розроблених в стандарті MCS-51. Існує також і радянський аналог Intel 8051 – мікросхема К1816ВЕ51.*

*Структура мікроконтролера МК-51:*

* *восьмирозрядний центральний процесор, оптимізований для реалізації функцій управління;*
* *вбудований тактовий генератор (максимальна частота 12 МГц);*
* *адресний простір пам’яті програм – 64 Кб;*
* *адресний простір пам’яті даних – 64 Кб;*
* *внутрішня пам’ять програм – 4 Кб;*
* *внутрішня пам’ять даних – 128 байт;*
* *додаткові можливості по виконанню операцій булевої алгебри (побітові операції);*
* *32 двонаправлені і індивідуально адресовані лінії введення/виведення;*
* *два 16-розрядних багатофункціональних таймера/лічильника;*
* *повнодуплексний асинхронний приймач (послідовний порт);*
* *векторна система переривань з двома рівнями пріоритету і п’ятьма джерелами подій.*

**

*Рисунок 1.2 – Мікроконтролер Intel*

*Спочатку найбільш “вузькими” місцями архітектури MCS-51 були 8-розрядний арифметико-логічний пристрій на базі акумулятора і відносно повільне виконання інструкцій (для виконання найшвидших інструкцій потрібно 12 тактів). Це обмежувало застосування мікроконтролерів сімейства в додатках, що вимагають підвищеної швидкодії і складних обчислень (16- і 32- бітових). Нагальним стало питання принципової модернізації старої архітектури. Проблема ускладнювалася тим, що до початку 90-х років вже була створена маса напрацювань у галузі програмного і апаратного забезпечення, і однією з основних задач розробки нової архітектури була реалізація апаратної і програмної сумісності зі старими розробками на базі МК-51. Для вирішення цього завдання була створена спільна група з фахівців компаній Intel і Philips. У результаті в 1995 р. з’явилося 2 сімейства: MCS-251/151 в Intel і 51XA у Philips.*

*Основні характеристики архітектури MSC-251:*

* *24-розрядний лінійний адресний простір, що забезпечує адресацію до 16 Mб пам’яті;*
* *система команд мікроконтролерів сімейства MCS-251 містить всі 111 команд, що входять в систему команд сімейства MCS-51 (“старі” команди), і, крім того, в неї входять 157 “нових” команд. Коди деяких нових команд мають формат 4 байти.*
* *перед використанням мікроконтролера його необхідно конфігурувати, тобто за допомогою програматора “пропалити” конфігураційні байти, що визначають, який з наборів інструкцій стане активним після включення живлення;*
* *регістрова архітектура, яка припускає звернення до регістрів як до байтів, слів і подвійних слів;*
* *сторінковий режим адресації для прискорення вибірки інструкцій з зовнішньої програмної пам’яті;*
* *розширений набір команд, що включає 16-бітові арифметичні і логічні інструкції;*
* *розширений адресний простір стеку до 64 Кб;*
* *виконання найшвидшої інструкції за 2 такти;*
* *сумісність на рівні двійкового коду з програмами для MCS-51.*

*Структурна схема розроблювальної мікропроцесорної системи представлена на кресленику ІАЛЦ 462416.005 Е1, що знаходиться в додатках.*

1. ***Архітектура створюваної системи***

*Під архітектурою розуміють конфігурацію основних компонентів системи з врахуванням їх можливостей, здібностей, способу взаємодії і розділення функцій між ними. В нашому випадку, розробка архітектури розроблюваної системи складається з вибору необхідних компонентів, його обґрунтування та налагодження між ними необхідних зв’язків.*

*Відповідно до технічного завдання, розроблювана мікропроцесорна система має наступні компоненти:*

* + *Процесорне ядро на базі мікроконтролера PIC16C71;*
  + *Зовнішня пам’ять даних, що складається з 16 сторінок об’ємом по 128Б;*
  + *128 зовнішніх пристрої, кожен з яких має відведений для нього діапазон адресів для регістру даних та регістру стану;*
  + *Централізовані контролер пріоритетних переривань та прямого доступу до пам’яті;*
  + *Спецобчислювач має адресуh та виконує додаткові операції;*
  + *Програмний периферійний адаптер К580ВВ55 – має виділений діапазон адрес.*

*Всі вищенаведені пристрої об’єднані за допомогою системної шини – магістралі обміну даними. В даній роботі вона складається з 16 провідників та розділена на шину даних і адреси (по 8 розрядів), що дозволяє не використовувати тимчасове мультиплексування при передачі адреси і даних.*

*Структура мікроконтролера PIC16С71*

*Структурна схема PIC наведена на рис.1. Мікроконтролер містить пам'ять програм 1К\*14, пам'ять даних 36\*8, аккумулятор (регістр W), таймер, 8-рівненвий стек, порти вводу/виводу.*

*Основні характеристики*

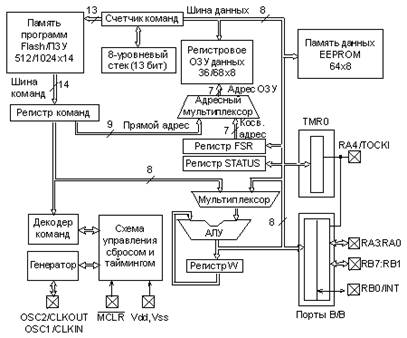
* *тільки 35 простих команд;*
* *усі команді виконуються за один цикл крім команд переходу;*
* *рабоча частота 0 Гц ... 20 МГц*
* *14- бітові команди;*
* *8- бітові дані;*
* *2К х 14 пам’яті програм;*
* *36 х 8 регістрів загального використання;*
* *15 спеціальних аппаратних регістрів SFR;*
* *36 x 8 EEPROM пам’яті для даних;*
* *восьмирівненвий аппаратный стек;*
* *пряма, опосередкована і відносна адресація даних і команд;*
* *чотири джерела переривань :* 
  + *зовнішній вхід INT;*
  + *переповнення таймера TMR0;*
  + *при зміні сигналів на лініях порту B;*
  + *по завершенню запису даних в память EEPROM;*

*Периферія и Ввід/Вивід*

* *13 ліній вводу-виводу з індивідуальними налаштуваннями;*
* *вхідний/вихідний струм для управлення світдодіодами.*
* *макс. вхідний струм - 20 мА. ,*
* *макс. вихідний струм - 25 мА.,*
* *TMR0: 8 - бітний таймер/лічильник TMR0 з 8-бітним програмованим попереднім дільником.*

##### *Спеціальні властивості*

* *Автоматический скидання при запуску;*
* *таймер включення при скиданні;*
* *таймер запуску генератора;*
* *EEPROM біт секретності для захисту коду;*
* *економічний режим SLEEP;*
* *доступ до вибору бітів для установки режиму збудження внутрішнього генератора:*
* *RC генератор : RC;*
* *звичайний кварцевый резонатор : XT;*
* *високочастотний кварцевий резонатор : HS;*
* *економічний низкочастотный кристал : LP;*

**

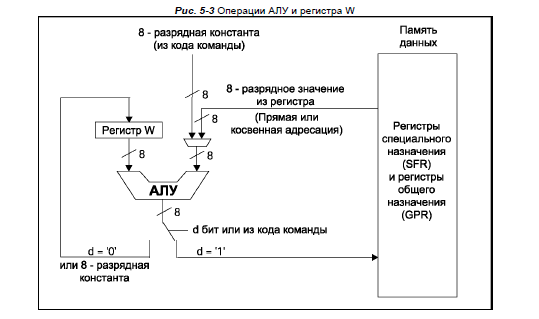
*Рис.2.1 - Структурна схема мікроконтролера PIC16C71*

***Акумулятор***

*Акумулятор W являється восьмирозрядним регістром, який використовується в якості приймача або джерела операнда.*

***Арифметико-логічний пристрій***

*Мікроконтролер містить 8-розрядний універсальний арифметичний модуль і 8-розрядний робочий регістр. АЛП виконує арифметичні і булеві операції між робочим регістром і будь-яким регістром пам’ті даних*

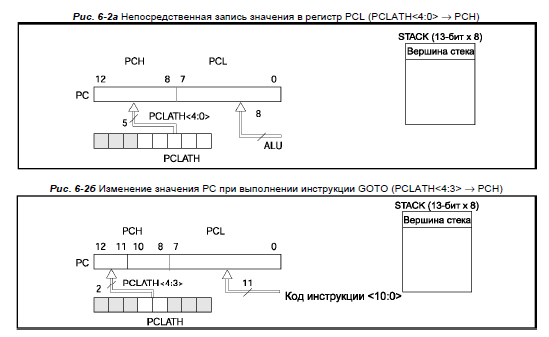
**

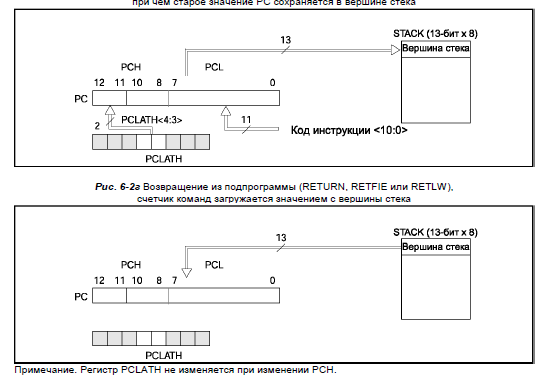
*Рис.2.2 - Структурна схема АЛП*

*8-розрядний АЛП може виконувати додавання, віднімання, порозрядний зсув и логічні операції. Арифметичні операції виконуюються по принципу доповнення до двух. В командах з двума оперантами, перший знаходиться в регісті-аккамуляторі, а інший в регістрі пам’яті даних або константа. В командах з одним оперантом – аккамулятор або регістр.*

***Лічильник команд***

*13-розрядний регістр лічильника команд РС вказує адрес вибраної команди для виконання. Молодший байт лічильника програм PCL доступний для читання та запису. Старший байт PCH, що містить з 12 по 8 біти не доступний. Усі операції з регістром PCH виконуються через додатковий регістр PCLATH.*

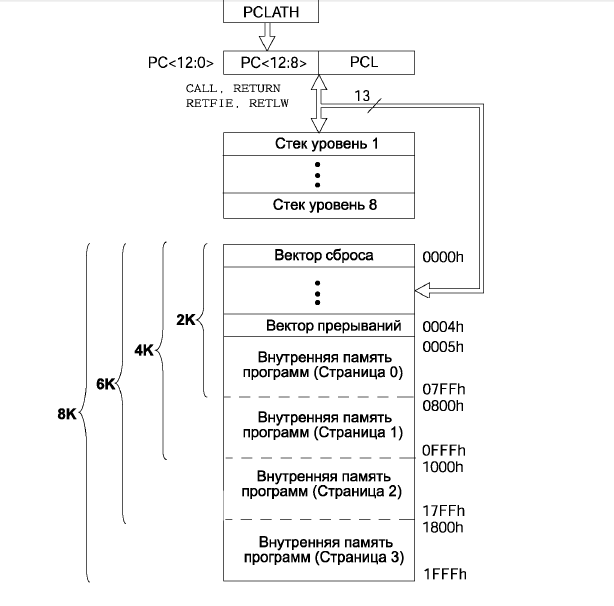
**

**

*Рис. 2.3. - Можливі варіанти адресації*

***Пам’ять програм***

*Лічильник команд в МК PIC16C7ХX здатен адресувати 8Кх14 біт об’ємом програмної пам’яті. Однак фізично на кристалах PIC16C71 – 1Кх14 памяти (адреса 0000h-03FFh).**Звернення до адрес вище 1FFh (3FFh) фактично є**адресація в ті ж перші 1К адреси.* *В памяті програм є виділені адреси. Вектор скидання знаходиться за адресою 0000h, вектор**прерывания – по адресу 0004h.*

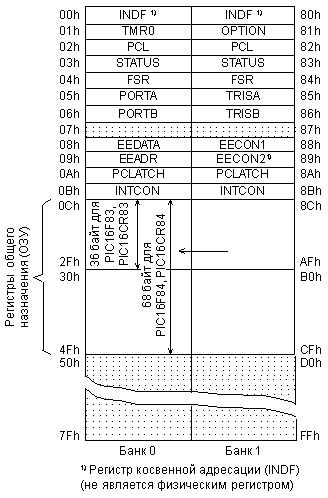
**

*Рис. 2.4. - Організація пам’яті програм*

***Пам’ять даних***

*Память даних МК розбита на дві області. Перші 12 адрес –це область регістрів спеціальних функцій (SFR), а друга – область регістрів загального користування (GPR). Область SFR виконує управління работою пристроя.*

*Обидві області, в свою чергу, розбиті на банки 0 и 1. Банк 0 вибирається обнуленням біту RP0   регістра статусу ( STATUS ). Встановлення біту RP0 в одиницю перемикає на банк 1. Коден банк має об’єм 128 байтів. Однако для PIC16F84 и PIC16CR84 пам’ть даних існує тілько до адреси 04Fh.*

**

*Рис. 2.5.- Організація пам’яті даних.*

*Деякі регістри спецільного призначення продубльовані в обох банках, а деякі знаходяться в банку 1 окремо.*

*Регістри з адресами 0Ch - 4Fh можуть використовуватись як регістри загального користування. Адреси регістрів загального користування банку 1 відображаються на банк 0. Отже, коли встановлений банк 1, то звернення до адрес  8Ch - CFh фактично адресує до банку 0.*

*В регістрі статусу окрім біту RP0 є ще біт RP1, що дозволяє звертатися до чотирьох сторінок (банків) майбутніх модификаций цтого кристалу.*

*До комірок ОЗУ можливо адресуватися прямо, використовуючи абсолютну адресу кажного регістру, або опосередковано, через регістр вказівник FSR. Опосередкована адресація використовує поточне значення розрядів RP1:RP0 для доступу к банкам. Це відноситься і до  EEPROM памяті даних. В обох випадках можна адресувати до 512 регістрів.*

## *Система команд*

*Формат команди*

*Кожна команда контролері серії представляє собою 14-розрядне слово, що містить поле коду операції OPCODE і поле операндів. Система команд містить команди работи з байтами, команди работи з бітами, команди управлення і операції з константами.*

*Для****команд работи з байтами****"f" визначає регістр з яким виконується операція, а біт "d" визначає регістр назначення. При "d"=0 результат розміщюється в регістр W, при "d"=1 - в регістр "f", заданий в команді.*

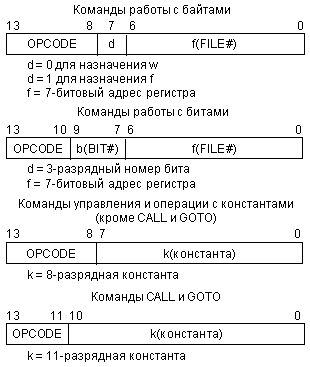
*Для****команд работи з бітами****"b"визначає номер біту, що бере участь в команді, а "f" - регістр, в якому біт знаходиться.*

*Для****команд управлення і операцій з константами****"k" визначає 8- чи 11-бітову константу або ідентифікатор.*

*Усі команди виконуються одним командним циклом, окрім двох випадків:*

* *Перехід після перевірки умови , якщо результат перевірки - істина.*
* *Зміна лічильника команд, як результат виконаної команди*

*В цих випадках команда виконується за два цикла з виконанням другого циклу як NOP. Один командний цикл складається з чотирьох періодів генератора. Таким чином, для генератора з частотою 4 МГц час виконання команди становить 1 мкс.*

**

*Рис. 2.6.- Формат команд*

*Табл. 2.1. - Визначення символів*

|  |  |
| --- | --- |
| ***f****:* | *Адреса регістру* |
| ***W****:* | *Робочий регістр* |
| ***b****:* | *Номер біту в 8-ми розрядному регістрі* |
| ***k****:* | *Константа* |
| ***x****:* | *Не використовується. Асемблер формує код з х=0* |
| ***d****:* | *Регістр призначення: d=0 - результат в регістрі W d=1 - результат в регістрі f. За замовчуванням d=1* |
| ***label****:* | *Ім’я мітки* |
| ***TOS****:* | *Вершина стеку* |
| ***РС****:* | *Лічильник команд* |
| ***ТО****:* | *Тайм-аут* |
| ***PD****:* | *Вимкнення живлення* |
| ***dest****:* | *Регістр призначення: робочий регістр W або регістр, заданий в команді* |
| ***[]****:* | *Необов’язкові параметри* |
| ***()****:* | *Вміст* |

* 1. *Система команд*

*Табл. 2.2. - Система команд*

|  |  |  |  |
| --- | --- | --- | --- |
| *Мнемоніка* | *Опис команди* | *Цикли* | *Біти стану* |
| *ADDWF f, d* | *Додавання W і f* | *1* | *C, DC, Z* |
| *ANDWF f, d* | *Логічне И W і f* | *1* | *Z* |
| *CLRF f* | *Скидання регістру f* | *1* | *Z* |
| *CLRW* | *Скидання регістру W* | *1* | *Z* |
| *COMF f, d* | *Інверсія регістру f* | *1* | *Z* |
| *DECF f, d* | *Декремент регістру f* | *1* | *Z* |
| *DECFSZ f, d* | *Декремент f, пропустити команду, якщо 0* | *1(2)* |  |
| *INCF f, d* | *Інкремент регістру f* | *1* | *Z* |
| *INCFSZ f, d* | *Інкремент f, пропустити команду, якщо 0* | *1(2)* |  |
| *IORWF f, d* | *Логічне АБО W з f* | *1* | *Z* |
| *MOVF f, d* | *Перенесення регістру f* | *1* | *Z* |
| *MOVWF f* | *Перенесення W в f* | *1* |  |
| *NOP -* | *Холоста команда* | *1* |  |
| *RLF f, d* | *Зсув f вліво через перенос* | *1* | *C* |
| *RRF f, d* | *Зсув f вправо через перенос* | *1* | *C* |
| *SUBWF f, d* | *Віднімання W з f* | *1* | *C, DC, Z* |
| *SWAPF f, d* | *Обмін місцями тетрад в f* | *1* |  |
| *XORWF f, d* | *Виключне АБО W і f* | *1* | *Z* |
| *BCF f, b* | *Скидання біта в регістрі f* | *1* |  |
| *BSF f, b* | *Встановлення біта в регістрі f* | *1* |  |
| *BTFSC f, b* | *Пропустити команду, якщо біт в f дорівнює 0* | *1(2)* |  |
| *BTFSS f, b* | *Пропустити команду, якщо біт в f дорівнює 1* | *1(2)* |  |
| *ADDLW k* | *Додання константи і W* | *1* | *C, DC, Z* |
| *ANDLW k* | *Логічне І константи і W* | *1* | *Z* |
| *CALL k* | *Виклик підпрограми* | *2* |  |
| *CLRWDT -* | *Скидання* *таймера**WDT* | *1* | */TO, /P* |
| *GOTO k* | *Перехід за адресоу* | *2* |  |
| *IORLW k* | *Логічне АБО константи і W* | *1* | *Z* |
| *MOVLW k* | *Перенесення константи в W* | *1* |  |
| *RETFIE -* | *Повернення з переривання* | *2* |  |
| *RETLW k* | *Повернення з підпрогрпми з завантаженням константи в W* | *2* |  |
| *RETURN -* | *Повернення з підпрограим* | *2* |  |
| *SLEEP -* | *Перехід в режим**SLEEP* | *1* | */TO, /P* |
| *SUBLW k* | *Віднімання W з константи* | *1* | *C, DC, Z* |
| *XORLW k* | *Виключне АБО константи і W* | *1* | *Z* |

* 1. *Карта розподілу пам’яті*

*Табл. 2.3 – Таблиця адрес*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Сторінка 1* | | *Сторінка 2* | | *Сторінка 3* | |
| *3Fh* | *ЗП63 (РД)* | *3Fh* | *ЗП127 (РД)* | *3Fh* | *…* |
| *3Eh* | *ЗП63 (РС)* | *3Eh* | *ЗП127 (РС)* | *3Eh* | *…* |
| *…* | *…* | *…* | *…* | *…* | *…* |
| *2Dh* | *ЗП1 (РД)* | *…* | *…* | *…* | *…* |
| *2Ch* | *ЗП1 (РС)* | *…* | *…* | *03h* | *ППА(РD)* |
| *…* | *…* | *…* | *…* | *02h* | *ППА (РС)* |
| *01h* | *ЗП2 (РД)* | *01h* | *ЗП64 (РД)* | *01h* | *ППА (РВ)* |
| *00h* | *ЗП2 (РС)* | *00h* | *ЗП64 (РС)* | *00h* | *ППА (РА)* |

*Сторінки пам’яті даних мають обсяг 128 байтів, тому конжа сторінка може вмістити 64 зовнішніх пристрої. На сторінці 3 знаходиться 4 адреси, що працюють с ППА.*

* 1. *Селектори адрес для зовнішніх пристроїв та ЗПД*
     1. *Зовнішній пристрій 1 має адреси 2Ch, 2Dh на сторінці 1.*

**

*Рис. 2.7. – Селектор адреси 1- го зовнішнього пристрою*

*Біти з 0 до 3 вибирають номер сторінки, останній біт адреси подається на зовнішній пристрій для визначення регістру, що працює ( РС чи РД).*

*Сторінки пам’яті обираються дешифратором. Маючи 4 біти для номерів сторінок, отримаємо 16 сторінок пам’яті.*

**

*Рис. 2.8. – Дешифратор сторінок зовнішньої пам’яті даних.*

* + 1. *Селектор для ППА*

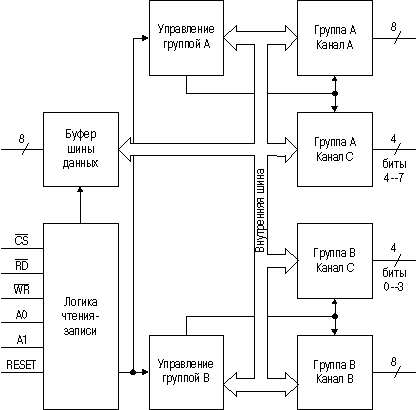
*ППА має чотири робочі регістри. Для них відведені 4 адреси на 3 сторінці пам’яті. 2 біти подаються на сам ППА, для вибору регістру, що працює в цей момент.*

**

*Рис. 2.9. – Селектор адреси для ППА*

* 1. *Програмований переферійний адаптер КР580ВВ55.*

*ППА складається з трьох каналів А, В, С по 8 бітів кожний. Також існує три режими роботи. У режимі 0 три канали можуть працювати незалежно один від одного. Режим 1 дозводє передавати інформаціюпо каналах А та В, використовуючи біти каналу С для управління. В режимі 2 канал А стає двонаправленою шиною. Структурна схема наведена на рисунку*

**

*Рис.2.10 . – Структурна схема ППА.*

* + 1. *Опис режимів роботи*
       1. *Режим 0 (просте введення / виведення)*

*Робота в цьому режимі дозволяє організувати просте введення або виведення для кожного з трьох каналів. Дані просто записуються або зчитуються з обраного каналу. Таким чином, основні особливості функціонування мікросхеми в режимі 0 наступні:*

* *два 8-бітових каналу (канали A і B) і два 4-бітових каналу (старша і молодша половинки каналу C);*
* *кожен канал може бути незалежно від інших запрограмований на введення або виведення; таким чином, всього може бути 16 різних комбінацій введення / виведення;*
* *виходи мають пам'ять, входи працюють без буферної пам'яті.*
  + - 1. *Режим 1 (періодичного введення / виведення)*

*Цей режим служить для односпрямованого обміну даними по обраному каналу з периферійним пристроєм. Для управління передачею або прийомом по каналу A використовується молодша тетрада каналу C, канал B управляється старшої тетрадой каналу C.*

*Основні особливості роботи в даному режимі:*

* *два канали (канали A і B); кожен канал може бути запрограмований на вхід або вихід;*
* *одна четвірка каналу C використовується для управління одного каналу передачі даних.*
  + - 1. *Режим 2 (двонаправлений тактируемого введення / виведення)*

*В цьому режимі можливий обмін даними по 8-бітової двобічної шині даних. Управління обміном даними, а також напрямок передачі задається за допомогою керуючих сигналів аналогічним режиму 1 способом. Можливо також генерування сигналів переривань і заборони дозволу.*

*Основні особливості роботи в даному режимі:*

* *можливе використання тільки каналу A;*
* *управління обміном забезпечується 5 бітами каналу C;*
* *входи і виходи мають буферноїпам'яттю.*
  1. *Таймер*

*PIC16 мають 3 таймера: TMR0, TMR1, TMR2, які також можуть працювати в режимі лічильника. Таймер TMR0 є 8-ми розрядним, з програмованим предделителя, налаштовується в регістрі OPTION\_REG. Таймер TMR1 16-ти розрядний, також має програмований переддільник, налаштовується в регістрі T1CON. Таймер TMR2 8-ми розрядний, крім предделителя має програмований вихідний дільник, регістри настройки T2CON і PR2. Таймери TMR0, TMR1 можна налаштувати на зовнішній тактовий сигнал з висновків T0CKI і T1CKI відповідно (режим рахунку), або використовувати внутрішній тактовий сигнал Fosc / 4 (режим таймера), при цьому приріст (ікремент) значень таймерів виконується за кожен машинний цикл. TMR2 працює тільки від внутрішнього тактового сигналу.*

*Для організації паузи будемо використовувати переривання по переповнення таймера. Частоту тактового генератора приймемо за 4 МГц, машинний цикл відповідно 1 мкс. Нижче представлений приклад коду для організації паузи в 10 мс за допомогою таймера TMR0:*

*org 0000h ;начать выполнение программы с адреса 0000h*

*goto Start ;переход на метку Start*

*org 0004h ;начать выполнение программы с адреса 0004h*

*movwf W\_temp ;сохранение значений ключевых регистров*

*swapf STATUS,W*

*clrf STATUS*

*movwf STATUS\_temp*

*bcf INTCON,T0IE ;запрет прерываний по переполнению TMR0*

*swapf STATUS\_temp,W ;восстановление значений ключевых регистров*

*movwf STATUS*

*swapf W\_temp,F*

*swapf W\_temp,W*

*retfie ;выход из подпрограммы обработки прерывания*

*;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;*

*;Основная программа*

*Start ....................... ;здесь происходит первоначальная настройка*

*....................... ;регистров специального назначения*

*.......................*

*bsf STATUS,RP0 ;запись двоичного числа 11010101 в регистр*

*movlw b'11010101' ;OPTION\_REG, тем самым устанавливаем внутренний*

*movwf OPTION\_REG ;источник тактового сигнала для TMR0*

*bcf STATUS,RP0 ;включаем предделитель перед TMR0*

*;устанавливаем коэффициент предделителя 1:64*

*movlw .100 ;запись числа 100 в регистр таймера TMR0*

*movwf TMR0*

*bсf INTCON,T0IF ;сброс флага прерывания по переполнению TMR0*

*bsf INTCON,T0IE ;разрешение прерываний по переполнению TMR0*

*bsf INTCON,GIE ;разрешение глобальных прерываний*

*.......................*

*.......................*

*.......................*

*END ;конец всей программы*

* 1. *Програмний режим обміну*

*Способи обміну інформацією між пристроями обчислювальної системи. У цьому розділі розглядається обмін між ядром обчислювальної системи і елементами системи введення-виведення. Обмін проводиться з точки зору програми, що виконується на центральних процесорах, яка взаємодіє з блоками СВВ через порти введення виведення. Все різноманіття способів такого обміну можна розділити на кілька видів:*

*Програмно керовані:*

* *Синхронний.*
* *Асинхронний з програмною перевіркою готовності (програмний*

*Полінг, «за опитуванням»).*

* *Асинхронний з апаратної перевіркою готовності (обмін по*

*переривання).*

*У режимі прямого доступу (без участі центральних процесорів).*

*Синхронний обмін даними.*

*Синхронний обмін даними передбачає відсутність ситуації неготовністі обмінюватися. Наприклад, при читанні даних з порту передбачається, що пристрій завжди готове передати їх читаючій стороні. При записи в порт, навпаки, пристрій завжди готовий прийняти дані. При синхронному обміні їм повністю управляє програма, а елемент СВВ, з яким відбувається взаємодія, ніяк не може вплинути на хід обміну. Тобто, навіть якщо пристрій працює із затримками, то ці затримки враховує програма, яка з ним взаємодіє, але сам пристрій НЕ має ніякої можливості повідомити програмі про свою готовність або неготовність. Коротко про головне: Потенційно, синхронний обмін - найшвидший з усіх розглянутих в даному розділі. Синхронний обмін вимагає мінімум апаратного забезпечення. Основний мінус: синхронний обмін складно (або взагалі неможливо) організувати з асинхронними пристроями (тобто з пристроями, що мають*

*різний час виконання операцій і / або безліч вироблених операцій з*

*сильно розрізняються часом виконання).*

*Асинхронний обмін даними з програмною перевіркою*

*готовності*

*Асинхронний обмін з програмною перевіркою готовності передбачає*

*можливість програмно оцінити ступінь готовності елемента СВВ, до*

*яким відбувається взаємодія. Зазвичай для цих цілей служить програмно доступний (через порт) регістр стану пристрою. Перед тим, як передати дані пристрою або забрати їх з нього, програма має можливість визначити, чи готове сам пристрій до цієї операції, прочитавши значення з порту стану.*

*Простим прикладом може служити робота з контролером послідовного каналу (UART) «за опитуванням»: перед тим, як прочитати дані з порту даних контролера, необхідно перевірити, чи є ці дані результатом прийому посилки і не забиралися вони програмою раніше. Простіше кажучи, необхідно перевірити дані на достовірність. Перед тим же, як записувати дані для передачі в буфер контролера, необхідно переконатися, що в буфері є місце, тобто що запис нових даних в буфер НЕ призведе до знищення раніше поміщених й ще не передані даних. Очевидно, що такий спосіб обміну вимагає додаткових зусиль з*

*боку програми на опитування готовності. Потенційно можлива ситуація*

*виходу пристрою з ладу ( «вічна неготовність»), тому необхідно*

*відповідним чином будувати алгоритм роботи з ним, щоб програма НЕ*

*«Зависала» в нескінченному циклі, чекаючи готовності з боку пристрою.*

**

*Рис. 2.11. – Небажаний алгоритм перевірки*

*Типовий алгоритм асинхронного обміну з програмним опитуванням*

*готовності (фрагмент програми) зображений на малюнку. Очевидно, такий*

*спосіб дозволяє уникнути ситуації «зависання» програми, а також провести*

*діагностику причини (тривалої) неготовності пристрою і встановити факт*

*його виходу з ладу. Безсумнівним достоїнством асинхронного обміну з програмною перевіркою готовності є те, що програма здатна визначати ступінь*

*готовності пристрою самостійно і враховувати факти відмови (їх частоту,*

*тривалість неготовність і т.п.) в своїх подальших діях.*

**

*Рис. 2.12. – Коректний алгоритм перевірки готовності.*

*При цьому очевидними «мінусами» є:*

* *Наявність програмних зусиль щодо визначення готовності. інструкції*

*за опитуванням віднімають процесорний час, протягом якого могли б*

*виконуватися інші частини алгоритму. Якби процедура опитування*

*виконувалася паралельно з основним алгоритмом (скажімо, апаратно),*

*то це б дозволило суттєво прискорити роботу всієї програми.*

* *Неефективність при обміні з великою кількістю пристроїв і / або*

*високій частоті процедури обміну. Через те, що опитування готовності*

*здійснюється програмою, накладні витрати на опитування при великому*

*кількості звернень стають дуже високими і знижують*

*продуктивність всієї системи.*

*На малюнку наведено типовий варіант програмного Полінга (опитування*

*прапорів переривання ПУ). Першим перевіряється прапорець готовності ПУ1 з*

*найбільшим пріоритетом. Якщо воно не просило обслуговування,*

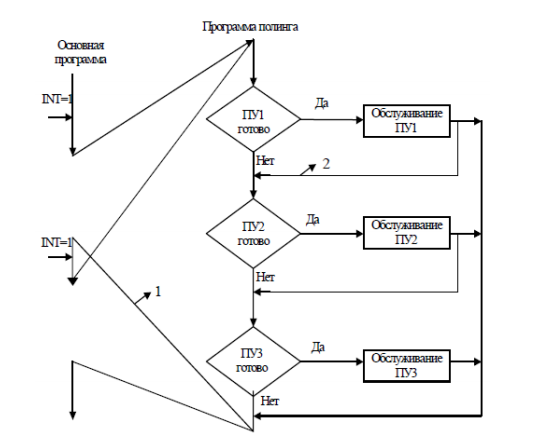
*опитується наступне ПУ і т.д. Коли зустрічається перший пристрій готове до операцій введення-виведення (ВВ), управління передається підпрограмі обслуговування цього пристрою. При завершенні обслуговування в Полінг може бути запрограмовано одне з наступних дій: Управління повертається в основну програму без перевірки готовності інших ПУ. Тут гарантується обов'язкова перевірка в кожному циклі Полінга ПУ з високим пріоритетом, так як*

*обслуговування їх блокує обслуговування пристроїв з меншим*

*пріоритетом. Управління повертається до програми Полінга, тобто в точку перевірки переривання наступного ПУ (на малюнку нижче це показано цифрою).*

*Цей спосіб гарантує перевірку в кожному циклі Полінга всіх*

*пристроїв.*

**

*Рис.2.13. – Програмний полінг.*

*Приклад:*

*ЗП1 – F0h, F1h*

*ЗП2- F2h, F3h*

*MOV R7,#10*

*LL2: MOV R0,#F0H*

*MOVX A, @R0*

*JB7 A, LL1*

*JMP LL2*

*LL1: MOV R1, #F2H*

*MOVX A, @R1*

*JB7 LL3*

*JMP LL1*

*LL3:*

*MOV R0, #F1H*

*MOVX A, @RO*

*MOV R1, #F3H*

*MOVX @R1, A*

*DJNZ R7, LL3*

*END*

## *КПП. Реалізація переривань.*

*Під перериваннями розуміють тимчасове припинення виконан­ня програми і перехід на підпрограму з можливістю повернення на перервану програму.*

*Розрізняють внутрішні і зовнішні переривання. Внутрішні пе­реривання у свою чергу поділяються на програмні і апаратні. Про­грамні переривання є ефективним засобам для виклику стандартних підпрограм базової системи вводу-виводу, дозволяють спростити процес налагодження програм взаємодії процесора із зовнішніми пристроями, які, можливо, розробляються паралельно з програмним забезпеченням і таке інше. Внутрішні апаратні переривання дозволяють процесору відреагувати на непередбачені ситуації (наприклад, переповнювання розрядної сітки, спрацьовування внутрішнього таймера, збій при зверненні до магістралі) і скоректувати обчислю­вальний процес, змінити режим роботи або, принаймні, провести безпечне зупинення системи.*

*Існують векторні і безвекторні зовнішні переривання. Запити на безвекторні переривання поступають на спеціальні входи процесо­рів. Ці запити мають більший пріоритет, ніж запити на векторні пе­реривання. Механізм обробки безвекторних переривань закладений в процесорі на мікропрограмному або апратному рівні.*

*У цілому, механізм забезпечує перехід до підпрограми за визначеної адреси (із запам'ятовуванням адреси повернення і стану перерваної програми). Безвекторні переривання, як правило, використовуються процесо­ром на його локальній магістралі, причому, входи запитів мають ви­значене призначення (відключення живлення, спрацьовування зов­нішнього таймера і таке інше).*

*Зовнішні векторні переривання є важливим засобом синхроні­зації процесів в мікропроцесорній системі. За допомогою сигналів зовнішніх переривань пристрої системи оповіщають один одного про готовність до передачі даних або про змінення режимів роботи, тобто обмінюються управляючою інформацією. Якщо для прийому сигналів переривань в процесорах передбачені спеціальні входи, то для формування сигналів вимоги переривання можуть знадобитися додаткові засоби.*

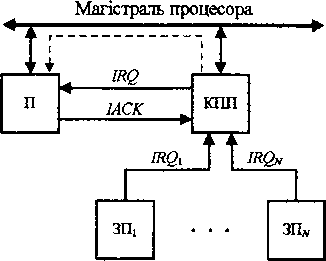
*Зовнішні векторні переривання реалізуються за допомогою спе­ціального контролера наступним чином.*

*По запитах від зовнішніх пристроїв контролер переривань із урахуванням системи пріоритетів видає на процесор сигнал вимоги переривання IRQ*. *Умовою формування такого сигналу є готовність пристроїв до взаємодії з процесором і відсутність маскування запи­тів з боку процесора.*

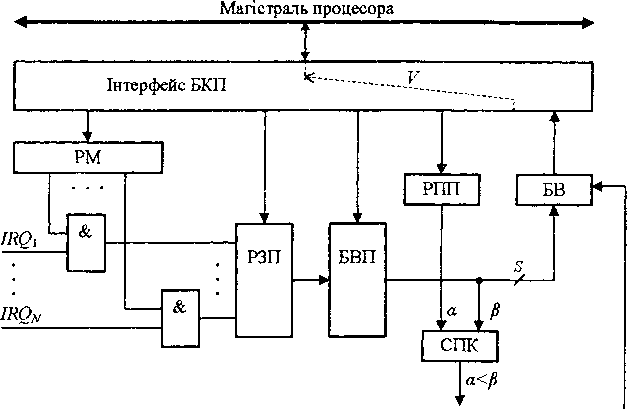
*Отримавши сигнал IRQ*, *процесор завершує до кінця виконання чергової команди, видає на контролер сигнал підтвердження пере­ривання* ІАСК *і зчитує з шини даних вектор, який виставляє на шину контролер переривання.*

*Процесор зберігає (зазвичай в стеку) адреси повернення і стан програми, що тимчасово перервана, обчислює адресу переходу, ви­користовуючи для цього отриманий вектор переривання й після цьо­го здійснює перехід на першу команду підпрограми обробки пере­ривання. Процесор виконує підпрограму. Остання команда підпрог­рами є спеціальною командою повернення з переривання, яка відно­влює стан перерваної основної програми і передає ш управління.*

*Схема підключення централізованого контролера пріоритетних переривань (КПП) показана на рис. 2.14. Зовнішні пристрої (ЗП), в числі яких можуть бути і інші процесори, формують запити на пере­ривання IRQ*. *Особливість такої схеми полягає в тому, що підклю­чення ЗП до магістралі процесора не є обов’язковим. Це пояснюється тим, що вектор* V *в процесор передає КПП, а не зовнішні пристрої. Завдяки цьому через переривання можуть взаємодіяти процесори, підключені до різних магістралей.*

**

*Рис. 2.14. - Система с централізованим контролером пріоритетних переривань*

**

*Рис. 2.15. - Централізований КПП*

*Можливий варіант побудови централізованого КПП показаний на рис. 2.15. Контролер підключений до магістралі процесора через інтерфейс, який забезпечує процесору доступ до регістру маски (РМ) і регістру поточного пріоритету (РПП). Адреси вказаних регіс­трів включені в адресний простір процесора. Записом слова маски в регістр РМ процесор може дозволити або заборонити переривання від певних джерел. Незамасковані запити* IRQ, *записуються в регістр запитів переривань (РЗП). Блок вибору пріоритету (БВП) формує код самого старшого пріоритету, який в схемі порівняння кодів (СПК) порівнюється з поточним пріоритетом - пріоритетом викону­ваної процесором програми. Якщо запрошений пріоритет вище по­точного, то формується сигнал* IRQ. *Далі, у відповідь сигналу* ІАСК, *через буфер вектора (БВ) і інтерфейс в магістраль видається вектор, який приймається процесором. У регістр РПП записується новий код пріоритету. У даному контролері роль вектора виконує код номера запиту.*

*До достоїнств централізованих контролерів слід віднести на­ступне:*

* *можливість динамічно змінювати стратегію обслуговування заявок;*
* *швидке вибіркове маскування запитів на переривання.*

*Централізований КПП потенційно дозволяє забезпечувати різні дисципліни обслуговування заявок, оскільки всі заявки поступають в один пристрій. Для забезпечення пріоритетного обслуговування застосовується блок вибору пріоритету. У простому випадку в якості БВП використовується пріоритетний шифратор (ПШ), який формує код старшого рівня пріоритету. При цьому всі запити мають фіксо­вані пріоритети, рівень яких визначається номером входу КПП. За рахунок ускладнення схеми можна забезпечити інші дисципліни об­слуговування заявок. Найчастіше застосовують циклічну зміну рів­нів пріоритетів, що забезпечує гарантоване обслуговування будь- якого запиту на певному проміжку часі, причому, незалежно від ін­тенсивності запитів.*

*Оскільки регістр маски включений в адресний простір процесо­ра, то за умови, що число запитів не перевищує розрядності шини даних (що зазвичай виконується), процесор може за одне звернення до цього регістра записати в нього будь-яку маску.*

*До недоліків централізованих КПП слід віднести:*

* *велику кількість ліній запитів в шині управління (що дорів­нює кількості ЗП);*
* *обмеження на максимальне число джерел переривань;*
* *можлива неоднорідність процесорних модулів.*

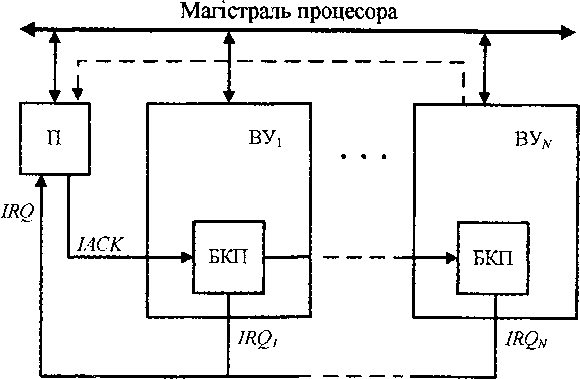
*Останній з вказаних недоліків обумовлений тим, що в централі­зованих системах приймачем переривань зазвичай є управляючий процесор. Отже, за необхідності використання в системі резерву апа­ратури на рівні модулів буде потрібно різні типи резерву.*

*Вказані недоліки обмежують можливості застосування цього способу при побудові однорідної модульної МПС.*

*Під час використання розподіленого контролера переривань (рис. 2.15) все ЗП повинні бути підключені до магістралі процесора. Це обумовлено тим, що вектор переривання на шину даних в даному випадку видає сам активний ЗП.*

*До складу кожного ЗП включений блок контролера переривань БКП, який видає сигнал запиту* IRQ, *на загальну лінію* IRQ. *Техно­логічні особливості елементної бази повинні допускати таке об'єд­нання виходів елементів (наприклад, використовуються елементи з відкритим колектором, а сигнали запитів мають активний низький рівень). У відповідь сигнал процесора* ІАСК *розповсюджується по­слідовно через елементи БКП, створюючи так званий пріоритетний ланцюжок або "гірлянду"* (daisy chain). *Елементи ланцюжка в кож- ному БКП пропускають сигнал* ІАСК *або розривають ланцюжок. Пріоритетний ланцюжок розривається на першому (по шляху розпо­всюдження сигналу) активному ЗП, який виставив сигнал запиту IRQ*. *Даний активний ЗП видає на шину даних вектор переривання, який приймається процесором.*

*Приклад побудови БКП показаний на рис. 2.16.*

**

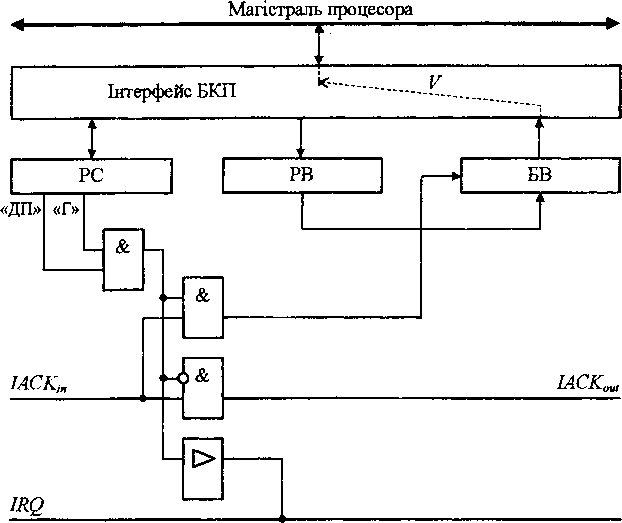
*Рис. 2.16. – Система с децентралізованим КПП*

*Блок БКП містить доступні для процесора регістр стану (РС) і регістр вектора (РВ). Під час ініціалізації режиму роботи системи процесор записує в регістр РВ вектор переривання, а в регістр РС - біт дозволу переривання (ДП). Якщо ЗП готовий до взаємодії з про­цесором, то в регістрі РС встановлюється біт готовності «Г». Це встановлення виконується засобами внутрішнього управління ЗП. За збігу сигналів «Г» і «ДП» формується запит IRQ*, *який через елемент узгодження поступає на лінію Вхідний для кожного блоку сигнал* ІАСКіп *передається на вихід* ІАСКоut *або забезпечує видачу век­тора через буфер вектора (БВ), що визначається значенням IRQ*.

*До достоїнств розподілених КПП можна віднести:*

* *невелику кількість ліній зв'язку в шині управління;*
* *простоту нарощування числа ЗП.*

*Завдяки вказаним достоїнствам розподілені контролери нахо­дять широке застосування в МПС системах.*

**

*Рис. 2.17. – Блок контролера переривань розподіленого КПП.*

*Недоліками розподілених КПП є:*

* *велика кількість звернень процесора до магістралі під час іні­ціалізації системи;*
* *використання фіксованих рівнів пріоритетів запитів, які не можна змінювати динамічно.*

*Перший недолік обумовлений необхідністю запису біта дозволу переривання в кожен блок окремо. Коли ініціалізація режимів здійс­нюється рідко, цей недолік можна вважати неістотним.*

*Використання фіксованих рівнів пріоритетів не забезпечує га­рантованого обслуговування заявок на певному відрізку часу. Заявки з низьким рівнем пріоритету за великої інтенсивності заявок з висо­кими пріоритетами можуть не виконуватися тривалий час. Ця обста­вина може привести до уповільнення обчислювального процесу, а іноді - до тупикової ситуації.*

*Обидва розглянуті підходи до реалізації зовнішніх векторних переривань можуть поєднуватися. Наприклад, декілька пристроїв об’єднуються ланцюжком «гірлянди», а декілька ланцюжків, у свою чергу, замикаються на централізованому контролері.*

* 1. *КПДП*

*Режим прямого доступа до спільної пам’яті (СП) необхідний для того, аби розвантажити процесор в режимі обміну даними з СП та для забезпечення збільшення швидкості обміну даними між пристроями системи. Процесор відключається або вирішує свої завдання, які не вимагають даних зі спільної пам’яті. Для реалізації режиму прямого доступу до пам’яті використовується контролер прямого доступу до пам’яті (КПДП).*

*КПДП може бути як і процесор – активним пристроєм, оскільки може звертатися до зовнішніх пристроїв (ЗП) та спільної пам’яті, а також в подальшому керувати режимом обміну, захоплюючи системну магістраль. Обидва КПДП та процесор (П) по черзі захоплюють системну магістраль, завдяки чому здійснюється паралельна робота цих пристроїв. КПДП має декілька адрес в загальному адресному просторі для таких регістрів:*

*СТ – лічильник кількості слів, які передаються у масив;*

*РПА – регістр початкової адреси (ЗП)ж*

*РК – регістр команд;*

*РР – регістр режиму (передача слова чи масиву).*

*Ініціалізація КПДП відбувається у програмному режимі, за рахунок передачі у відповідні регістри контролера ПДП інформації, необхідної для керування обміном (адреса комірки пам’яті, в якій знаходиться перший блок даних, що записуються або зчитуються; адреса порту; загальна кількість даних, що передаються; напрям передачі тощо). Після закінчення обміну центральний процесор (ЦП) отримує сигнал від контролера і переходить до виконання основної програми.*

*Контролер прямого доступу до пам’яті працює в режимі одиничної або пакетної(блокової) передачі. Кожен зовнішній пристрій містить власний інтерфейс, через який відбувається передача даних.*

*Під час блокової передачі даних відбувається захоплення системної магістралі(СМ) на весь час передачі масиву. Даний тип передачі забезпечує передачу масиву швидше, ніж процесор, тому що контролер є пристроєм, що не потребує зчитування команд для передачі даних та керування на апаратному рівні.*

*В режимі одиничної передачі захоплення системної магістралі відбувається на один цикл звернення до СП та ЗП. В режимі блокової передачі процесор не зайнятий протягом часу, необхідного для передачі масиву. В режимі одиничної передачі процесор та КПДП працюють паралельно – процесор виконує свою програму (звернення до СП за командами та даними), а КПДП захоплює цикли для передачі одного слова.*

*КПДП використовується у складі МПС для реалізації прямого доступу до СП. Для побудови піраміди КПДП використовуються мікросхеми КПДП (8257A – Intel(DMA Controller), аналогом якої є К1810ВТ37). Кожна мікросхема обслуговує чотири зовнішніх пристрої. КПДП дозволяє реалізувати передачу пам’ять-пам’ять та має широкі можливості програмного керування та каскадування.*

**

*Рис. 2.18. – Структура КПДП (8257А - Intel).*

*Кожен канал містить у собі по чотири регістри:*

*CAR – регістр поточної адреси;*

*BAR – WCR – регістр базової адреси та числа циклів;*

*CWR – регістр циклів (декрементуючий лічильник; якщо ТС = 1, то відбувається завершення циклів);*

*MR – регістр режиму.*

*Регістри відносяться до спільного адресного простору ОЗП та ЗП.*

*Регістр поточної адреси* ***CAR*** *зберігає поточну адресу комірки пам’яті при виконанні циклу ПДП. Після виконання циклу ПДП вміст цього регістра збільшується або зменшується на одиницю. Вміст регістра може буде прочитаним або завантаженим за допомогою двох команд введення-виведення.*

*Регістр зберігання базової адреси* ***BAR*** *та регістр зберігання базового числа циклів ПДП* ***WCR*** *зберігають базові значення адреси та числа циклів ПДП, приймають участь у автоініціалізацію. Під час початкового завантаження контролера ПДП вихідними параметрами одночасно відбувається запис до регістрів CAR, BAR, CWR та WCR. В процесі виконання циклів ПДП вміст BAR і WCR не змінюється. Прочитати стан цих регістрів неможливо.*

*Регистр режиму* ***MR*** *визначає режим роботи (в молодших розрядах Dl, DO вказується код номера каналу). За допомогою інших розрядів задається один з типів передачі – читання, запис, перевірка; визначається режим автозавантаження та роботи каналу – передача по запиту, одинична передача, блокова передача, контролер в режимі каскадування, а також визначається режим зміни регістра CAR (на зменшення чи збільшення).*

*Регістр циклів ПДП* ***CWR*** *зберігає кількість слів, призначених для передачі. Константа, що завантажується, має бути більшою на одиницю від кількості слів, необхідних для передачі. Читання та запис вмісту регістра здійснюється двома послідовно виконуваними командами введення-виведення. Вміст CWR може бути оновлений під час автоініціалізації за сигналом ЕОР, інакше в регістрі зберігається значення FFFFH.*

*КПДП складається з трьох функціональних блоків:*

* ***БШД******(буфер шини даних)*** *– необхідний для узгодження роботи з ЦП;*
* ***БКК*** *(****блок керування контролером)*** *– містить один регістр – TR (регістр тимчасового зберігання даних), що забезпечує зберігання байта в циклі передачі пам’ять-пам'ять на час зміни адреси;*
* ***БКРР (блок керування режимом роботи)*** *– виробляє необхідні сигнали для керування під час передачі даних в циклах ПДП. Містить в собі чотири регістри: CR, RR, SR, MASK;*
  + ***CR*** *– регістр команд – визначає основні параметри роботи каналу, задає режим роботи;*
  + ***SR*** *– регістр умови, розряди якого встановлюються апаратно після закінчення циклів ПДП чи за зовнішнім сигналом ЕОР;*
  + ***RR*** *– регістр запитів, кожен розряд якого відповідає одному з каналів;*
  + ***MASK*** *– маскує сигнали DREQ кожного каналу, розряди якого можуть бути встановлені одночасно або роздільно за спеціальною командою.*

***Частина 2***

1. ***Спеціалізований обчислювач***
   1. *Аналіз рішення аглоритму заданої задачі*

*Обчислення максимального по модулю власного значення і відповідного до нього власного вектора симетричної матриці метожом скалярного добутку. Задана матриця A = [aij] , aij = aji*

*Максимальне власне значення матриці є Л = (yk,yk) / (yk-1,yk).*

*Yk=A\*yk*

*Yk\* = yk/sqrt(yk,yk)*

*Керуючими умовами є Б та ABS (M-F).*

*СО має такі вхідні параметри вектор А, що містить нижню частину матриці під головною діагоналлю, значення точності E, розмірність матриці N.*

*Вихідними даними є власне значення Л та власний вектор У.*

* 1. *Розробка аглоритму задачі*



*Рис 3.1 – Аглоритм задачі для СО*

* 1. *Система команд*

|  |  |
| --- | --- |
| *Mov* |  |
| *add* |  |
| *Inc* |  |
| *Sqrt* |  |
| *Div* |  |
| *mul* |  |
| *abs* |  |
| *output* |  |
| *loop* |  |

1. ***Опис функціональної та принципової схеми заданого блоку***
2. ***Розробка вузла ЕОМ на ПЛІС***

*Z = 1/X + 4\*Y;*

*Спосіб ділення 1;*

*Розрядність операндів 6 біт;*

*Початкова адреса мікрокоманди 7h.*

*Адресація відносна*

*Опис арифметичного-логічного пристрою*

*При реалізації ділення за першим варіантом здійснюється зсув вліво залишку при нерухомому дільнику. На рис. 3.1 показана можлива побудова пристрою ділення. Чергова остача формується в регістрі RG2 (у вихідному стані в цьому регістрі записаний Х). Виходи RG2 підключені до входів суматора SM безпосередньо, тобто ланцюги видачі коду з RG2 не потрібні. Дільник Y знаходиться в регістрі RG1. Результат формується в регістрі RG3 за (n + 1) циклів. Знак остачі визначається розрядом RG2[n+2]. Розряд RG3[n+1] використовується для визначення кінця операції, ознакою цього є маркерний нуль на виході розряду. Максимальний час одержання цифри результату визначається виразом tЦ = tД+ tЗ, де tД – тривалість виконання мікрооперації додавання/віднімання; tЗ – тривалість виконання мікрооперації зсуву. Час для одержання n+1 цифри частки визначається виразом t=(n+1) tЦ.*



*Рис. 3.2. Операційна схема ділення за першим способом із зсувом остачі*

*Змістовний мікроалгоритм*



*Рис 3.3.- Змістовний мікроалгоритм*

*Функціональна схема арифметично-логічного пристрою.*



*Рис 3.4 – Функціональна схема АЛУ СО.*

*Закодований мікроалгоритм*

|  |  |
| --- | --- |
| *Мікрооперації* | *Управляючі сигнали* |
| *W1* | *y1* |
| *W* | *y2* |
| *SL2, SL1* | *y3* |
| *V1* | *y4* |
| *V2* | *y5* |
| *V3* | *y6* |
| *V4* | *y7* |

*Таблиця 3.1. Карта програмування БМУ*

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***№*** | ***АДРЕСА*** | ***S*** | ***M*** | ***Y1*** | ***Y2*** | ***Y3*** | ***Y4*** | ***Y5*** | ***Y6*** | ***Y7*** | ***B3*** | ***B4*** |
| *П(1)* | *000 111* | *000 001* | *00* | ***0*** | ***0*** | *1* | *0* | *1* | *1* | *X1 AND Y6* | *000* | *0* |
| *2* | *001 000* | *000 110* | *00* | ***1*** | ***1*** | *1* | *0* | *1* | *1* | *X1 AND Y6* | *000* | *1* |
| *3* | *001 110* | *011 111* | *00* | *1* | ***0*** | *1* | *0* | *0* | *1* | *X1 AND Y6* | *000* | *0* |
| *4* | *101 101* | *011 111* | *00* | *1* | ***1*** | ***0*** | *0* | *0* | *1* | *X1 AND Y6* | *101* | *1* |
| *5* | *001 100* | *000 010* | *10(Х2)* | *1* | *1* | ***1*** | ***0*** | *0* | ***1*** | *X1 AND Y6* | *000* | *0* |
| *6* | *001 111* | *000 001* | *00* | *1* | ***0*** | *1* | ***1*** | *0* | ***0*** | *X1 AND Y6* | *000* | *1* |
| *7* | *010 000* | *000 001* | *00* | *1* | ***1*** | *1* | *1* | *0* | *0* | *X1 AND Y6* | *101* | *1* |
| *8* | *010 001* | *000 001* | *00* | *1* | ***0*** | *1* | *0* | *0* | *0* | *X1 AND Y6* | *000* | *0* |
| *9* | *010 010* | *000 001* | *00* | *1* | ***1*** | *1* | *0* | *0* | *0* | *X1 AND Y6* | *101* | *1* |
| *К(10)* | *010 011* | *000000* | *00* | *0* | *0* | *0* | *0* | *0* | *0* | *X1 AND Y6* | *000* | *0* |



*Рис 3.5 – Закодований мікроалгоритм*

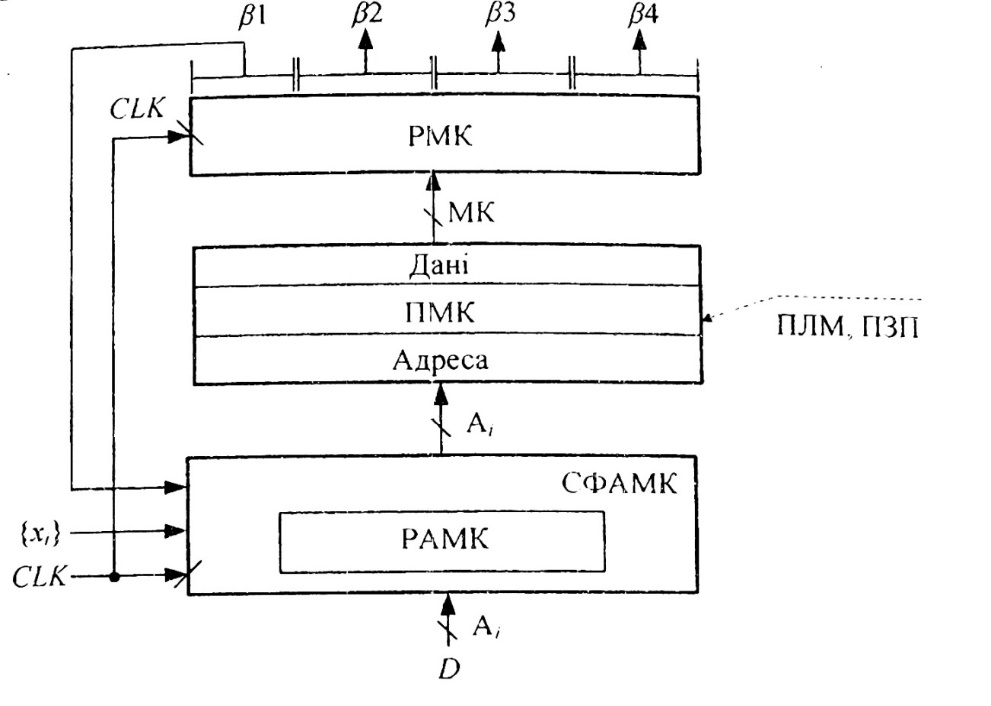
*Опис роботи БМУ*

*БМУ функціонує у відповідності з принципом мікропрограмного управління, що полягає у наступному.*

*Під час виконання мікропрограми в кожному такті із постійної пам’яті БМУ зчитується та розшифровується чергова мікрокоманда. В результаті виконання мікрокоманди формуються управляючі сигнали необхідної тривалості, що поступають на всі функціональні частини обчислювальної системи, а також формується адреса наступної мікрокоманди.*

*Можна виділити наступні етапи виконання команди в обчислювальній системі:*

1. *Вибірка команди. З ОП зчитується команда в регістр команд процесора, для чого виконується відповідна МП, що записана у пам’ять БМУ.*
2. *Розпакування команди. Команда розшифровується (аналізуються поля слова команди, визначаються операнди), що забезпечується виконанням відповідної МП.*
3. *Виконання операції. Виконується МП виконання заданої операції над визначеними операндами.*
4. *Формування адреси наступної команди. Відповідна МП формує адресу наступної команди у лічильнику команд.*

**

*Рисунок 3.6 - Структурна схема БМУ.*

*Основні функціональні частини БМУ:*

* *РАМК - регістр адреси МК;*
* *СФАМК - схема формування адреси МК;*
* *ПМК - пам'ять МК;*
* *РМК - регістр МК;*
* *Аі - адреса МК;*
* *CLK - синхросигнал;*
* *{xi} - логічні умови;*
* *D - вхід завдання початкової адреси мікропрогрми.*

*Основним призначенням СФАМК є реалізація управляючих структур, що зустрічаються у мікропрограмах: лінійна послідовність, структури виду «якщо А, то Б, інакше В» та структури виду «поки А, роби Б».*

*При цьому схема виконує наступні функції:*

* *проводить дешифрацію коду операції команди (КОП) для звернення до першої мікрокоманди мікропрограми, що інтерпретує дону команду;*
* *формує адреси наступних мікрокоманд;*
* *зберігає ознаки переходів, до виходять з операційного блоку то створюються при виконанні мікрокоманд умовного переходу;*
* *виконує управління перериваннями на мікропрограмному рівні.*

*Пам'ять мікропрограм призначена для зберігання мікрокоманд, її ємність та розрядність однозначно визначаються набором мікропрограм, що реалізуються. Шляхом зміни набору мікропрограм можна змінювати систему команд мікропроцесора і тим самим орієнтувати його функціональну спрямованість.*

*У кожному такті за синхросигналом CLK адреса мікрокоманди поновлюється у РАМК і надходить на адресний вхід ПМК. За адресою, що надійшла у ПМК, обирається відповідна мікрокоманда і видається на вихід даних ПМК. Слово мікрокоманди записується у РМК за зворотнім перепадом синхросигналу CLK.*

*Сигнали зони β2 управляють вузлами МПС, зони β3 - визначають тривалість цих сигналів, сигнали зони β1 разом із логічними умовами {xi} поступають на вхід СФАМК і формують адресу наступної МК. За черговим сигналом CLK адреса наступної МК буде сформована у РАМК. Зона β4 використовується для виконання допоміжних функцій, наприклад, контролю апаратури.*

*Розрахунок параметрів БМУ*

*Зона в1 – S + M*

*M = 2; S=6;*

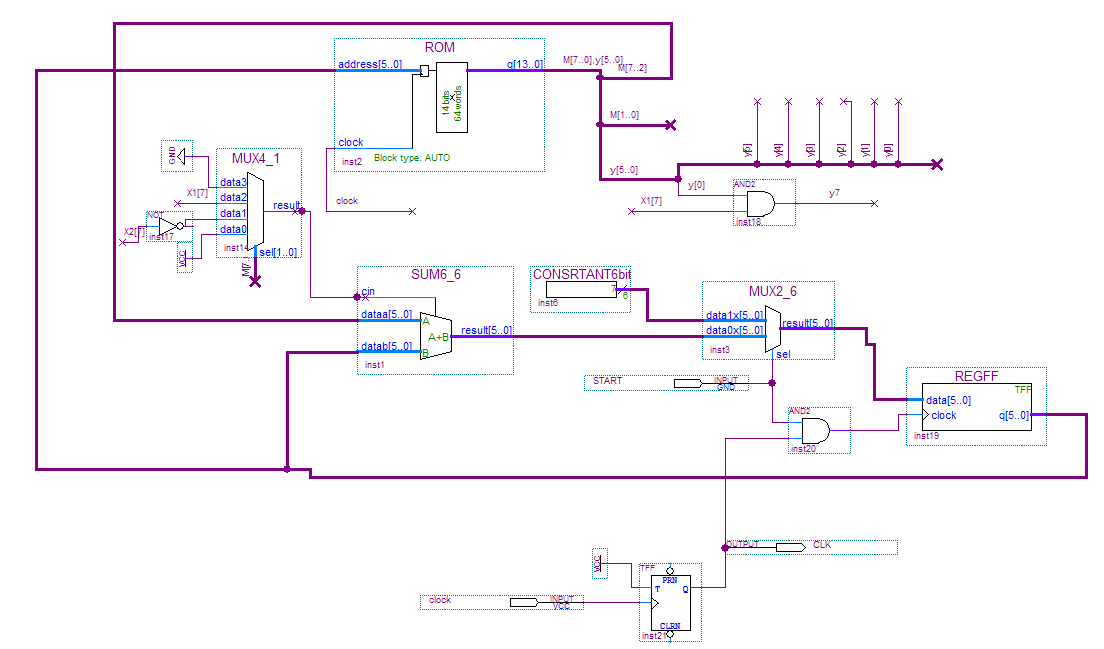
*Зона в2 – кількість керуючих сигналів 7, але 7-и1 сигнал отримується логічним виразом з 6-го. Тому до пам’яті буде занесено лише шість. Отже в2=6.*

*Загалом довжина слова = 14 біт. На рисунку 3.7 наведена функціональна схема БМУ.*



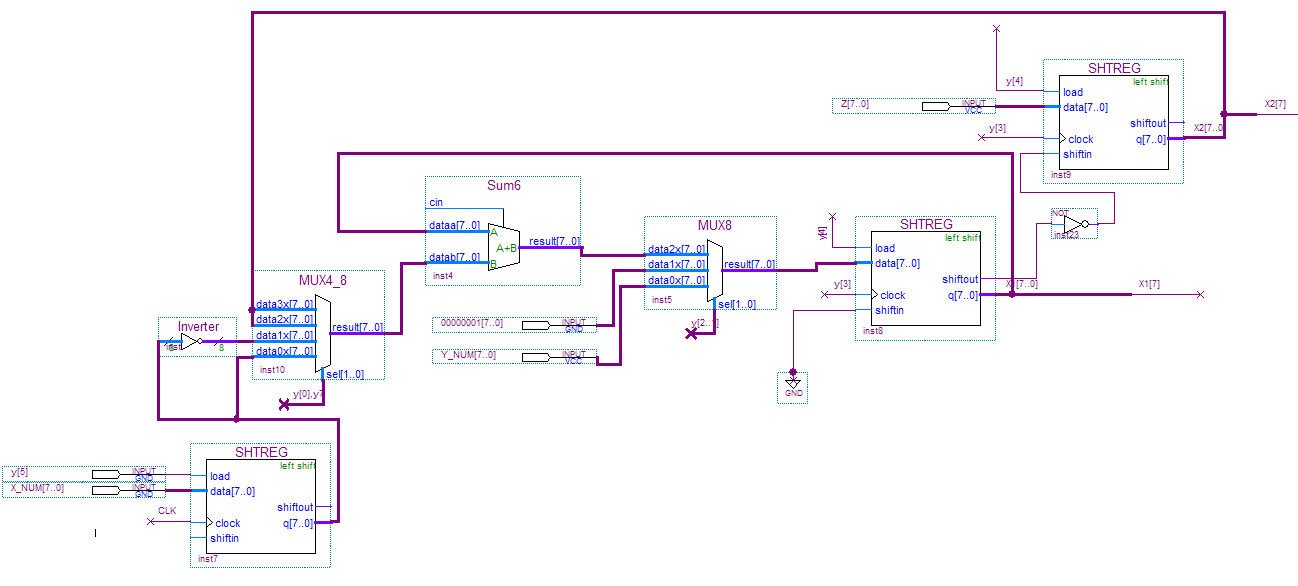
*Рис 3.7 – Функціональна схема БМУ СО*

1. ***Побудова заданої задачі на ПЛІС***

**

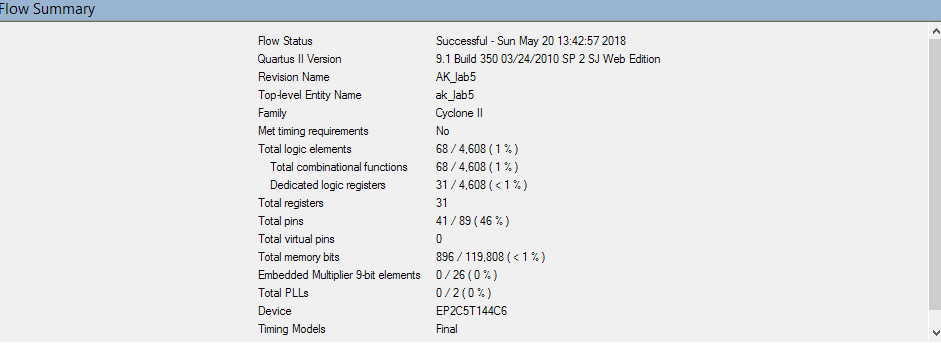
*Рис 3.9 – Схема БМУ СО*

*Особливість розробленої схеми АЛП в тому, що вона управляється лише сигналами з БМУ,не використовуючи синхросигал. Сигнали БМУ керують записом початкових значень в регістри, лічильником, зсувами та додаванням. Для занесення початкових значень в регістр використали елемент константи та мультиплексор, котрий по сигналу початкового стану записує початковий стан регістру, а при його відсутності подається значення з суматора. АЛП формує сигнали х1 та х2, котрі є управляючими сигналами для БМУ.*

**

*Рис. 3.10 – Схема АЛУ СО*

*Результат компіляції проекту наведено на рис 3.11.*



*Рис 3.11 – Результат компіляції*

***Висновки***

*Â õîä³ âèêîíàííÿ êóðñîâîãî ïðîåêòó áóëà ðîçðîáëåíà ì³êðîïðîöåñîðíà ñèñòåìà, çàñíîâàíà íà ì³êðîïðîöåñîðíîìó ÿäð³ ATmega8515. Âîíà ì³ñòèòü çîâ- í³øíþ ïàì’ÿòü äàíèõ òà ïðîãðàì, ïðèñòðî¿, ïðîãðàìîâàíèé ïåðèôåð³éíèé àäàï- òåð, äîäàòêîâ³ ïîðòè ââîäó-âèâîäó. Äëÿ îðãàí³çàö³¿ ïåðåðèâàíü òà ïðÿìîãî äîñòóïó äî ïàì’ÿò³ îðãàí³çîâàí³ öåíòðàë³çîâàí³ àðá³òðè ïð³îðèòåòíèõ ïåðåðè- âàíü ³ ïðÿìîãî äîñòóïó äî ïàì’ÿò³.*

*Òàêîæ â õîä³ ðîáîòè áóâ ðîçðîáëåíèé ñïåö³àë³çîâàíèé îá÷èñëþâà÷ íà áàç³ ïðîãðàìîâàíî¿ ëîã³÷íî¿ ³íòåãðàëüíî¿ ñõåìè, ÿêèé âèêîíóº ð³çí³ àðèôìåòè÷- í³ é ëîã³÷í³ îïåðàö³¿. Éîãî âèêîðèñòàííÿ äîçâîëÿº ï³äâèùèòè åôåêòèâí³ñòü ðîáîòè ðîçðîáëåíî¿ ñèñòåìè çà ðàõóíîê âèíåñåííÿ ãðîì³çäêèõ îá÷èñëåíü (íà- ïðèêëàä ä³ëåííÿ) â îêðåìèé ïðèñòð³é.*

*Ñõåìà îá÷èñëþâà÷à ðîçðîáëÿëàñÿ â ñèñòåì³ àâòîìàòèçîâàíîãî ïðîåêòó- âàííÿ Quartus II, ÿêà äîçâîëÿº ñòâîðþâàòè ð³çí³ ïðèñòðî¿ íà áàç³ ïðîãðàìîâà- íèõ ëîã³÷íèõ ³íòåãðàëüíèõ ñõåì. Ïåðåâàãîþ äàíî¿ ìåòîäèêè º òå, ùî, ïî- ïåðøå, ¿õ âèêîðèñòàííÿ äîçâîëÿº ñêîðîòèòè ÷àñ ðîçðîáêè ³, ïî-äðóãå, âèðîá- íèöòâî ìàëèõ ïàðò³é º åêîíîì³÷íî âèïðàâäàíî, íà â³äì³íó â³ä âèðîáíèöòâà çà- ìîâëåíèõ ³íòåãðàëüíèõ ñõåì, ùî íå ìîæóòü áóòè ïåðåïðîãðàìîâàí³.*

***Список використаної літератури***

1. *Балашов е.п., Пузанков Д.В. Микропроцессоры и микропроцессорные системы. – М.: Радио и связь, 1981. – 328 с.*
2. *Бородин В.Б., Калинин А.В. Системы на микроконтроллерах и БИС программируемой логики. – М.: Издательство ЭКОМ, 2002. – 400 с.*
3. *Брик Дж., Мик Дж. Проектирование микропроцессорных устройств с разрядно–модульной организацией: В 2-х частях. – М.: Мир, 1984. – 479 с.*
4. *Булгаков С.С. и др. Проектирование цифровых систем на комплектах микропрограммируемых БИС. – М.: Радио и связь, 1984. –   
   240 с.*
5. *Глушков В.М. и др. Программное обеспечение ЭВМ МИР-1 и МИР-2. Том 2. – К.: "Наукова думка", 1976. – 371 с.*
6. *Жабин В.И. Архитектура вычислительных систем реального времени. – К.: ВЕК+, 2003. – 176 с.*
7. *Жабін В.І., Жуков І.А.,Клименко В.В., Стиренко С.Г. Арифметичні та управляючі пристрої цифрових ЕОМ: Навчальний посібник. К.: Видавництво ВЄК +, 2008. 176 с.*
8. *Жабін В.І., Жуков І.А.,Клименко В.В., Ткаченко В.В. Мікропроцесорні системи: Навчальний посібник. – К.: Видавництво «СПД Гуральник», 2009. 492 с.*
9. *Каган Б.М. Электронные вычислительные машины и системы.– М.: Энергоатомиздат, 1985. – 552 с.*
10. *Карцев М.А. Архитектура цифровых вычислительных машин.– М.: "Наука", 1978. – 295 с.*
11. *Ларионов А.М., Майоров С.А., Новиков Г.И. Вычислительние комплексы, системы и сети. – Л.: Энергоатомиздат, 1986. – 286 с.*
12. *Микропроцесорные системы: Учебное пособие для вузов / Е.К. Александров, Р.И. Грушевицкий, М.С. Куприянов и др.; Под общ. ред. Д.В. Пузанкова. – СПб.: Политехника, 2002. – 935 с.*
13. *Молчанов А.А., Корнейчук В.И., Тарасенко В.П. Справочник по микропроцессорным устройствам. – К.: Техніка, 1987. – 288 с.*