# *Національний технічний університет України*

***«Київський політехнічний інститут»***

#### ***Факультет інформатики та обчислювальної техніки***

### ***Кафедра обчислювальної техніки***

*Розрахунково-графічна робота*

*з дисципліни «Архітектура комп’ютерів»*

|  |  |  |
| --- | --- | --- |
|  | *Виконав**Куцовол Віктор Вікторович* | |
|  | *Факультет* *ІОТ* | |
|  | *Група ІО-82  Залікова книжка № 8213* | |
|  | |  |

***Допущена до захисту\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***Номер технічного завдання*** *- 10000000010101*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(підпис керівника)*

***Київ - 2010р.***

***Опис альбому***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***№ рядка*** | ***Формат*** | ***Позначення*** | ***Найменування*** | ***Кількість*** | | ***Примітка*** |
| *1* |  |  |  |  |  | |
| *2* |  |  | *Документація загальна* |  |  | |
| *3* |  |  |  |  |  | |
| *4* |  |  | *розроблена заново* |  |  | |
| *5* |  |  |  |  |  | |
| *6* | *А4* | *ІАЛЦ.463626.002 ТЗ* | *Технічне завдання* | *3* |  | |
| *7* | *А3* | *ІАЛЦ.463626.003 Е1* | *Мікропроцесорна система.* | *1* |  | |
| *8* |  |  | *Схема електрична* |  |  | |
| *9* |  |  | *структурна* |  |  | |
| *10* | *А4* | *ІАЛЦ.463626.004 ПЗ* | *Пояснювальна записка* | *33* |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |
|  |  |  |  |  |  | |

*Зм.ю.*

*Арк*.

*№ докум*.

*Пiдпис*

*Дата*

*Розроб.*

*Перевiр.*

*Н. контр.*

*Затв.*

*Куцовол В.В.*

*Лiт.*

Аркуш

Аркушiв

*ІАЛЦ.463626.001 ОА*

*Опис*

*альбому*

НТУУ ''КПІ'' ФІОТ

*Група ІО-82*

*1***2**

*1*

*Ткаченко В.В.*

*.*

*1*

##### ***Технічне завдання***

##### 

*Зм.ю.*

*Арк*.

*№ докум*.

*Пiдпис*

*Дата*

*Розроб.*

*Перевiр.*

*Н. контр.*

*Затв.*

*Куцовол В.В.*

*Лiт.*

Аркуш

Аркушiв

*ІАЛЦ.463626.002 ТЗ*

*Технічне*

*завдання*

НТУУ “КПІ” ФІОТ

*Група ІО-82*

*1***2**

*1*

*Ткаченко В.В.*

*3*

***Зміст***

*1. Найменування завдання……………………………………………………………………………………………………...........2*

*2. Вхідні дані для розробки…………………………………………..………………………………………………….............…2*

*3. Перелік текстової і графічної документації………………………………………………………............3*

***1 Найменування завдання***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*2*

*ІАЛЦ.463626.002 ТЗ*

*Розробити мікропроцесорну систему (МПС) з урахуванням елементної бази, яка вибирається по варіанту. Виконати оцінку ефективності ухвалених технічних рішень.*

*До складу ЕОМ, що розробляється, повинні входити процесор (П), основна пам’ять (ОП), що містить ОЗП і ПЗП, а також зовнішні пристрої (ЗП), контролери переривань і прямого доступу до пам’яті.*

***2 Вхідні дані***

*Варіант вибирається за номером залікової книжки.*

*№ залікової книжки (8213)10, (10000000010101 )2*

*Таблиця 2.1 - Таблиця вхідних даних для розробки*

|  |  |
| --- | --- |
| *Вибір елементної бази* | *1816ВЕ51* |
| *Організація шини* | *З об’єднаними шинами адреси та даних* |
| *Вибір системи команд* | *Комплексна* |
| *КПП. КПДП* | *Централізований - 30, децентралізований – 30* |
| *Спосіб множення, ділення, розрядність операндів* | *Розрядність операндів - 16* |
| *Кількість ЗУ* | *60* |
| *Функція ……………….* |  |
| *Допоміжні порти, периферійні адаптери* | *ВВ55,Р4,Р7* |
| *Адреси для інтерфейсу зовнішнього пристрою* | *Самостійно* |
| *Адреси для ППА* | *98h,99h,9Ah,9Bh* |
| *Зовнішня пам’ять даних* | *64к (15 cторінок)* |
| *Зовнішня пам’ять програм* | *256 (5 сторінок)* |
| *Структурна схема* | *МПС* |

*Завдання видав Ткаченко В.В. \_\_\_\_\_\_\_\_\_\_*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*3*

*ІАЛЦ.463626.002 ТЗ*

*Завдання прийняв Куцовол В.В. \_\_\_\_\_\_\_\_\_\_*

***3 Перелік текстової і графічної документації***

1. *Титульний лист;*
2. *Аркуш з написом «Опис альбому»;*
3. *Опис альбому;*
4. *Аркуш з написом «Технічне завдання»;*
5. *Технічне завдання;*
6. *Аркуш з написом «Мікропроцесорна система. Схема електрична структурна»;*
7. *Мікропроцесорна система. Схема електрична структурна;*
8. *Аркуш з написом «Пояснювальна записка»;*
9. *Пояснювальна записка.*

***Мікропроцесорна система.***

***Схема електрична структурна***

***Пояснювальна записка***

*Зміст*

*Зм.ю.*

*Арк*.

*№ докум*.

*Пiдпис*

*Дата*

*Розроб.*

*Перевiр.*

*Н. контр.*

*Затв.*

*Куцовол В.В.*

*Лiт.*

Аркуш

Аркушiв

*ІАЛЦ.463626.004 ПЗ*

*Пояснювальна*

*записка*

НТУУ ''КПІ'' ФІОТ

*Група ІО-82*

*1***2**

*1*

*Ткаченко В.В.*

*38*

*Вступ……………………………………………………………………………………………………………………………………………………………………….2*

*1 Огляд існуючих рішень…………………………………………………………………………………………………………………………………2*

*1.1 Мікроконтролери: короткий огляд………………………………………………………………………………………………….2*

*1.2 Порівняльна характеристика мікроконтролерів …………………………………………………………………2*

*1.3 Особливості мікроконтролерів сімейства ATmega……………………………………...………………………….6*

*2 Архітектура мікропроцесорної системи……………………………………………………………………………………………6*

*2.1 Вступ…………………………………………………………………………………………………………………………………………………..……….6*

*2.2 Опис мікроконтролера KР1816BE51…………………………………………………………………………………………..……7*

*2.3 Система команд мікроконтролера КР1816ВЕ51.……...……………………………………………………..........15*

*2.4 Підключення зовнішньої пам’яті даних ………………………………………………………………………………..…20*

*2.5 Підключення зовнішньої пам’яті програм..……………………………………….……………………………………..21*

*2.6 Режими доступу до пам’яті. Режим переривань ………………………………………………………………..21*

*2.7 Режими доступу до пам’яті. Прямий доступ до пам’яті…………………………………………..…….25*

*2.8 Підключення додаткових портів…..................................................………………………………………..……25*

*3 Програмна частина…………………………………………………………………………………………………………..………………….....27*

*3.1 Опис програми………………………………………………………………………………….…………………………………………………...27*

*3.2 Розробка алгоритму програми…………………………………………………………….……………………………………..28*

*3.3 Розробка алгоритму підпрограм………………………………………………………………………………………………..29*

*Висновок………………………………………………………………………………………………………………………………………………………….…32*

*Список літератури……………………………………………………………………………………………………....………………………………33*

*записка*

*Додатки*

*Додаток А. Код програми………………………………………………………………………………………………………………34*

*Додаток Б. Список скорочень………………………………………………………………………………………………………38*

***Вступ.***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*2*

*ІАЛЦ.463626.004 ПЗ*

*У даній курсовій роботі виконується розробка мікропроцесорної системи на базі мікроконтролера КР1816ВЕ51. Розробка виконується на підставі «Технічного завдання ІАЛЦ.463626.002 ТЗ».*

***1 Огляд існуючих рішень.***

***1.1 Мікроконтролери короткий огляд***

*Мікроконтролер - мікросхема, призначена для керування електронними пристроями. Типовий мікроконтролер поєднує в собіфункції процесора і периферійних пристроїв, може містити ОЗУ і ПЗУ. По суті, цеоднокристальний комп'ютер, здатний виконувати прості завдання. Велика частинавипускаються в сучасному світі процесорів - мікроконтролери.*

*МК використовуються для створення приладів, ціллю яких є керування та забезпечення роботи складних електромеханічних систем, таких як телефони, пральні машини, контролери двигунів і систем гальмування сучасних автомобілів, з їх допомогою створюються системи контролю і збору інформації.*

*На сьогоднішній день існує бігато фірм, що випускають мікроконтролери, серед них лідуюче положення займають виробники Atmel, Microchip, Motorola, Scinex, Hitachi, Toshiba.*

***1.2 Порівняльна характеристика мікроконтролерів***

*Перий мікроконтроллер з'явився на світ в 1976 році, через 5 років після створення першого мікропроцесора. Це була мікросхема фірми Intel, що отримала ім'я 8048.  
   Крім центрального процесора, на кристалі перебували 1 КБайт пам'яті програм, 64 байта пам'яті даних, два восьмібітних таймера, генератор тактів і 27 портів введення / виводу.  
   Мікроконтролери сімейства 8048 використовувалися в консольних ігрових приставках Magnavox Odyssey, в клавіатурах перших IBM PC і в ряді інших пристроїв.  
   Існує також думка, що першим мікроконтролером був 4-х pозpядний TMS1000 від Texas Instruments, якому міститися ОЗУ (32 байти), ПЗУ (1К), годинник і підтримка введення-виведення, що дозволяло вважати його саме першим мікpоконтpоллеpом. Випущений в 1972 році, він мав нову на той час можливість - додавання нових інструкцій.  
   8051  
   Наступний мікроконтролер Intel 8051, випущений в 1980 році, став воістину класичним зразком пристроїв даного класу. Цей 8-бітний чип поклав початок цілому сімейству мікроконтролерів, які панували на ринку аж до недавнього часу.  
   Аналоги 8051 випускали радянські підприємства в Мінську, Києві, Воронежі, Новосибірську, на них виросло ціле покоління вітчизняних розробників.*

*Більшість фірм виробників мікроконтролерів і сьогодні випускають пристрої, засновані на цій архітектурі. Серед них Philips, Atmel, Dallas, OKI, Siemens - можна перерахувати більше півтора десятків імен. Але 51-е сімейство поступово здає свої позиції більш молодим і досконалим мікроконтролерам.  
   Motorola і Zilog  
   Іншими яскравими представниками восьмирозрядних мікроконтролерів з'явилися вироби компаній Motorola (68HC05, 68HC08, 68HC11) і Zilog (Z8).  
   Motorola тривалий час не надавала коштів, що дозволяють дешево і швидко почати працювати з її контролерами, що явно не сприяло їх популярності у некорпоративних розробників. Проте варто зауважити, що за кордоном мікроконтролери від Motorola займають лідуюче положення на ринку. У нашій країні їх популярність не дуже висока, можливо, ще в силу відсутності достатньої кількості доступних навчальних матеріалів і засобів розробки.  
   Мікроконтролери фірми Zilog, заснованої колишніми співробітниками Intel, що недавно здавалися настільки багатообіцяючими, не витримали гонки в секторі ринку, що стрімко розвивається, і сьогодні система команд Z8 виглядає досить застарілою.  
   Microchip  
   Перші значні зміни відбулися з появою PIC-контролерів фірми Microchip. Ці чіпи пропонувалися за рекордно низькими цінами, що дозволило їм у короткий термін захопити значну частину ринку мікроконтролерів. До того ж кристали від Microchip не поступалися, а нерідко і переважали мікроконтролери х51 по продуктивності і не вимагали дорогого програматора.  
   Разом з контролерами з'явилися дешеві комплекти PICSTART, що містили все, що було потрібно для того, щоб, не маючи ні коштів, ні навичок роботи з PIC-контролерами, швидко створити і налагодити на ньому продукт.  
   Ці мікроконтролери мали зручну роботу з портами, але все інше було зроблено досить незручно. Архітектура залишала бажати кращого, система команд була вкрай обмежена. Тим не менш, PIC-контролери залишаються популярними в тих випадках, коли потрібно створити недорогу систему, що не пред'являє високих вимог щодо її управління.  
   Scinex  
   На хвилі успіху PIC-контролерів з'явилися дуже схожі на них вироби фірми Scinex. Вони мали вже 52 команд проти 33-х команд PIC-контролерів. Були додані зручні інструкції для роботи з пам'яттю, поліпшена архітектура, кожна команда виконувалась за один такт, що за інших рівних умовах було вчетверо швидше, ніж у Microchip, і до того ж їх тактова частота досягала 100 МГц.  
   Настільки висока швидкість контролера дозволяє його творцям відмовитися від різної периферії - таймерів, лічильників, регістрів зсуву в прийомопередавачах, - все це рекомендується реалізовувати чисто програмними засобами, благо швидкодії для цього вистачає: всередині - лише надшвидке ядро, пам'ять та порти вводу / виводу.  
   Atmel  
   Справжня революція в світі мікроконтролерів сталася в 1996 році, коли корпорація Atmel представила своє сімейство чіпів на новому прогресивному ядрі AVR. Більш продумана архітектура AVR, швидкодію, що перевершує контролери Microchip, приваблива цінова політика сприяли відтоку симпатій багатьох розробників від недавніх претендентів на звання контролера номер 1.  
   Мікроконтролери AVR мають більш розвинену систему команд, що налічує до 133 інструкцій, продуктивність, що наближається до 1 MIPS / МГц, Flash ПЗУ програм з можливістю внутрішньосхемного перепрограмування. Багато чіпи мають функцію самопрограмування. AVR-архітектура оптимізована під мову високого рівня Сі. Крім того, всі кристали сімейства сумісні "знизу вгору".  
   Величезну роль 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 знайомство з будь-яким мікроконтролерних сімейством грають не менш значущу роль, ніж самі кристали. Фірма Atmel приділяє цьому питанню велику увагу. Надзвичайно вдале і абсолютно безкоштовна середовище розробки AVR Studio, що працює під Windows було розроблене для розробки та відладки програм для мікроконтролерів з архітектурою фірми Atmel.  
   Провідні сторонні виробники випускають повний спектр компіляторів, програматорів, асемблерів, відладчиків, роз'ємів і адаптерів.  
   Для початківця розробника важливим є і те, що для програмування AVR можна обійтися зовсім без апаратного програматора. Найпопулярнішим сопособом програмування цих мікроконтролерів є п'ять провідків, під'єднаних до паралельного порту персонального комп'ютера.  
  Можна вважати, що AVR поступово стає ще одним індустріальним стандартом серед*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*3*

*ІАЛЦ.463626.004 ПЗ*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*4*

*ІАЛЦ.463626.004 ПЗ*

*8-розрядних мікроконтролерів загального призначення. Вони легкодоступні в Росії і відрізняються в середньому невисокою вартістю, успішно конкуруючи з виробами компанії MICROCHIP. Все це робить мікроконтролери Atmel AVR одними з найпривабливіших для навчання.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*5*

*ІАЛЦ.463626.004 ПЗ*

*Таблиця 1.1 - Порівняння існуючих мікроконтролерів різних фірм-виробників*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *МК* | *ПП* | *ПД* | *Т* | *Приклади* |
| *MCS 51 (Intel)*  *КР1816ВЕ51* | *4к,*  *EEPROM* | *128* | *2* | *Intel 8051, 8X31, 8X32, 8X52; X=0, 3, or 7* |
| *MCS 48 (Intel)*  *КМ1816ВЕ48*  *Зм.*  *Арк.*  *№ докум.*  *Підп.*  *Дата*  *Арк.*  *6*  *ІАЛЦ.463626.004 ПЗ* | *1К, 2K (EP)ROM*  *4K ROM* | *64*  *128*  *256* | *1* | *Intel 8048, 8035, 8038, 8039, 8040, 8050* |
| *megaAVR (Atmel)* | *256к flash*  *SRAM 8 Кб, EEPROM 4 Кб;* | *512*  *1к-4к* | *2*  *3* | *ATmega8, 8L, 48, 16, 8535, 325, 645* |
| *Freescale*  *(мин.Motorola)* | *до 64к* | *до 768* | *2* | *68HC05, 68HC08, 68HC11* |
| *PICmicro* | *до 128к*  *flash* |  | *1*  *2* | *PIC10, PIC12, PIC16, PIC18 (PIC16F628, PICAXE)* |

***1.3 Особливості мікроконтролерів фірми Atmel на ядрі AVR.***

*Мікроконтролери, побудовані на ядрі AVR використовують технологію  конвеєризації, що скорочує цикл "вибірка - виконання" команди. Наприклад, у мікроконтролерів сімейства x51 коротка команда виконується за 12 тактів генератора. В PIC-Контролерах фірми Microchip, де вже реалізований конвеєр, коротка команда виконується за 4 періоди тактової частоти. У мікроконтролерах AVR коротка команда в загальному потоці виконується всього за один період тактируючого сигналу. Така побудова кристала забезпечило істотне підвищення продуктивності, що може досягати значення 1MIPS на 1Мгц. Це в багатьох випадках при заданій продуктивності дозволяло знизити тактову частоту, а виходить, і споживану потужність пристрою. AVR-Мікроконтролери надають більше широкі можливості по оптимізації продуктивності/енергоспоживання, що є особливо важливим при розробці пристроїв з батарейним живленням.*

***2 Архітектура мікропроцесорної системи***

***2.1 Вступ***

*Предметом дослідження даної розрахунково-графічної роботи є побудова мікропроцесорної системи на основі мікроконтролера КР1816ВЕ51.*

*Схема архітектури МПС наведена у документі «Мікропроцесорна система. Схема електрична структурна ІАЛЦ.463626.003 Е2».*

*Зм.*

*Арк.*

*№ докум.*

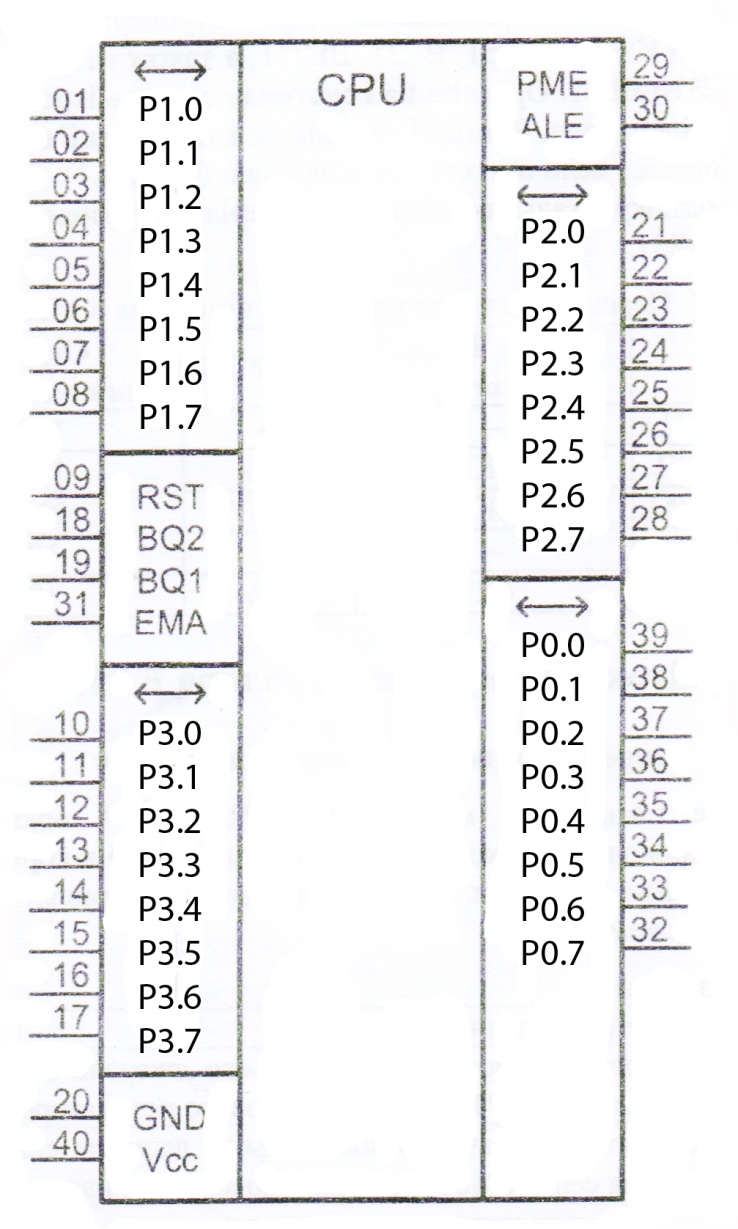
*Підп.*

*Дата*

*Арк.*

*7*

*ІАЛЦ.463626.004 ПЗ*

*****2.2 Опис мікроконтролера KР1816BE51***

*Рисунок 2.1 – Умовне графічне позначення*

***Опис виводів мікросхеми.***

*GND – загальний;*

*Vcc – напруга живлення +5В;*

*BQ1 – вхід для підключення зовнішнього джерела синхронізації або кварцового резонатора;*

*BQ2і – вхід для підключення зовнішнього джерела синхронізації або кварцового резонатора;*

*RST і– вхід сигналу загального скидання;*

*EMA – вхід дозволу зовнішній пам’яті; вхід для подачі імпульсів під час програмування ППЗУ;*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*8*

*ІАЛЦ.463626.004 ПЗ*

*PME – дозвіл читання зовнішньої пам’яті програм;*

*ALE і– строб адреси зовнішньої пам’яті;*

*P0 і– восьмирозрядний двоспрямований порт вводу-виводу;*

*P1, P2, P3 – восьмирозрядні квазідвоспрямовані порти вводу-виводу;*

*Кожна лінія порту Р3 має альтернативну функцію:*

*РЗ.0 – RxD – вхід послідовного порту;*

*Р3.1 і– TxD і- вхід послідовного порту;*

*Р3.2 – INT0 – вхід 0 сигналу запиту переривання від зовнішнього джерела;*

*Р3.3 – INT1 і- вхід 1 сигналу запиту переривання від зовнішнього джерела;*

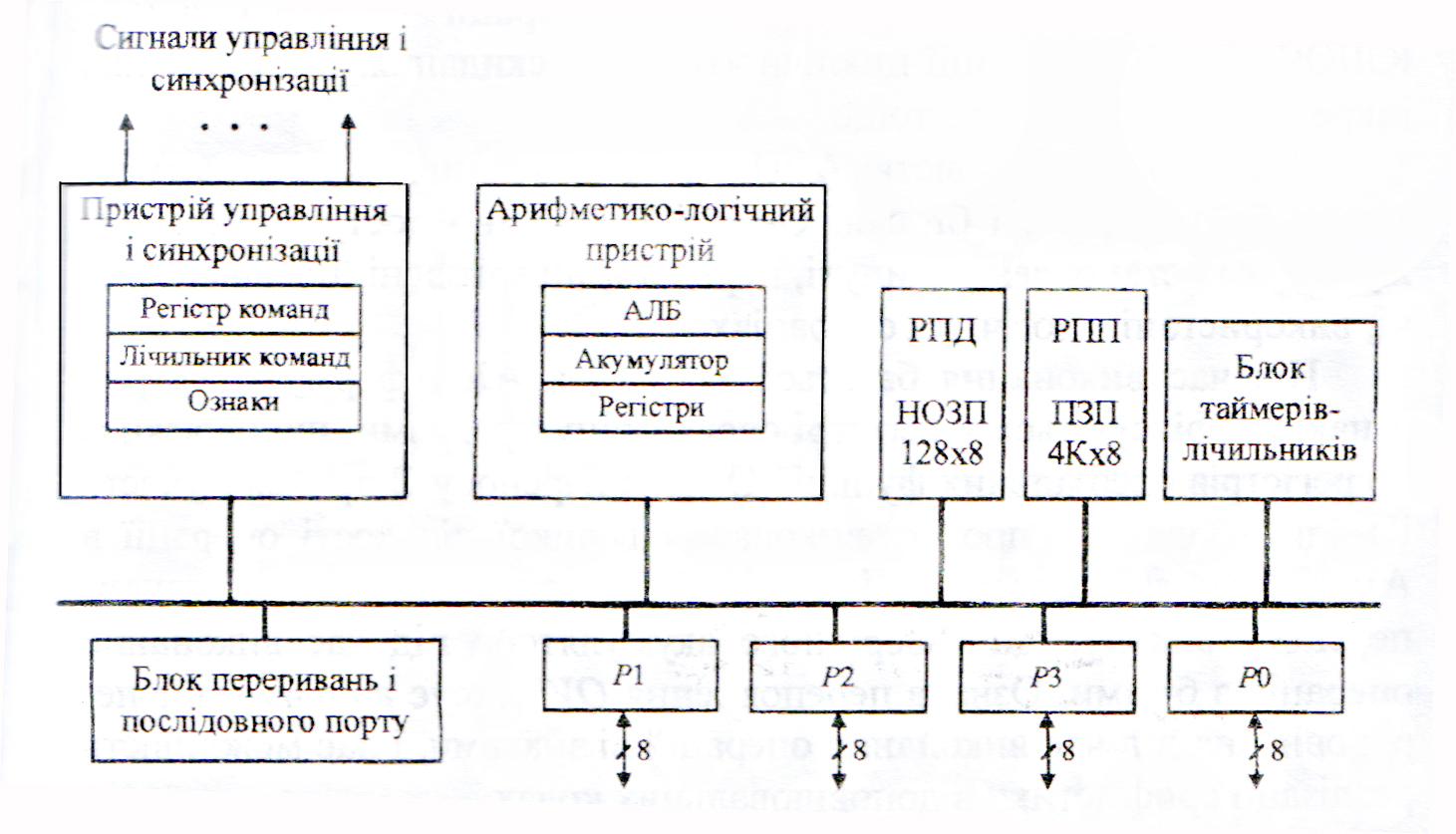
*Р3.4 – T0 ї- вхід лічильника зовнішніх подій Т/Л0;*

*Р3.5 – T1 і- вхід лічильника зовнішніх подій Т/Л1;*

*Р3.6 – WR - вихід сигналу запису в зовнішню пам’ять даних;*

*Р3.7 – RD і- вихід сигналу читання із зовнішньої пам’яті даних;*

***Структура мікроконтролера.***

**

*Рисунок 2.2 – Структурна схема*

*Мікроконтролер містить резидентну пам’ять програм (РПП) та резидентну пам’ять даних (РПД); пристрій управління і синхронізації, до складу якого входить лічильник команд, регістр команд і регістр ознак; арифметико-логічний пристрій, до складу якого входить АЛБ, акумулятор і регістри; блок таймерів-лічильників та блок послідовного інтерфейсу і переривань. Обмін даними здійснюється через чотири порти Р0, Р1, Р2, Р3, або через послідовний порт.*

***РПП.*** *Резидентна пам’ять програм має ємність 4Кб. Призначена для зберігання команд, констант, управляючих слів ініціалізації, таблиць кодування вхідних і вихідних змінних. РПП підключена до шістнадцятибітної шини адреси, що надходить з лічильника команд, або регістру покажчика даних. Організація РПП зображена на рис 2.5..*

***РПД.*** *Резидентна пам’ять даних призначена для зберігання змінних у процесі виконання програми, адресується одним байтом і має ємність 128 байт. До адресного простору РПД належать регістри спеціальних функцій. Організація РПД зображена на рис 2.6.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*9*

*ІАЛЦ.463626.004 ПЗ*

***Арифметико-логічний пристрій.*** *Важливою особливістю АЛП МК51 є можливість оперувати не тільки байтами але й бітами. Окремі програмно доступні біти можуть бути встановлені, скинуті, передані, інвертовані, проаналізовані, використані в логічних операціях. Під час виконання багатьох команд в АЛП формується ряд ознак, які фіксуються в регістрі слова стану програми, що належить до регістрів спеціальних функцій.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*10*

*ІАЛЦ.463626.004 ПЗ*

***Регістри спеціальних функцій.*** *Умовні позначення регістрів спеціальних функцій зображені на моделі програміста (рис. 2.3).*

*DPTR – регістр-покажчик даних;*

*ACC\* – акумулятор;*

*B\* І– регістр-розширювач акумулятора;*

*SP і– покажчик стека;*

*PSW\* – слово стану програми;*

*P0\* – порт 0;*

*P1\* і– порт 1;*

*P2\* і– порт 2;*

*P3\* і– порт 3;*

*IP\* – регістр пріоритетів;*

*IE\* – регістр маски переривань;*

*TMOD і– регістр режиму таймерів/лічильників;*

*TCON\* – регістр управління/статусу таймера;*

*TH0 – таймер 0 (старший байт);*

*TL0 – таймер 0 (молодший байт);*

*TH1 – таймер 1 (старший байт);*

*TL1 – таймер 1 (молодший байт);*

*SCON\* – регістр управління приємопередавачем;*

*SBUF – буфер приємопередавача;*

*PCON – регіср управління потужністю;*

*Регістри, помічені позначкою (\*), допускають адресацію окремих біт.*

***Акумулятор.*** *Аккумулятор АСС це восьмирозрядний регістр, який використовується як приймач або джерело операнда під час виконання арифметичних і логічних операцій та ряду операцій передачі даних. За застосування акумулятора виконуються операції зсувів, перевірки на нуль, формування ознаки парності.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*11*

*ІАЛЦ.463626.004 ПЗ*

***Регістр В.*** *Регістр В використовується під час виконання операцій множення і ділення. Під час виконання інших операцій може використовуватись як допоміжний регістр.*

***Регістр слова стану програми.*** *PSW призначений для зберігання інформації про ознаки, що формуються в АЛП під час виконання обчислень.*

*С - ознака переносу;*

*АС i- ознака додаткового переносу;*

*F0 i- ознака нуля;*

*RS1, RS2 – вибір банку регістрів;*

*OV i- ознака переповнення;*

*P - ознака парності;*

***Покажчик стека.*** *Восьмибітний покажчик стека SP може адресувати будь-яку область РПД і призначений для зберігання адреси комірки стека, до якої було останнє звернення.*

***Регістр-покажчик даних.*** *Двобайтний DPTR застосовується для фіксації шістнадцятибітної адреси під час виконання операції звернення до зовнішньої пам’яті.*

***Таймер/лічильник.*** *У склад МК входять регістрові пари TH0, TL0, TH1, TL1, на основі яких функціонують два незалежних шістнадцятибітних таймера/лічильника подій.*

***Буфер приймача/передавача.*** *SBUF складається з двох незалежних регістрів – буферу приймача і буферу передатчика. Завантаження байта в регістр викликає початок процесу передачі через послідовний порт.*

***Пристрій управління та синхронізації.*** *Пристрій управління та синхронізації є кварцовим резонатором, що підключається до зовнішніх виводів Х1 та Х2 корпуса мікросхеми МК51, управляє роботоювнутрішнього генератора, який в свою чергу формує сигнали синхронізації.*

*Зм.*

*Арк.*

*№ докум.*

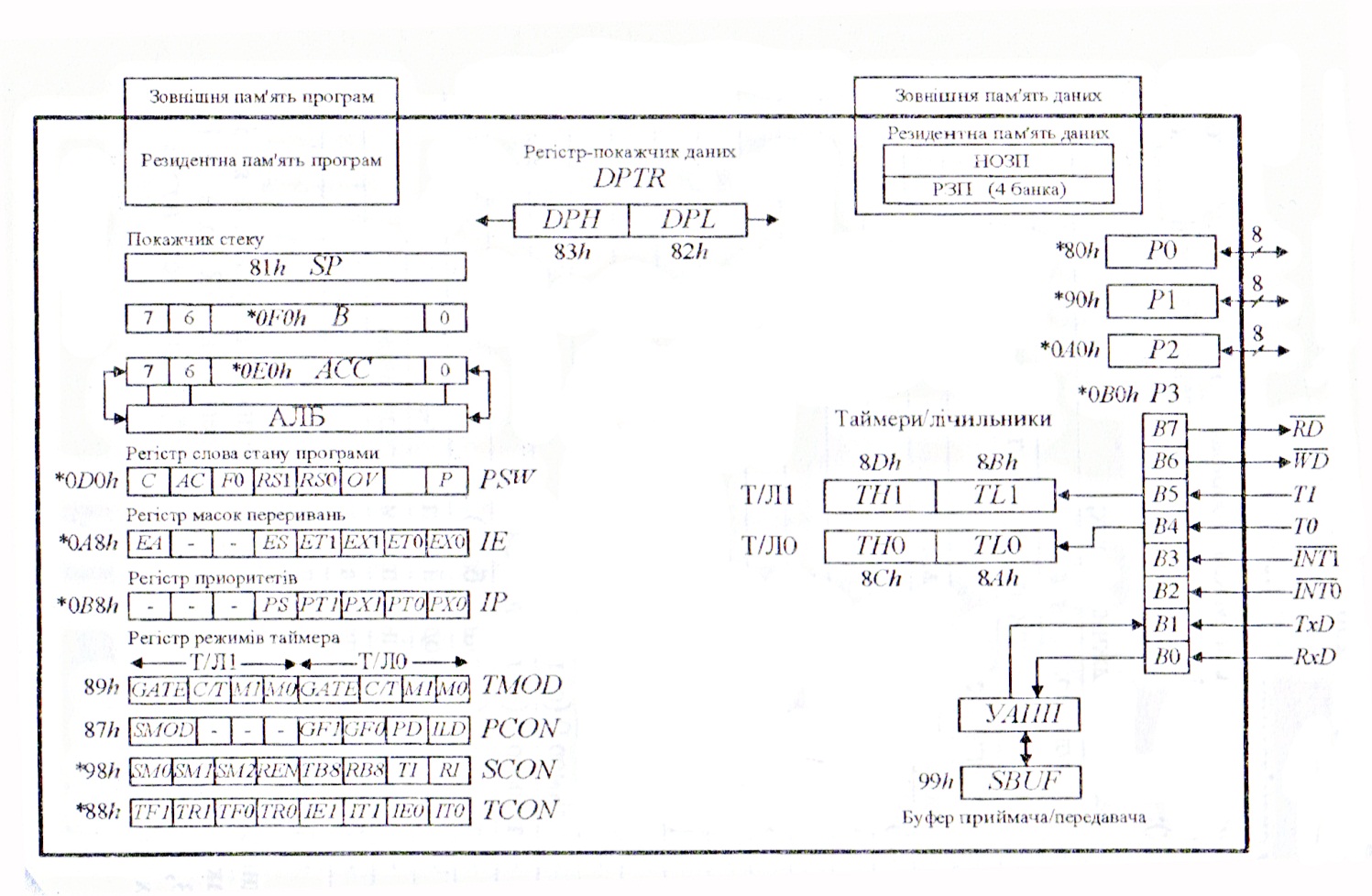
*Підп.*

*Дата*

*Арк.*

*12*

*ІАЛЦ.463626.004 ПЗ*

****

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*13*

*ІАЛЦ.463626.004 ПЗ*

*Рисунок 2.3 – Модель програміста*

***Організація пам’яті.*** *МК51 має фізично і логічно розділену пам’ять програм і пам’ять даних, ємність кожної з котрих може бути розширена до 64Кб за рахунок підключення зовнішніх мікросхем пам’яті. Пам’ять програм і пам’ят даних мають різні механізми адресації і працюють під управлінням різних управляючих сигналів. Організація пам’ті ілюструється на рис. 2.4.*

*Зм.*

*Арк.*

*№ докум.*

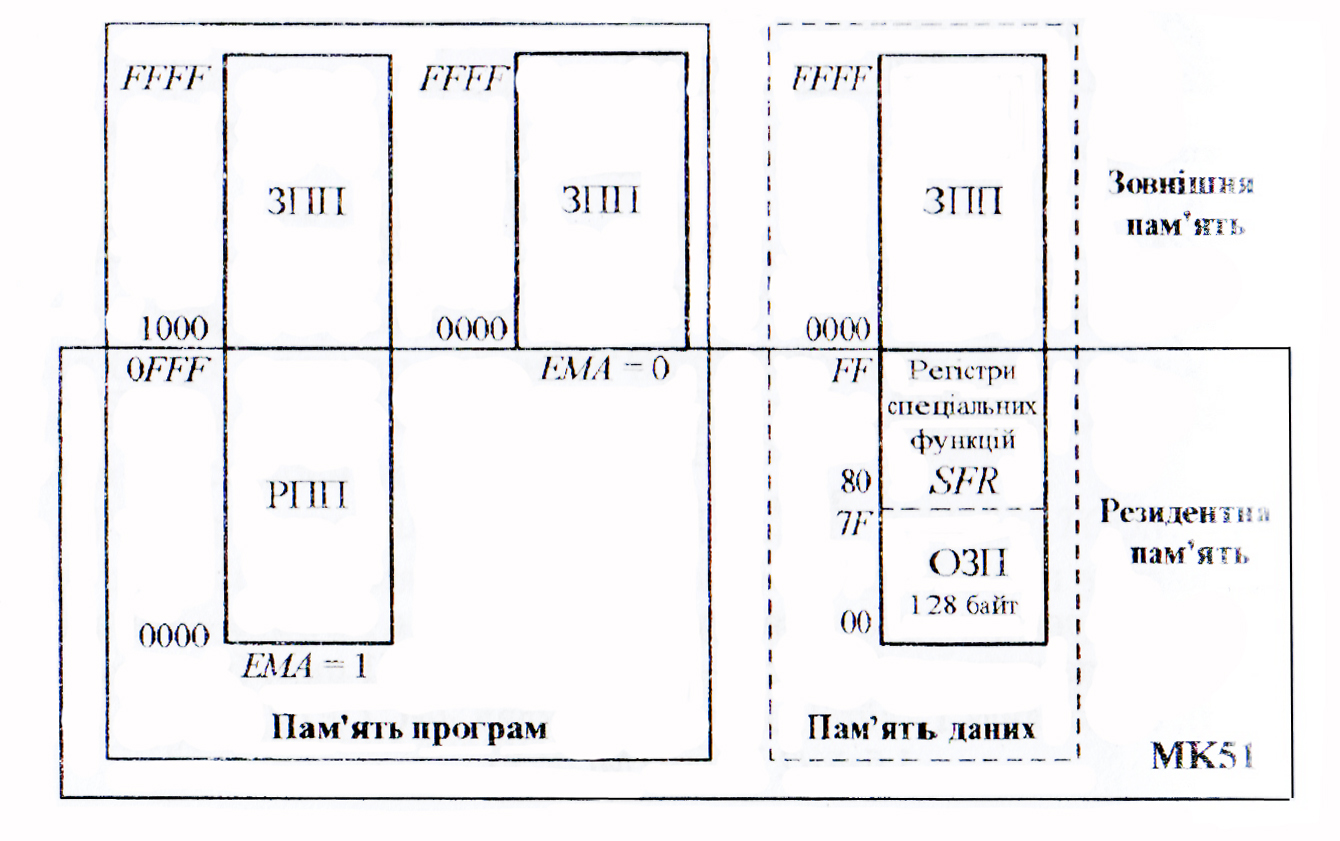
*Підп.*

*Дата*

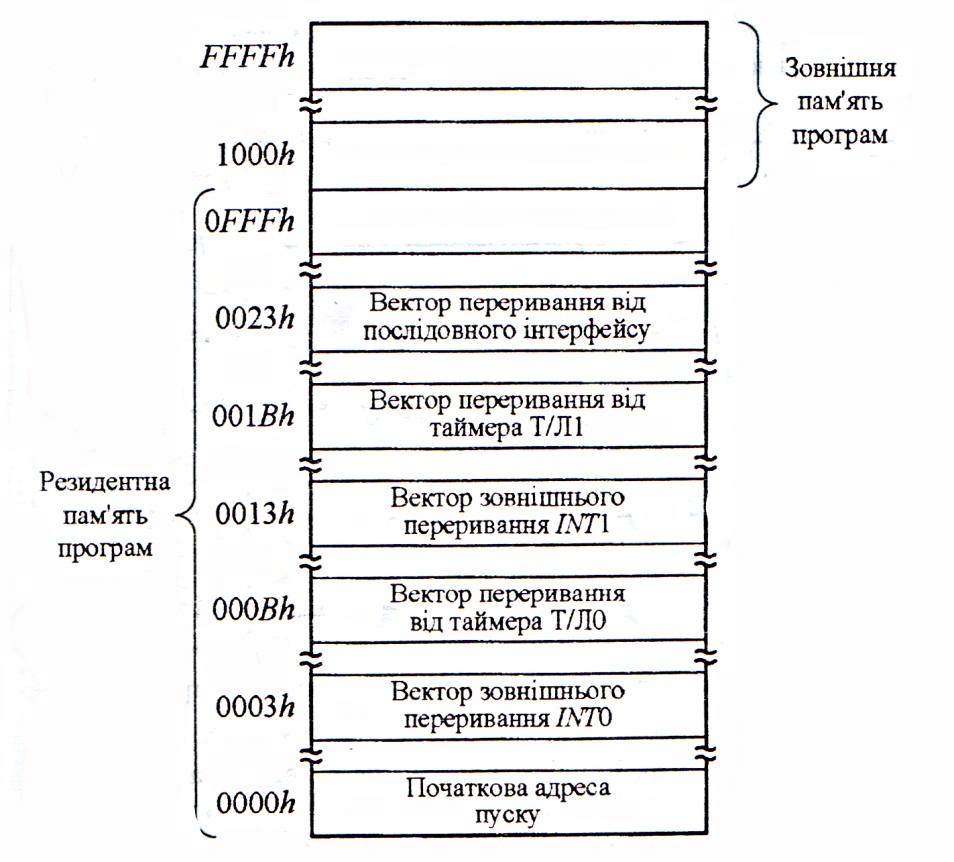
*Арк.*

*13*

*ІАЛЦ.463626.004 ПЗ*

**

*Рисунок 2.4 – Організація пам’яті*

**

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

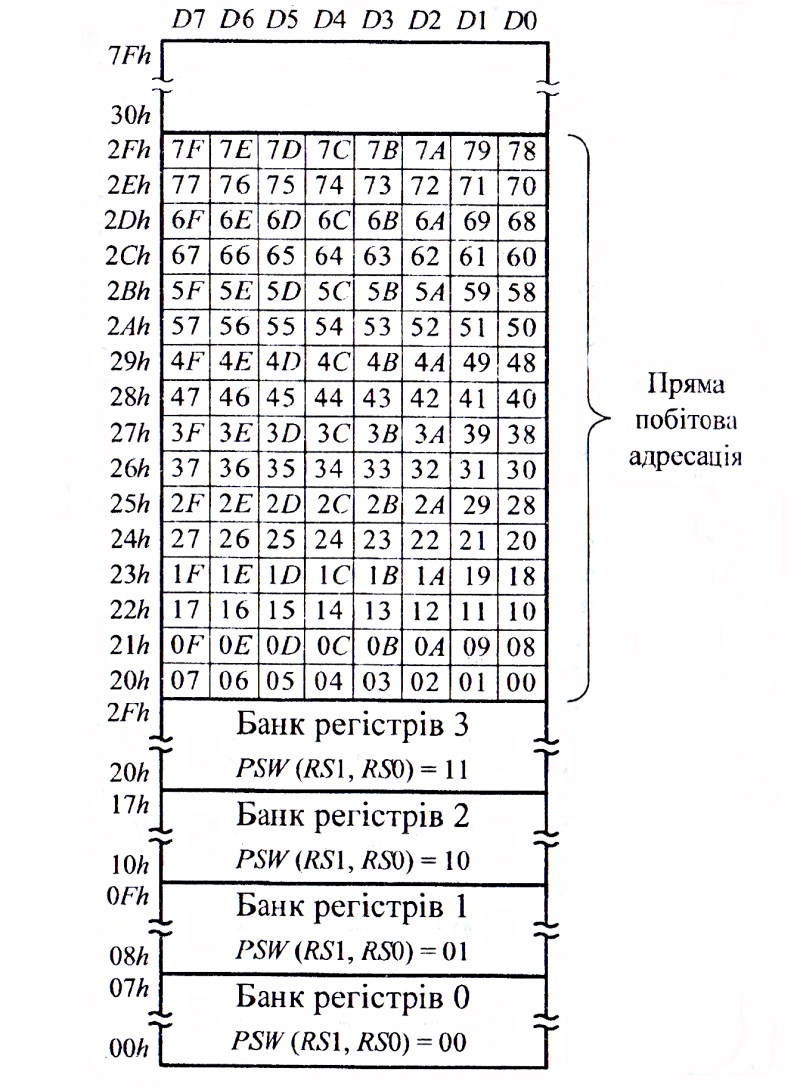
*Дата*

*Арк.*

*14*

*ІАЛЦ.463626.004 ПЗ*

*Рисунок 2.5 – Організація РПП*

****

*Рисунок 2.6 – Організація РПД*

***Таймери/лічильники.*** *Два шістнадцятирозрядні таймери-лічильники – Т/Л1 і Т/Л0, призначені для отримання програмно керованих часових затримок і підрахунку зовнішніх подій.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*15*

*ІАЛЦ.463626.004 ПЗ*

***Порти вводу/виводу.*** *Порти вводу/виводу Р0, Р1, Р2, Р3 можуть бути використані для організації вводу/виводу інформації за тридцяти двома лініями передачі. Порт Р0 є двоспрямованим, а порти Р1, Р2, Р3 – квазідвоспрямованими. Їх особливість полягає в тому, що під час вводу над вхідними даними і поточним станом порту – даними, які виводилися з порту останніми, виконується порозрядна логічна операція І. Вихідні дані з порту запам’ятовуються.*

***Блок послідовного інтерфейсу і переривань.*** *Блок послідовного інтерфейсу і переривань призначений для вводу і виводу послідовної інформації і організації системи переривання програм. У його склад входять універсальний асинхронний приймач/передавач УАПП, буфер приймача/передавача SBUF, регістр управління/статусу УАПП SCON, регістри масок переривань IE і пріоритетів переривань IP.*

*УАПП, який називається також послідовним портом, призначений для обміну інформацією, поданою послідовним кодом.*

*Система переривань дозволяє автоматично реагувати на зовнішні і внутрішні події.*

***2.3 Система команд мікроконтролера КР1816ВЕ51.***

*Система команд мікроконтролера КР1816ВЕ51 містить сто одинадцять команд. Відносно функціональних ознак команди класифікуються за наступними групами:*

1. *команди передачі даних;*
2. *команди виконання арифметичних операцій;*
3. *команди виконання логічних операцій;*
4. *команди виконання операцій з бітами;*
5. *команди передачі управління;*

*Команди МК51 мають довжину один, два або три байти і виконуються відповідно за один, два або чотири машинні цикли. За тактової частоти генератора f=12Мгц тривалість циклу складає 1 мкс.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*16*

*ІАЛЦ.463626.004 ПЗ*

*Операнди можуть бути чотирьох типів:*

1. *однобітні операнди (біти), в якості яких можливо застосовувати окремі біти регістрів спеціальних функцій SFR та портів; для адресації бітів застосовуються пряма восьмибітна адреса (bit); непряма адресація бітів неможлива;*
2. *чотирибітні операнди (тетради) застосовуються під час виконання операцій обміну тетрадами SWAP та XCHD;*
3. *восьмибітні операнди (байти), являють собою комірки пам’яті програм або даних, порти вводу/виводу; порти та регістри спеціальних функцій SFR адресуються тільки прямим способом; байти пам’яті можуть адресуватися і непрямим способом, за допомогою адресних регістрів R0, R1, DPTR, PC;*
4. *двобайтні операнди це константи та прямі адреси, для подання яких застосовуються другий і третій байти команди.*

*У мікроконтролері МК51 застосовуються наступні способи адресації джерел операндів:*

1. *пряма адресація;*
2. *безпосередня адресація;*
3. *пряма регістрова адресація;*
4. *пряма побітова адресація;*
5. *непряма регістрова адресація;*
6. *непряма адресація за сумою базового індексного регістра.*

*Перші три із зазначених способів адресації використовуються для адресації приймачів операндів. Використовувані способи адресації, поєднані в різних комбінаціях, забезпечують двадцять один режим адресації.*

*Формати більшості команд містить поля: «приймач» і «джерело» операндів, які визначають тип даних, способи адресації і саме операнди, що приймають участь в обчисленнях.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*17*

*ІАЛЦ.463626.004 ПЗ*

*В командах, що не виконують операції перезапису, операнд призначення є операндом-джерелом.*

*Велика кількість команд включає операнди, розташовані у внутрішньому ОЗП даних МК51. Вибір адресного простору пам’яті програм або зовнішньої пам’яті даних як другий операнд визначається командною мнемонікою (якщо тільки другий операнд не є безпосередньою величиною).*

*Область внутрішнього ОЗП даних, що адресуються, визначається способом адресації і величиною адреси. Наприклад, звернення до регістрів спеціального призначення може бути виконане тільки за допомогою прямої адресації.*

*Пряма адресація використовується для звернення до комірок резидентної пам'яті даних ОЗП з адресами (0 - 127) і до регістрів спеціального призначення.*

*Пряма регістрова адресація використовується під час звернення до робочих регістрів. Пряма регістрова адресація використовується також для звернення до акумулятора АСС, регістрів В, АВ (АВ - використовується, як регістр подвійної довжини), регістру DPTR і до ознаки переносу С.*

*Пряма побітова адресація використовується для звернення до прямо адресованих бітів комірок ОЗП з адресами 20Н - 2FH. Вказані комірки послідовно пронумеровані від молодшого біта молодшого байта до старшого біта старшого байта.*

*Непряма регістрова адресація використовується під час звернення до комірок резидентної пам'яті даних, при цьому як покажчики адреси використовуються регістри R0, R1 вибраного банку регістрів. У командах PUSH і POP в якості покажчика адреси використовується вміст покажчика стека SP. Під час звернення до будь-якої комірки адресного простору зовнішньої пам'яті даних об'ємом до 64 Кб застосовується шістнадцятирозрядний покажчик даних DPTR.*

*Непряма регістрова адресація за сумою базового і індексного регістрів використовується для читання даних з пам'яті програм, зокрема для вибірки елементу з порядковим номером з таблиці.*

*Команди, виконання яких модифікує ознаки результату, наведені у табл. 2.1. Значення ознаки парності змінюється всіма командами, що змінюють вміст акумулятора. Окрім зазначених у табл. 2.1 команд ознаки змінюють команди, в яких призначення результату визначає регістр PSW, або його окремі біти, а також операції над бітами.*

*Таблиця 2.1 - формування ознак результату*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*18*

*ІАЛЦ.463626.004 ПЗ*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Мнемоніка* | *Ознаки* | | | *Мнемоніка* | *Ознаки* | | |
| *ADD* | *C* | *OV* | *AC* | *CLR C* | *0* | *-* | *-* |
| *ADDC* | *C* | *OV* | *AC* | *CPL C* | *C* | *-* | *-* |
| *SUBB* | *C* | *OV* | *AC* | *ANL C, bit* | *C* | *-* | *-* |
| *MUL* | *0* | *OV* | *-* | *ANL C, /bit* | *C* | *-* | *-* |
| *DIV* | *0* | *OV* | *-* | *ORL C, bit* | *C* | *-* | *-* |
| *DA* | *C* | *-* | *-* | *ORL C, /bit* | *C* | *-* | *-* |
| *RRC* | *C* | *-* | *-* | *MOV C, bit* | *C* | *-* | *-* |
| *RLC* | *C* | *-* | *-* | *CJNE* | *C* | *-* | *-* |
| *SETB C* | *1* | *-* | *-* |  |  |  |  |

*Приклад прямої адресації:*

*ADD A, 31h ; Додавання вмісту акумулятора до вмісту комірки резидентної*

*; пам'яті з адресою 31h. Результат розміщується в акумуляторі.*

*Приклад прямої регістрової адресації:*

*MOVA, R5 ; Завантаження акумулятора операндом із регістра R5*

*Приклад непрямої регістрової адресації:*

*ADD A, @R0 ; Складання вмісту акумулятора з вмістом комірки пам'яті,*

*; адреса якої знаходиться в регістрі R0*

*Приклад непрямої регістрової адресації за сумою базового і індексного регістрів:*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*19*

*ІАЛЦ.463626.004 ПЗ*

*MOVC A, @A+DPTR ; Пересилання в акумулятор вмісту комірки пам'яті програм,*

*; адреса якої обчислюється шляхом додавання двобайтного*

*; значення регістра-покажчика DPTR і однобайтного операнду*

*; без знаку, розміщеного в акумуляторі.*

*; Цю операцію інтерпретують, як читання елементу з номером*

*; i, заданим в акумуляторі, із таблиці, початкова адреса,*

*; якої задана в DPTR*

*Приклади виконання команд передачі даних:*

*MOV @R0,#10 ; запис числа 10 в байт на який вказує R0*

*MOV DPTR, 7F00H ; завантаження початкової адреси масиву даних*

*MOV TCON, 00000101B ; скидання області бітів*

*Приклад виконання арифметичних операцій. Складання двох двійкових багатобайтних чисел. Обидва доданки розташовуються у РПД послідовно, розпочинаючи з молодшого байту. Початкові адреси доданків задані в R0 і R1. Формат доданків, що визначає кількість байтів задано в R2.*

*CLR C ; скидання переносу*

*L1: MOV A, @R0 ; завантаження в акумулятор*

*; поточного байту першого доданку*

*ADDC A, @R1 ; складання байт з урахуванням переносу*

*MOV @R0, A ; розміщення байта результату*

*INC R0 ; просування покажчиків*

*INC R1*

*DJNZ R2, L1 ; цикл, якщо не всі байти додані*

*Приклад виконання логічних операцій (обчислення логічної функції F = 2(X1 v X2), вихідні дані розміщуються в регістрах R1, R2):*

*MOV A,R1 ; пересилання R1 в А*

*ORL A,R2 ; виконання (Х1 v X2)*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*20*

*ІАЛЦ.463626.004 ПЗ*

*CLR C ; встановлення С := 0*

*RLC A ; зсув вліво 2(Х1 v Х2)*

*MOV R2,A ; пересилання результату А в R2*

*Приклади виконання операції з бітами:*

*ANL D0H, 11100111B ; скидання бітів RS1 і RS0*

*ORL P1, 00001111B ; P1[3..0] <- 1111*

*ANL P2, 10111010B ; скидання бітів P2[6], P2[2,0] порту P2*

*XRL P1,A ; ВИКЛЮЧНЕ АБО порту Р1 і акумулятора*

*Приклад виконання команд передачі управляння. Обчислення Х4 = Х1 & Х2, якщо Р1[0] = 1, та Х4 = Х3 - 1, якщо P1[0] = 0. Вихідні дані розміщені у регістрах R1, R2, R3, R4 відповідно.*

*ORL P1,00000001B ; налагодження нульового біту Р1[0] на ввід*

*JB P1.5,L1 ; перехід, якщо Р1[0] = 1*

*DEC R3 ; виконання Х3 - 1,*

*MOV A, R3 ; пересилання (А) <- (R3)*

*JMP L2 ; перехід на кінець*

*L1: MOV A, R1 ; пересилання (А) <- (R1)*

*ANL A,R2 ; виконання Х1 & Х2*

*L2: MOV R4, A ; пересилання (R4) <- (A)*

***2.4 Підключення зовнішньої пам’яті даних***

*Зовнішня пам’ять даних реалізується за допомогою додаткових мікросхем пам’яті і може мати ємність до 64 Кб з адресами 0 – FFFFFh.*

*Схема підключення ОЗП (рис. 2.4.1)*



*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*21*

*ІАЛЦ.463626.004 ПЗ*

*Рисунок 2.4.1. Схема підлючення зовнішньої пам’яті даних.*

***2.5 Підключення зовнішньої пам’яті програм***

*Підключення зовнішньої пам’яті програм (рис. 2.5.1) у випадку, коли ЕМА = 0, показано на рис. 2.5.1, де D та A входи даних та адреси відповідно.*



*Рисунок 2.5.1. Схема підключеня зовнішньої пам’яті програм.*

***2.6 Режим переривань.***

*Переривання можна класифікувати наступним чином: внутрішні і зовнішні, у свій час внітрішні діляться на апаратні і програмні, а зовнішні - на векторні і безвекторні.*

*В апаратних перериваннях вимога переривання формується певними схемами процессора при появі деяких подій (ділення на нуль, зависання при зверненні до пам’яті або ЗП тощо).*

*Програмні переривання визиваються при виконанні команд переривання програми.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*22*

*ІАЛЦ.463626.004 ПЗ*

*Внутрішние переривання мають фіксовані початкові адреси підпрограм для їх обслуговування.*

*Процесор має спеціальні входи для запитів на переривання програми. Для деяких входів існують стандратні підпрограми обслуговування з фіксованою початковою адресою.*

*Будь-якому ЗП можна дозволити переривання програми. Для подачі сигналу такого переривання використовується один вхід процессора. Ідентифікація пристрою процесором здійснюється за допомогою читання на шині даних вектора (номера ЗП). спеціальна процедура на апаратному чи програмному рівні ставить у відповідність вектору початкову адресу програми обслуговування.*

*В процесі ініціалізації системи процесор записує в регістр стану ЗП одиницю в біт дозволу переривань, якщо цей пристрій буде працювати в режимі переривання.*

*Крім цього процесор може записувати вектор в регістр вектора.*

*Коли ЗП готовий до обміну, встановлюється біт готовності в регістрі стану своїм контролером.При збігу сигналів готовності і дозволу переривань формується сигнал вимоги переривань на загальній однопровідній шині.*

*Процесор перевіряє сигнал після виконання команди і формує послідовно два сингали по шині керування: підготовка і попередження переривання.*

*По сигналу підготовка у всіх ЗП забороняється зміна всіх тригерів. У кожному інтерфейсі комутується шлях проходження сигналу підтвердження переривань. Якщо ЗП запросило переривання, то ланцюг проходження далі сигналу ПП розривається, а в цьому ЗП по даному сигналу видається вектор на шині даних, який приймається процесором.*

*В цілому апаратно переривання діляться на:*

*1) Централізовані (з централізованим КПП);*

*2) Децентралізовані (з розподіленим КПП).*

*Схема підключення централізованого контролера пріоритетних переривань показана на рис. 2.6.1. ЗП, в числі яких можуть бути 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 у процесор передає КПП, а не зовнішні пристрої. Завдяки цьому через переривання можуть взаємодіяти процесори, підключені до різних магістралей.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*23*

*ІАЛЦ.463626.004 ПЗ*



*Рис.2.6.1. Система з централізованим контролером пріоритетних переривань.*

*До переваг 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.*

*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:*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*24*

*ІАЛЦ.463626.004 ПЗ*

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

*Під час використання розподіленого контролера(рис.2.6.2) переривань всі ЗП повинні бути підключені до магістралі процесора. Це обумовлено тим, що вектор переривання на шину даних в даному випадку передає 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. У відповідь сигнал процесора IACK розповсюджується послідовно через елементи БКП, створюючи так званий пріоритетний ланцюжок (daisy chain). Елементи ланцюжка в кожному БКП пропускають сигнал IACK або розривають ланцюжок. Пріорітетний ланцюжок розривається на першому активному ЗП, який виставив сигнал запиту IRQ. Даний активний ЗП видає на шину даних вектор переривання, який приймається процесором.*

*Рисунок 2.6.2. Система з децентралізованим КПП*

*Блок БКП містить доступні для процесора регістр стану РС і регістр вектора РВ. Під час ініціалізації системи процесор записує в регістр РВ вектор переривання, а в регістр РС - біт дозволу переривання. Якщо ЗП готовий до взаємодії з процесором, то в регістрі РС встановлюється біт готовності.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*25*

*ІАЛЦ.463626.004 ПЗ*

*До переваг цієї схеми можна віднести:*

*- невелику кількість ліній зв’язку в шині управління;*

*- простоту нарощування числа ЗП.*

*До недоліків:*

*- велика кількість звернень процесора до магістралі під час ініціалізації системи;*

*- використання фіксованих рівнів пріоритетів запитів, які не можна змінювати динамічно.*

***2.7 Режими доступу до пам’яті. Прямий доступ до пам’яті***

*Даний режим використовується для розвантаження процесора при обміні масивами даних між ОП і ЗП.*

*Ініціатором обміну являється процесор, який виконує ініціалізацію контролера прямого доступу до пам’яті і запускає його.*

*Надалі два активних пристрої (процесор і КПДП) по черзі захоплюють шину, за рахунок чого здійснюється паралельна робота цих пристроїв.*

*КПДП має ряд адрес в адресному просторі ЗП.*

***2.8 Підключення додаткових портів***

*Для збільшення кількості ліній зв'язку МК51 з об'єктом управ­ління підключають додаткові чотирирозрядні порти Р4, Р5, Р6, Р7.**Найбільш просто це здійснюється за використання спеціальної ІС КР580 ВР43, спосіб підключення якої до МК51 показаний на рис. 2.8.1. В цьому випадку забезпечується виконання всіх чотирьох команд роботи з додатковими портами - MOVD А, Рр; MOVD Рр, A; ANLD Рр, А та ORLD Рр,А (де p = 4,7), причо­му кожний вихід порту може бути налаштований як на введення так і на виведення інформації.*

*Команди передачі інформації між МК51 та додатковими порта­ми виконуються за два цикли. В першому циклі на виходах Р2[3..0] встановлюється управляюче слово, в другому циклі - через зазначе­ні виходи здійснюється обмін інформацією між МК48 та одним з додаткових портів. Для стробування даних в режимі підключення додаткових портів використовується сигнал**PROG.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*26*

*ІАЛЦ.463626.004 ПЗ*

*Відмітимо, що логічні операції І та АБО виконуються безпосе­редньо в ІС ВР43. Це необхідно враховувати при побудові додатко­вих портів з використанням інших апаратних засобів.*

*Для розширення функціональних можливостей системи до МК51 можна підключати різні ІС, наприклад, адаптери КР580ВВ51, КР580ВВ55 тощо.*



*Рисунок 2.8.1. Схема з’єднання виходів МК51 та ІС КР580ВР43*

***3 Програмна частина***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*27*

*ІАЛЦ.463626.004 ПЗ*

***3.1 Опис програми***

*В даній розрахунково-графічній роботі розроблено програму для обчилення арифметичного виразу згідно з технічним завданням.*

*Вхідні дані для розроблюваної програми представлено в таблиці 3.1.*

*Таблиця 3.1 - Вхідні дані для розробки програми*

|  |  |
| --- | --- |
| *Вираз для обчислення* |  |
| *Порт, з якого вводяться дані* | *Р0* |
| *Розрядність вхідних операндів* | *16* |
| *Розрядність реультату* | *32* |
| *Регістри ОЗП для зберігання проміжних операндів* | *Х1: #30h:#31h*  *Х2: #32h:#33h*  *Х5: #35h:#34h*  *Х6: #36h:#37h*  *Х9: #38h:#39h*  *Х10: #3Ah:#3Bh* |
| *Формат представлення операндів* | *прямий код* |
| *Регістри ОЗП для зберігання проміжних операндів при виклиці підпрограм* | *#20h:#21h, #22h:#23h,*  *#24h:#25h:#26h:#27h* |
| *Регістри ОЗП для зберігання проміжних результатів* | *#20h:#21h:#22h:#23h,*  *#02h:#03h:#04h:#05h* |
| *Регістри ОЗП для зберігання остаточного результату X* | *#20h:#21h:#22h:#23h* |
| *Порт, на який виводяться результати* | *Р0* |

*Алгоритм програми для обчислення виразу представлено на рис. 3.1. Код програми, виконаний на асемблері МК51, приведено у Додатку А.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*28*

*ІАЛЦ.463626.004 ПЗ*

***3.2 Алгоритм програми***



*Рисунок 3.1 - Блок-схема алгоритму програми*

***3.3 Алгоритми підпрограм***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*29*

*ІАЛЦ.463626.004 ПЗ*

*Опис алгоритму MUL:*

1. *Операція: X\*Y=Z, операнди X і Y 16-розрядні.*
2. *Вхідні дані: X в #22h,#23h-множене, Y в #20h,#21h-множник.*
3. *Виділення знаків та основних розрядів.*
4. *Виконати команду MUL @21h,@23h.*
5. *Виконати команду MUL @21h,@22h.*
6. *Виконати команду MUL @20h,@23h.*
7. *Виконати команду MUL @20h,@22h*
8. *Результат в #02h:#03h:#04h:#05h за схемою на рис. 3.2.*



*Рис 3.2 Схема отримання результату підпрограми MUL побайтним множенням операндів*

*Опис алгоритму SUB:*

1. *Операція: X-Y=Z, операнди X і Y 32-розрядні.*
2. *Вхідні дані: X в #24h,#25h,#26h,#27h-зменшуване, Y в #20h,#21h,#22h,#23h-від"ємник.*
3. *Переведення операндів у доповнювальний код.*
4. *Виконати команду SUBB @27h,@23h.*
5. *Виконати команду SUBB @26h,@22h.*
6. *Виконати команду SUBB @25h,@21h.*
7. *Виконати команду SUBB @24h,@20h.*
8. *Результат в #20h,#21h,#22h,#23h.*

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*30*

*ІАЛЦ.463626.004 ПЗ*

*Опис алгоритму ADD:*

1. *Операція: X+Y=Z, операнди X і Y 32-розрядні.*
2. *Вхідні дані: X в #24h,#25h,#26h,#27h-доданок1, Y в #20h,#21h,#22h,#23h-доданок2.*
3. *Переведення операндів у доповнювальний код.*
4. *Виконати команду ADD @27h,@23h.*
5. *Виконати команду ADDC @26h,@22h.*
6. *Виконати команду ADDC @25h,@21h.*
7. *Виконати команду ADDC @24h,@20h.*
8. *Результат в #20h,#21h,#22h,#23h.*

*Примітка: можна оптимізувати код, використовуючи цикл.*

*Опис вищезазначених алгоритмів зображений на відовідних блок-схемах (рис. 3.3-3.5).*



*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*31*

*ІАЛЦ.463626.004 ПЗ*

*Рис 3.3 Блок-схема алгоритму множення 16-розрядних операндів*



*Рис 3.3 Блок-схема алгоритму віднімання 16-розрядних операндів*



*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*32*

*ІАЛЦ.463626.004 ПЗ*

*Рис 3.3 Блок-схема алгоритму додавання 16-розрядних операндів*

***Висновок***

*У даній розрахунково-графічній роботі проведено дослідження мікропроцесорної системи на основі мікроконтролера 1816ВЕ51. Кожен з розділів розрахункової роботи дозволяє покращити практичні знання у використанні даного контролера для побудови мікропроцесорної системи.*

*У першому розділ розглядаються мікроконтролери, що найбільше використовуються у сучасному виробництві та було приділено особливу увагу контролеру 1816ВЕ51.*

*У другому розділі описується розробка мікропроцесорної системи. Було розглянуто як сам мікроконтролер 1816ВЕ51, так і підключення його до нього зовнішньої пам’яті та периферійних пристроїв. У роботі приведено систему команд контролера, розглянуто особливості його архітектури. Було розроблено структурну схему МПС.*

*У третьому розділі розглядається приклад побудови програми для обчислення арифметичного виразу на даному мікроконтролері. Використовувався асемблер МК51.*

*Отже, в даній роботі докладно розглянуто принципи побудови мікропроцесорної системи на основі 1816ВЕ51.*

***Список літератури***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*33*

*ІАЛЦ.463626.004 ПЗ*

1. *В.І.Жабін, І.А.Жуков, В.В.Ткаченко, І.А.Клименко. МІКРОПРОЦЕСОРНІ СИСТЕМИ. Навчальний посібник. Київ, Видавництво «СПД Гуральник О.Ю.», 2009.*
2. *Wikipedia - free encyclopedia,* <http://uk.wikipedia.org/>

***Додаток А***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*34*

*ІАЛЦ.463626.004 ПЗ*

*Лістинг програми.*

ljmp start ;переходимо на точку старту

;операція X\*Y=Z

MUL:mov r2,#0h ;очистка

mov r3,#0h ;байтів

mov r4,#0h ;результату

mov r5,#0h ;r2,r3,r4,r5

jnb 16.7, n1 ;перевіряємо знак множеного, якщо він нульовий, переходимо на мітку n1

mov a,#1h ;переслати 1h у акумулятор

mov @10h,a ;вміст акумулятора - у байт 10h

cpl 16.7 ;змінюємо знак множеного на протолежний

n1: jnb 0.7, n2 ;перевіряємо знак множника, якщо він нульовий, переходимо на мітку n2

mov a,#1h ;переслати 1h у акумулятор

mov @11h,a ;вміст акумулятора - у байт 11h

cpl 0.7 ;змінюємо знак множника на протолежний

n2: clr c ;обнуляємо біт переносу

mov a,23h ;завантажуємо в акумулятор молодший байт множеного

mov b,21h ;завантажуємо в регістр В молодший байт множника

mul ab ;множення вмісту акумулятора на вміст регістра В(результат - у В,А)

mov r5,a ;молодші розряди результату пересилаються у молодші розряди добутку

mov a,b ;пересилаємо старші розряди результату у акумулятор

mov r4,a ;старші розряди результату пересилаються у 2-ий байт добутку

mov a,22h ;завантажуємо в акумулятор старший байт множеного

mov b,21h ;завантажуємо в регістр В молодший байт множника

mul ab ;множення вмісту акумулятора на вміст регістра В(результат - у В,А)

addc a,r4 ;сумуємо молодші розряди результату із значенням у 2-му байті добутку

mov r4,a ;отримана сума пересилається у 2-ий байт добутку

mov a,b ;пересилаємо старші розряди результату у акумулятор

mov r3,a ;старші розряди результату пересилаються у 3-ий байт добутку

mov a,23h ;завантажуємо в акумулятор молодший байт множеного

mov b,20h ;завантажуємо в регістр В старший байт множника

mul ab ;множення вмісту акумулятора на вміст регістра В(результат - у В,А)

addc a,r4 ;сумуємо молодші розряди результату із значенням у 2-му байті добутку

mov r4,a ;отримана сума пересилається у 2-ий байт добутку

mov a,b ;пересилаємо старші розряди результату у акумулятор

addc a,r3 ;сумуємо старші розряди результату із значенням у 3-му байті добутку

mov r3,a ;старші розряди результату пересилаються у 3-ий байт добутку

mov a,22h ;завантажуємо в акумулятор молодший байт множеного

mov b,20h ;завантажуємо в регістр В старший байт множник

mul ab ;множення вмісту акумулятора на вміст регістра В(результат - у В,А)

addc a,r3 ;сумуємо молодші розряди результату із значенням у 3-му байті добутку

mov r3,a ;отримана сума пересилається у 3-ий байт добутк

mov a,b ;пересилаємо старші розряди результату у акумулятор

mov r2,a ;старші розряди результату пересилаються у 4-ий байт добутку

mov a,@10h ;завантажуємо з байту 10h мітку знаку множеного в акумулятор

xrl a,@11h ;сума по модулю 2 вмісту акумулятора та мітки знаку множника

jz l12 ;якщо результат - 0, перейти на мітку l12

orl @2h,#80h ;якщо результат - 1, знаковий розряд добутку встановлюється в 1

l12: RET ;повернення з підпрограми

;операція X-Y=Z

SUB:mov r2,#0h ;очистка

mov r3,#0h ;байтів

mov r4,#0h ;результату

mov r5,#0h ;r2,r3,r4,r5

clr c ;обнуляємо біт переносу

jnb 32.7, n3 ;перевіряємо знак зменшуваного, якщо він нульовий, переходимо на мітку n3

mov a,#1h ;переслати 1h у акумулятор

mov @10h,a ;вміст акумулятора - у байт 10h

cpl 32.7 ;змінюємо знак зменшуваного на протилежний

mov r0,#4h ;в r0 записуємо кількість батів зменшуваного

mov r1,#27h ;в r1 записуємо адресу молодшого байту зменшуваного

clr c ;встановлюємо біт с

cpl c ;в 1

ll1: mov a,@r1 ;в акумулятор-черговий байт зменшуваного

cpl a ;інвертуємо чеговий байт

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*35*

*ІАЛЦ.463626.004 ПЗ*

addc a,#0h ;враховуємо попередній перенос

mov @r1,a ;записуємо байт, переведений у ДК назад

dec r1 ;знаходимо адресу наступного байту зменшуваного

djnz r0,ll1 ;виконуємо цикл для усіх байтів зменшуваного

n3: jnb 0.7, n4 ;перевіряємо знак від"ємника, якщо він нульовий, переходимо на мітку n4

mov a,#1h ;переслати 1h у акумулятор

mov @11h,a ;вміст акумулятора - у байт 11h

cpl 0.7 ;змінюємо знак від"ємника на протилежний

mov r0,#4h ;в r0 записуємо кількість батів від"ємника

mov r1,#23h ;в r1 записуємо адресу молодшого байту від"ємника

clr c ;встановлюємо біт с

cpl c ;в 1

ll2: mov a,@r1 ;в акумулятор-черговий байт від"ємника

cpl a ;інвертуємо чеговий байт

addc a,#0h ;враховуємо попередній перенос

mov @r1,a ;записуємо байт, переведений у ДК назад

dec r1 ;знаходимо адресу наступного байту від"ємника

djnz r0,ll2 ;виконуємо цикл для усіх байтів від"ємника

n4: clr c ;обнуляємо біт переносу

mov dpl,#4h ;в dpl записуємо кількість байтів операндів

mov r0,#23h ;в r0 записуємо адресу молодшого байту від"ємника

mov r1,#27h ;в r1 записуємо адресу молодшого байту зменшуваного

ll3: mov a,@r1 ;в акумулятор записуємо черговий байт зменшуваного

subb a,@r0 ;віднімаємо з врахуванням позики від вмісту акумулятора черговий байт від"ємника

mov @r0,a ;результат записуємо на місце від"ємника

dec r0 ;розраховуємо адресу наступного байта від"ємника

dec r1 ;розраховуємо адресу наступного байта зменшуваного

djnz dpl,ll3 ;викрнуємо цикл для всіх байтів операндів

RET ;повернення з підпрограми

;операція X+Y=Z

ADD:mov r2,#0h ;очистка

mov r3,#0h ;байтів

mov r4,#0h ;результату

mov r5,#0h ;r2,r3,r4,r5

clr c ;обнуляємо біт переносу

jnb 32.7,n5 ;перевіряємо знак доданку1, якщо він нульовий, переходимо на мітку n5

mov a,#1h ;переслати 1h у акумулятор

mov @10h,a ;вміст акумулятора - у байт 10h

cpl 32.7 ;змінюємо знак доданку1 на протилежний

mov r0,#4h ;в r0 записуємо кількість батів доданку1

mov r1,#27h ;в r1 записуємо адресу молодшого байту доданку1

clr c ;встановлюємо біт с

cpl c ;в 1

ll4: mov a,@r1 ;в акумулятор-черговий байт доданку1

cpl a ;інвертуємо чеговий байт

addc a,#0h ;враховуємо попередній перенос

mov @r1,a ;записуємо байт, переведений у ДК назад

dec r1 ;знаходимо адресу наступного байту доданку1

djnz r0,ll4 ;виконуємо цикл для усіх байтів доданку1

n5: jnb 0.7, n6 ;перевіряємо знак доданку2, якщо він нульовий, переходимо на мітку n6

mov a,#1h ;переслати 1h у акумулятор

mov @11h,a ;вміст акумулятора - у байт 11h

cpl 0.7 ;змінюємо знак доданку2 на протилежний

mov r0,#4h ;в r0 записуємо кількість батів доданку2

mov r1,#23h ;в r1 записуємо адресу молодшого байту доданку2

clr c ;встановлюємо біт с

cpl c ;в 1

ll5: mov a,@r1 ;в акумулятор-черговий байт доданку2

cpl a ;інвертуємо чеговий байт

addc a,#0h ;враховуємо попередній перенос

mov @r1,a ;записуємо байт, переведений у ДК назад

dec r1 ;знаходимо адресу наступного байту доданку2

djnz r0,ll5 ;виконуємо цикл для усіх байтів доданку2

n6: clr c ;обнуляємо біт переносу

mov dpl,#4h ;в dpl записуємо кількість байтів операндів

mov r0,#23h ;в r0 записуємо адресу молодшого байту доданку2

mov r1,#27h ;в r1 записуємо адресу молодшого байту доданку1

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*36*

*ІАЛЦ.463626.004 ПЗ*

ll6: mov a,@r1 ;в акумулятор записуємо черговий байт доданку1

addc a,@r0 ;додаємо з врахуванням переносу до вмісту акумулятора черговий байт доданку2

mov @r0,a ;результат записуємо на місце доданку2

dec r0 ;розраховуємо адресу наступного байта доданку2

dec r1 ;розраховуємо адресу наступного байта доданку1

djnz dpl,ll6 ;викрнуємо цикл для всіх байтів операндів

RET ;повернення з підпрограми

start:mov r0,#30h ;записуємо адресу масиву операндів

mov r1,#0Ch ;кількість операндів

pr1: mov a,P0 ;зчитуємо значення з порту Р0 в акумулятор

mov @r0,a ;значення з акумулятора-в масив операндів

inc r0 ;знаходимо наступну адресу

djnz r1,pr1 ;повторюємо цикл, поки не запишемо усіі необхідні операнди

mov 20h,#0h ;X2

mov 21h,#0h

mov 22h,32h

mov 23h,33h

mov 20h,22h ;розповсюджуємо байт зі знаком

anl 22h,#7Fh ;видаляємо знак всередині операнду

anl 20h,#80h ;виділяємо знак

mov 24h,#0h ;X1

mov 25h,#0h

mov 26h,30h

mov 27h,31h

mov 24h,26h ;розповсюджуємо байт зі знаком

anl 26h,#7Fh ;видаляємо знак всередині операнду

anl 24h,#80h ;виділяємо знак

acall SUB ;викликаємо підпрограму віднімання

mov 28h,20h ;(X1-X2)

mov 29h,21h

mov 2Ah,32h

mov 2Bh,33h

mov 20h,34h ;X5

mov 21h,35h

mov 22h,34h

mov 23h,35h

acall MUL ;викликаємо підпрограму віднімання

mov 20h,2h ;X5^2

mov 21h,3h

mov 22h,4h

mov 23h,5h

mov 24h,#0h ;X6

mov 25h,#0h

mov 26h,36h

mov 27h,37h

mov 24h,26h ;розповсюджуємо байт зі знаком

anl 26h,#7Fh ;видаляємо знак всередині операнду

anl 24h,#80 ;виділяємо знак

acall ADD ;викликаємо підпрограму додавання:(X5^2+X6)

mov 24h,28h ;(X1-X2)

mov 25h,29h

mov 26h,2Ah

mov 27h,2Bh

acall ADD ;викликаємо підпрограму додавання

mov 24h,20h ;(X1-X2)+(X5^2+X6)

mov 25h,21h

mov 26h,22h

mov 27h,23h

mov 20h,38h ;X9

mov 21h,39h

mov 22h,38h

mov 23h,39h

acall MUL ;викликаємо підпрограму множення

mov 20h,2h ;X9^2

mov 21h,3h

mov 22h,4h

mov 23h,5h

acall ADD ;викликаємо підпрограму додавання

mov 24h,20h ;(X1-X2)+(X5^2+X6)+X9^2

mov 25h,21h

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*37*

*ІАЛЦ.463626.004 ПЗ*

mov 26h,22h

mov 27h,23h

mov 20h,3Ah ;X10

mov 21h,3Bh

mov 22h,3Ah

mov 23h,3Bh

acall MUL ;викликаємо підпрограму множення

mov 20h,2h ;X10^2

mov 21h,3h

mov 22h,4h

mov 23h,5h

acall ADD ;(X1-X2)+(X5^2+X6)+X9^2+ X10^2

mov r0,#4h ;кількість байтів у результаті

mov r1,#20h ;адреса першого байту

pr2: mov P0,@r1 ;пересилаємо байт результату на порт Р0

inc r1 ;знаходимо наступну адресу

djnz r0,pr2 ;повторюємо цикл, поки не передамо результат

***Додаток B***

*Зм.*

*Арк.*

*№ докум.*

*Підп.*

*Дата*

*Арк.*

*38*

*ІАЛЦ.463626.004 ПЗ*

*Список скорочень.*

*МК - мікроконтролер*

*МПС - мікропроцесорна система*

*П - процесор*

*ОП - оперативна пам’ять*

*ПЗП - постійній запам’ятовуючий пристрій*

*ОЗП - оперативний запам’ятовуючий пристрій*

*ЗП - зовнішній пристрій*

*ПВВ - порт вводу-виводу*

*ШІМ - широтно - імпульсна модуляція*

*КПП - контролер пріоритетних переривань*

*КПДП - контролер прямого доступу до пам’яті*

*ПП - пам’ять програм*

*ПД - пам’ять даних*