МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

Кировоградский национальный технический университет

Кафедра ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Профессор Сидоренко В.В.

**архитектура компьютера**

конспект лекций

Кировоград

**ÇÌ²ÑÒ**

*Тема 1.* Системный и реальное время компьютера IBM PC 2

*Тема 2.* Часы реального времени (RTC) 14

*Тема 3.* Таймер операционной системы (таймер BIOS) 21

*Тема 4.* Средства Windows для работы с таймером ОС 24

*Тема 5.* контроллер прерываний 26

*Тема 6.* И / О информации компьютера. программируемый порт

последовательной передачи данных 34

*Тема 7.* КПДП (DMA) в компьютерах IBM PC 45

литература 55

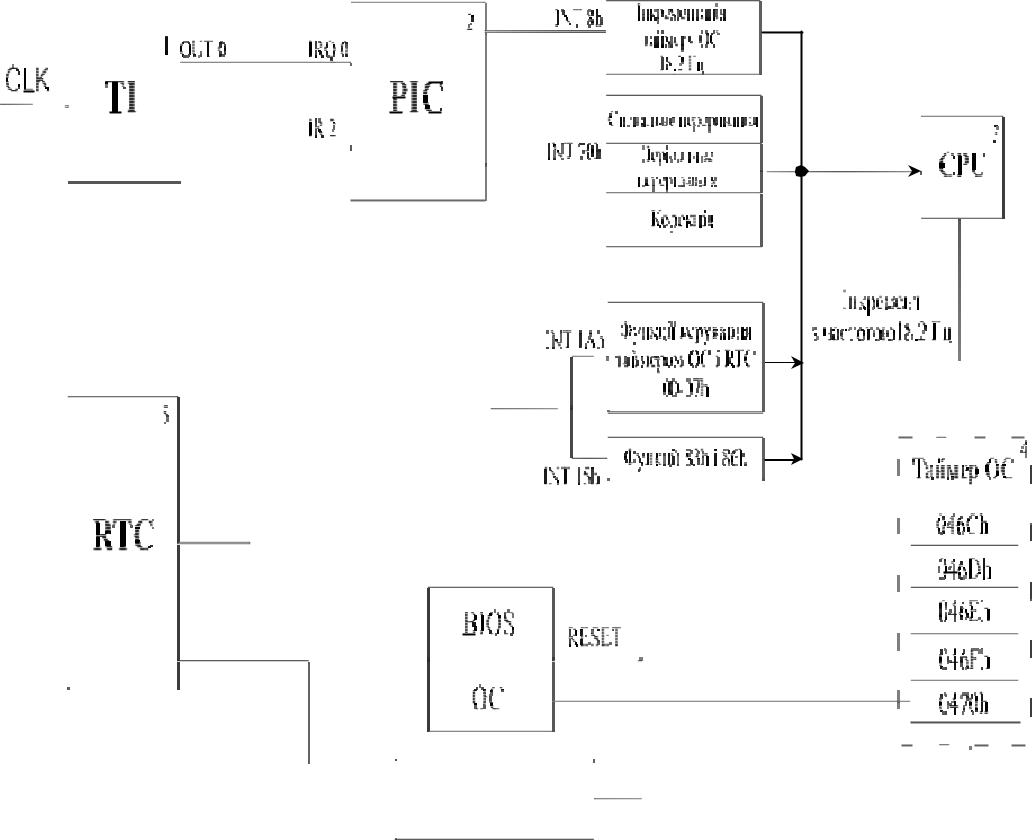
56 1

*Тема 1*

**Системный и реальное время компьютера IBM PC**

Все современные комп периферия содержат часы (таймеры) системного и реального времени.

Системное время образуется (рис.1) Операционной системой (ОС) комп периферия с помощью интервального таймера (1), ячеек памяти (046ch ... 0470h) (4), часов реального времени (5) -BTK, PI CPU (3 ).



Загрузка



таймера ОС и



коррекция 

*Рис. 1 - Функциональная схема с объединений устройств для образования системного*

*и реального времени в IBM РС*

При включении компьютера системное время устанавливается на основе данных

часов реальног времени (5) и в дальнейшем работать независимо от RTK. при

отключении ПК данные часов системы времени теряются.

Большая часть прикладных и системных программ используют это время для разв Связывание системных задач. Например, ОС записывает на диск дату и время, создание файлов, устанавливает различного рода паузы во время работы, и тому подобное.

Приложения могут периодически выполнять команды, которые зависят от времени. Например, хранить документ на диске через каждые 2 минуты. Кроме того, различные системы управления используют системное время для своих нужд (для определения сроков различных операций, планирование движения транспорта и т.д.). Реальное время компьютера образуется с помощью микросхемы (контроллера MС 46818, которая питается от аккумулятора позволяет контролировать время независимо от включения / выключения PC. RTC в своем составе содержит ячейки памяти, которые позволяют создавать календарь часы, будильник, который может выдавать сигнал в заданное время, а также хранить информацию о составе и настройки компьютера. Функциональная схема с объединений устройств, а также

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| прерываний, которые образуют системный и | реальный | время | приведена | на | рис.1 |
| Рассмотрим работу отдельных устройств приведенной схемы. Таймер на плате IBM PC | | | | | |
| используется для системного времени | выход | OUT0 | (канал | 0), | который |

запрограммирован на генерацию прямоугольных сигналов с частотой 18,2 Гц. Выход канала 0 соединен с входом IR0 PIC. Таким образом генерируется аппаратное прерывание IRQ0 (int8h). Таким образом, контроллер прерываний выдает адрес программы обработки этого прерывания int8h. Выполняя этот обработчик CPU (3) инкрементирует ячейки таймера ОС (4): 046ch-046fh с частотой 18,2 Гц, а при достижении количества тиков, которые соответствуют эпохе, (1572480) в ячейке памяти вашего компьютера по адресу 0470h с появляется 1 , которая является сигналом для ОС о вызове прерываний int1Аh, которое осуществляет сброс в 0 значений ячеек памяти вашего компьютера 046сh-0470h и коррекцию календаря. Таким образом происходит переход системного времени на следующие сутки. Программа обработки прерывания in1Аh осуществляет управление таймером ОС, а также установление и считывания данных с RTC.

Для получения временных задержек и интервалов используют функции 83h и 86h программного прерывания int15h. Выход RTC с Объединенные со входом IR2 PIC и аппаратное прерыванием IRQ2, которому соответствует программное прерывание int70h.

2 3

Это программное прерывание GAT адреса программ обработки сигнального прерывания (будильник), периодического прерывания, а также коррекции.

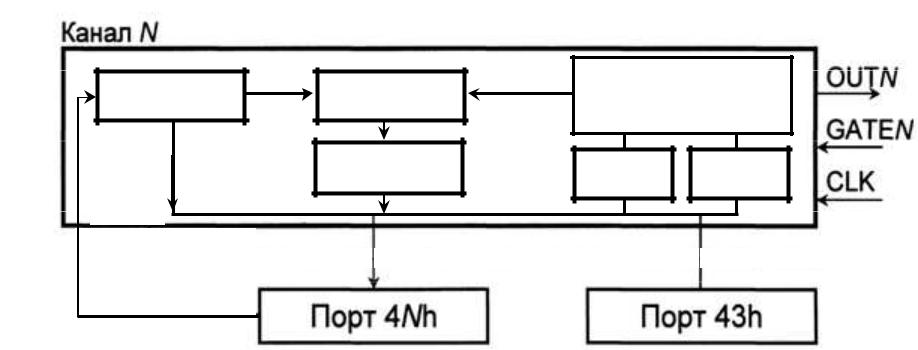
Рассмотрим каждое из устройств с помощью которых реализован системный и реальное время в IBM PC.

**интервальный таймер**

0-выдача сигнала по окончании счета;

1-формирования времени запрограммированной длины;

2-генератор импульса определенной длительности;



1. *= Биты D7D6 управляющего байта = 0, 1, 2*

|  |  |  |  |
| --- | --- | --- | --- |
| CRN | CEN | схема управления |  |
| каналом *N* |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| OLN | RTSN | WCRN |

*Рис. 2 - Блок-схема канала интервального таймера*

Интервальный таймер на плате IBM PC, который используется на системной плате выход out0, который подается на вход IR0 PIC (аппаратное прерывание IRQ0) PIC по входу IR0 выдает вектор прерываний int8h. По этому прерыванию CPU осуществляет инкрементация ячеек памяти вашего компьютера с помощью 046сh-046fh. При достижении заданного числа в трьохбайтних ячейках в ячейке 0470h устанавливается 1, которая служит сигналом для ОС включения программы корректировки часов таймера ОС, в том числе суток и сброс в 0 значение ячеек памяти вашего компьютера 046h-0470h.

Начальное значение времени (часов таймера ОС) при включении компьютера устанавливается с помощью часов реального времени RTK. В

ответ PIC выдает программное прерывание in70h, благодаря которому устанавливается начальное значение времени в ячейках 046сh-0468h.

Таймер на плате IBM PC содержит три независимых канала. Каждый канал можно представить в виде схемы на рис. 2.

Каждый канал таймера содержит регистры:

-16-разрядный регистр счетчика (СЕ)

-16-разрядный регистр загрузки константы пересчета (CR)

-16-разрядный буферный регистр OL;

-8-разрядный регистр управляющего слова (WCR)

-8-разрядный регистр состояния канала (RST)

Каналы таймера подключаются к внешним устройствам:

*GATE* - управляющий вход;

*CLK* - вход тактовой частоты;

*OUT* - выход канала.

Регистр СЕ работает в режиме вычитания и его содержание уменьшается по заднему фронту сигнала CLK, при условии, что GATE = 1.

Буферный регистр OL - предназначен для запоминания содержания регистра СЭ для считывания константы пересчета без остановки таймера.

Регистр CR - загружается программистом и его содержание перезагружается в регистр СЕ по сигналу GATE, в зависимости от режима.

Шесть режимов работы таймера с точки зрения программирования объединяются в три группы:

* **группа**: Однократное выполнение функций (режимы 0 и 4);
* **группа**: Работа с перезапуском (перезагрузкой) (режими1 и 5);
* **группа**: Работа с автоперезавантаженням (режимы 2 и 3);

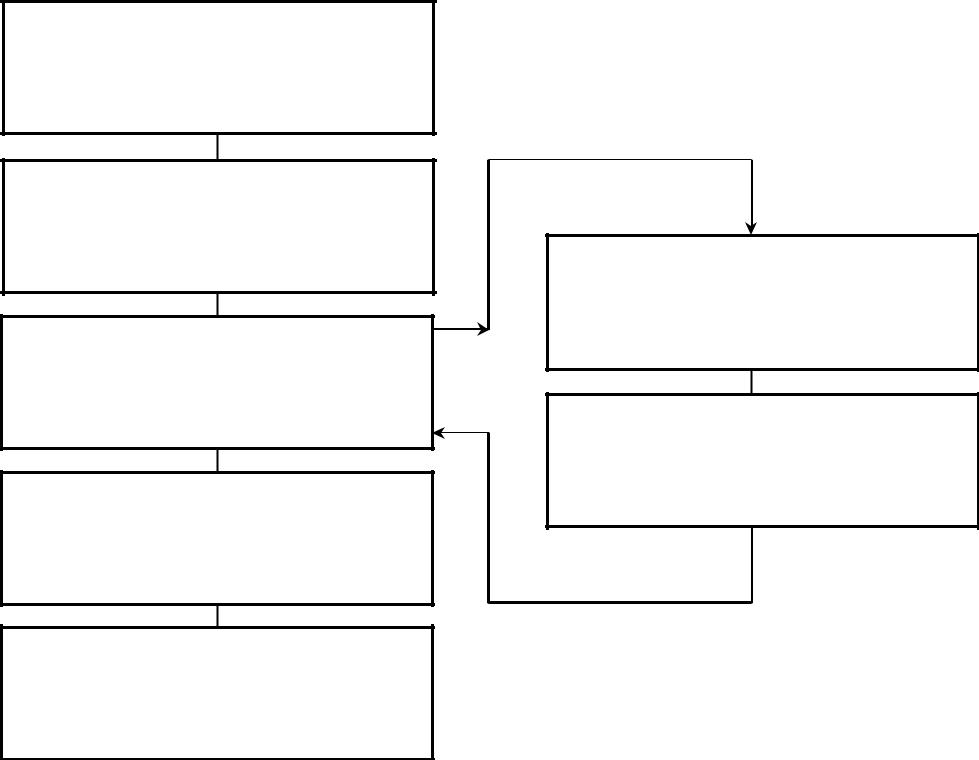
В однократном режиме перед началом счета содержание регистра CR переписывается в СЕ и константа в СH не хранит начальное значение. Для повтор ного выполнения функции нужно новое загрузки CR (перепрограммирования)

В режиме с пере запуском не требуется повторное перезагрузки таймера. По сигналу GATE значение константы с CRпереписуеться в СЕ и константа

4 5

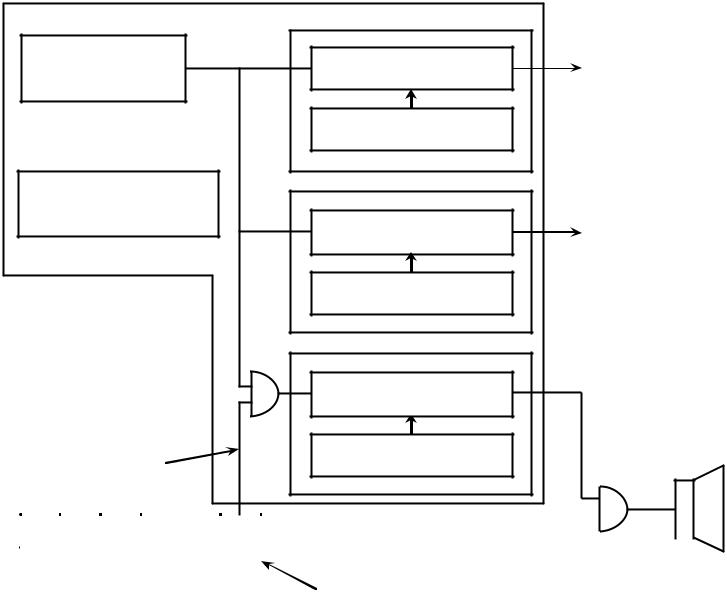
остается также в регистре CR (в режиме автозагрузки значение регистра CR автоматически переписывается в регистр СЕ по окончании счета.

**Блок-схема прерывания int 8h**



В IBM PC используют все 3 канала таймера (рис. 3).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | интервальный таймер | |  |  |  |
|  | Генератор |  | счетчик | Канал 0 - обеспечение |  |
|  | 1,193 МГц |  |  |
|  |  |  | работы системного |  |
|  |  |  |  |  |
|  |  | 40h | Константа | часов (INT 8h) |  |
|  |  |  |  |
|  | управляющий |  |  |  |  |
|  | регистр |  | счетчик | Канал 1 - регенерация |  |
|  | 43h |  |  |
|  |  |  | динамической памяти |  |
|  |  | 41h | Константа |  |  |
|  |  |  | счетчик | Канал 2 - звук |  |
|  |  |  |  | динамика; предоставленный |  |
| 1 - | канал включен | 42h | Константа | для пользовательских |  |
| программ |  |
| 0 - | канал выключен |  |
|  |  |  |
|  |  |  |  |



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 |  | 4 | 3 | 2 | 1 | | 0 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | динамик | | |  |
|  |  |  | порт 61h | | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  | 1 - |  |  |  |
|  |  |  |  |  |  |  |  |  |  | динамик включен | | | | |  |
|  |  |  |  |  |  |  |  |  |  | 0 - | динамик выключен | | | | |  |

инкремент счетчика

по адресу 0000: 046Ch,

проверка его на переполнение

Проверка времени последнего

обращение к НГМД,

если оно более 2 сек. -

выключения двигателя НГМД

INT 1Ch

Сброс контроллера прерываний

IRET

Выполнение действий,

определенных в программе

IRET

*Рис. 3 - Структурная схема назначения каналов таймера*

*на плате IBM РС*

***Канал 0*** - используют в системных часах времени (таймер ОС). Работает в режиме 3 и используется как генератор импульса 18,202 Гц, который вызывает аппаратную перервуIRQ0 (int8h)

***Канал 1*** - используется для регенерации содержания физической памяти

компьютера.

***Канал 2*** - отдан в распоряжение программиста. Можно воспроизводить генерацию розни х звуков, написание фоновых программ, или использовать как генератор случайных чисел.

**Программирование интервального таймера**

При включении компьютера выход интервального таймера устанавливается в логическую единицу, а номер режима, значение и константа пересчете определяется программистом.

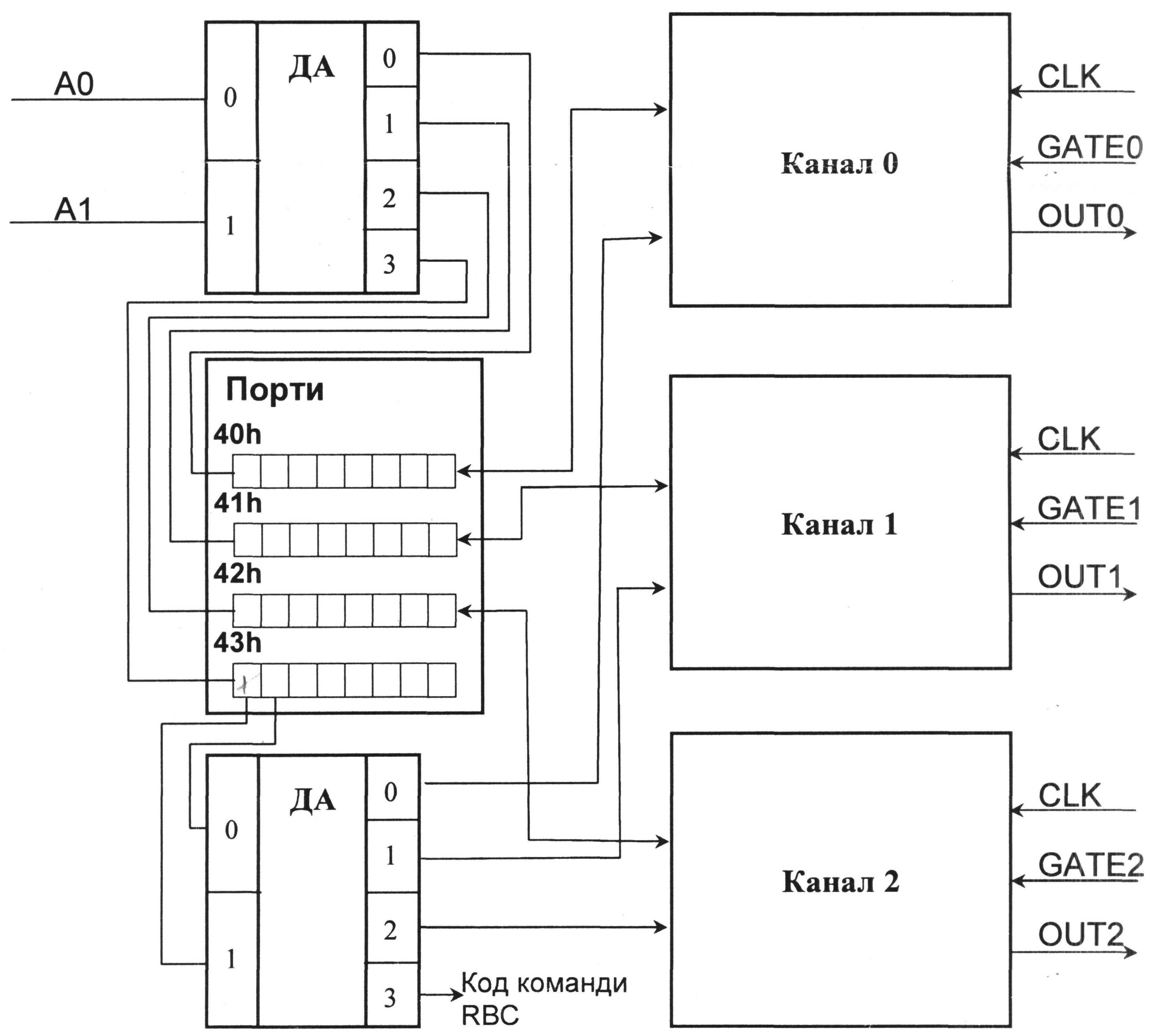
***Программирование таймера следует разделять на 2а направления:***

1. Инициализация канала 0, 1 или 2 интервального таймера - запись управляющего слова и константы пересчета. Этого достаточно, чтобы таймер начал работу;
2. Чтение байта состояния канала и константы пересчета каналов 0, 1 или 2.

6 7

Для программирования таймера используются порты 40h, 41h, 42h, 43h. 40h, 41h, 42h - предназначены для загрузки константы пересчета в соответствующий канал таймера. Порт 43h - предназначен для загрузки управляющего байта, причем номер канула, для которого предназначен байт, определяется D6, D7 управляющего байта.

**Функциональная схема выбора канала таймера для загрузки управляющего байта и константы пересчета (рис. 4)**



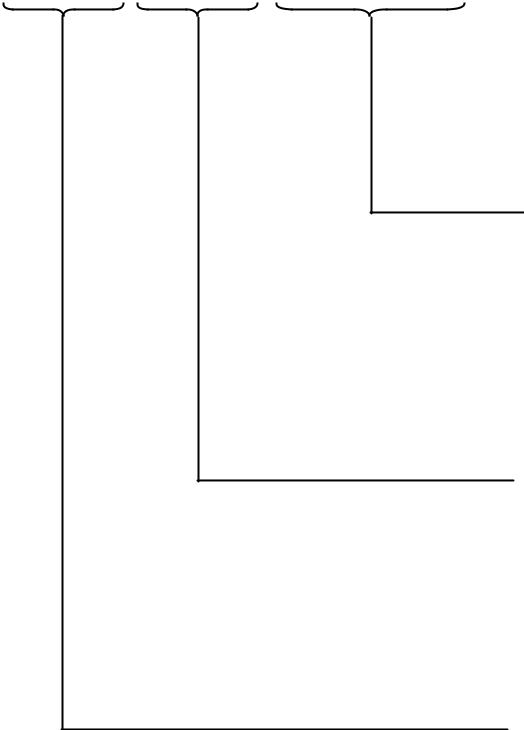
*Рис. 4*

Комбинация D6 = 1 и D7 = 1 в этом направлении запрещена.

**Формат управляющего байта**

D7D6D5D4D3D2D1 D0

 отсчет:



0 - двоичный;

1 - BCD.

режим:

000 - режим 0;

001 - режим 1;

010 - режим 2;

011 - режим 3;

100 - режим 4,

101 - режим 5,

Вид загрузки счетчика:

00 - чтение "на лету";

01 - только младший байт;

10 - только старший байт;

11 - младший, затем старший

б.

Номер канала:

00 - канал 0;

01 - канал 1;

10 - канал 2,

11 - запрещенная комбинация.

**Для программирования таймера следует выполнять 2 правила:**

1. В каждый счетный управляющего слова должно быть записано перед загрузкой счетчика
2. Каждый счетчик должен загружать такое количество байтов и в таком порядке, к указано в управляющем байте

8 9

Рассмотрим программирования второго направления. Оно заключается в том, что программисту необходимо проверить состояние таймера и константу пересчете таймера. Эта задача программно выполняется двумя способами:

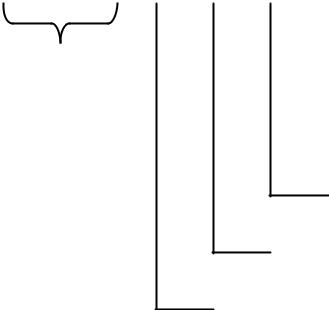
* остановкой таймера;
* без остановки таймера.

*Рассмотрим 1-й путь:* Чтение состояния тайме5ра и константы пересчетасоответствующих каналов. Для этого необходимо посылать управляющий байт RBC, который образуется с помощью битов D6 = 1 и D7 = 1 порта 43h.

**Формат команды RBC (Read Back Command)**

D7D6D5D4D3D2 D1 D0

1 1



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| код RBC |  |  |  | BCD |  |
|  |  |  |  |
|  |  |  |  |
|  |  |

K0

K1

K2

0- чтения RST (состояние таймера)

1- не чтение RST

0-чтения константы;

1-не чтение



D4 = 0

D5 = 0 - зоборонена комбинация, код чтения «на лету»

*Рассмотрим 2-й путь:* Чтение константы пересчета с таймера без егоостановки (чтение «на лету»): для этого программиста в порт 43h посылает команду следующего формата:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D7 | D6 | D5 | D4 | D3 | | D2 D1 | | D0 | |
|  |  |  |  |  |  |  |  |  |  |
|  |  | 0 | 0 | х |  | х | х | 0 |  |
|  |  |  |  |  |  | |  |  |  |
|  | номер | | код команды | | | | |  |  |
|  | канала | |  | чтения «на лету» | | | | | |



00-k0

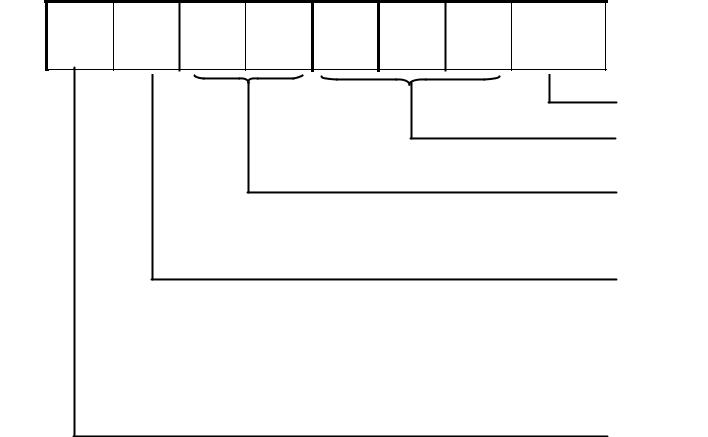
01-k1

10-k2

11-запрещено

Эта команда позволяет прочитать обратное значение счетчика. Она не меняет содержание регистра СЕ. При выполнении этой команды текущее значение регистра СЭ фиксируется в OL и находится там до команды считывания этого счетчика.

**Формат байта состояния интервального таймера**



D7D6D5D4D3D2D1 D0 (0)

BCD-0

номер режима

Чтение и запись const

пересчете

Флаг загрузки const

0-загружена

1- не загружена

выход OUT

0-на выходе 0

1- на выходе 1

10 11

***Особенности интервального таймера:***

1. Если счетчик программируется на 2 байта чтения, то 2 байта должны быть прочитаны;
2. Чтение может прерываться записью в тот же счетчик. К примеру:

* чтения младшего байта;
* записать новый младший байт;
* чтения старшего байта;
* записать новый старший байт;

Для чтения состояния таймера и константы пересчета в зависимости от способа выполнения следующего действия:

***способ 1***:чтенияRSTи константы пересчета с остановкой таймера;

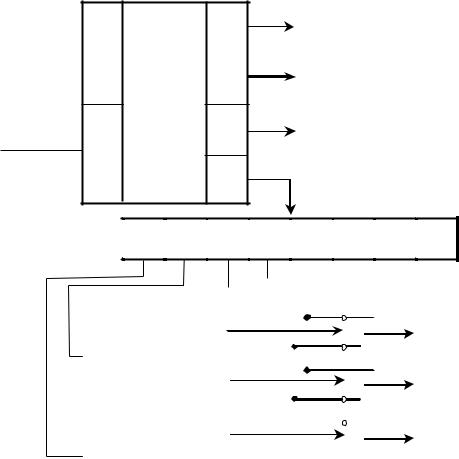
***способ 2***:чтенияRSTи константы пересчета без остановки таймера.

**Способ 1 - Чтение RSТ и константы пересчета с остановкой таймера**

Посылать управляющее слово в порт 43h, в котором D7 и D6 = 1 и в зависимости от того читать регистр состояния D4 = 0. Номер нужного канала указывается в битах D1, D2, D3. Затем необходимо командами чтения прочитать из соответствующего порта (40h, 41h, 42h) const пересчета. Следует обратить внимание, что если в управляющем слове указано загрузки младшего и старшего байта, то и читать const следует столько же раз. D4, D5 = 0 - запрещено.

**Способ 2 - Чтение константы пересчета без остановки таймера**

* регистр 43h необходимо записать управляющий байт, в котором D5 и D4 = 0.
* этом случае номер канала, в котором следует прочитать const пересчете указанную в битах D6, D7 (D6 = 1, D7 = 1 - запрещенная комбинация).



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| А0 = 1 | | **ДА** 0 | |  |
|  | 1 |  |
|  |  |  |

А1 = 1 2

1. Управляющее слово (порт 43h)

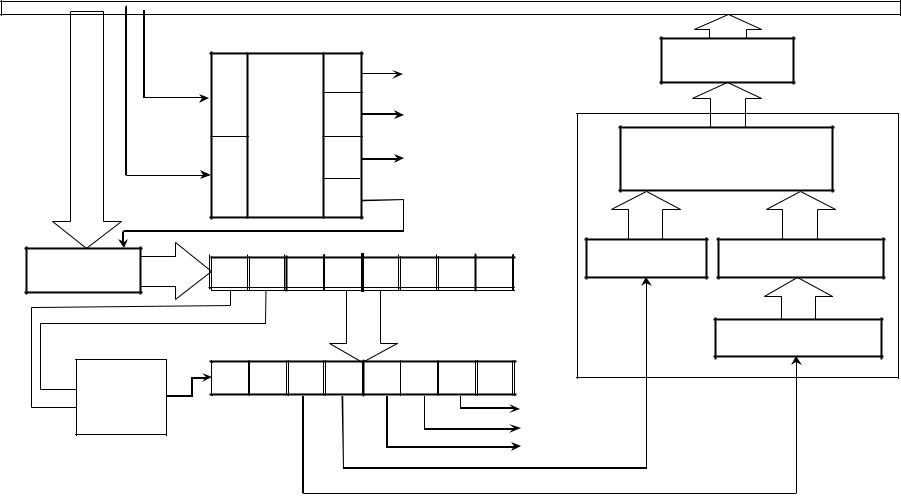
**43h** D7  D6  **0**  **0**  X  X  X  X

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **ДА** | | 0 | |  |  |  |  | & |  |  | CE0 |  |  | OL0 | |  | порт 40h |  |
|  |  |  |  | 1 | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | & |  |  | CE1 |  |  | OL1 | |  | порт 41h |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 2 | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | & |  |  | CE2 |  |  | OL2 | |  | порт 42h |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  | |  |  | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Х | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | |  |  | | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | при D5= 0, D4= 0 | | | | | | | | (Чтение "на лету") | | | | | |  |  |  |  |  |
|  |  | D7= 1, D6= 1 - запрещенная комбинация | | | | | | | | | | | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |



Приведем пример программы, которая перепрограммирует канал 2 интервального таймера на режим работы 0 с константой пересчете 3000.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **системная шина** |  |  |  |  |  |  |  |  |  |  |
|  |  | **ДА** | | 0 |  |  |  |  | порт 4Nh |  |
| А0 = 1 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | канал N |  |  |
|  |  |  |  | 1 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | 2 |  |  |  | буфер данных | |  |
| А1 = 1 |  |  |  |  |  |  |  | D7..D0 |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  | 3 |  | CWR | |  |  |  |
|  |  |  |  |  |  |  |  |  |
| порт 43h |  |  |  |  |  |  |  | RST N | CE N |  |
| 1 | 1 | D5 | D4 | D3 | D2 | D1 | 0 |  |  |
|  | **RBC** | |  |  |  |  |  |  | CR N |  |
|  |  |  |  |  |  |  |  |  |
| & | 1 января D5 | | | D4 | D3 | D2 | D1 | 0 |  |  |
|  |  |  |  |  |  |  | 1 - N = 0 |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | 1 - N = 1 |  |  |
|  |  |  |  |  |  |  |  | 1 - N = 2 |  |  |
|  |  |  |  |  | 0 - читать RTS N | | | |  |  |
|  |  |  |  |  | 0 - читать СЭ N | | | |  |  |



#include <stdio.h>

#include <dos.h>

main (void)

{

// записываем управляющий байт в соответствии с форматом

outportb (0x43,0xB0)

outportb (0x40,0xB8)

outportb (0x40,0x0B)

printf ( "Канал 2 перепрограммирован. \ n");

}

12 13

*Тема 2*

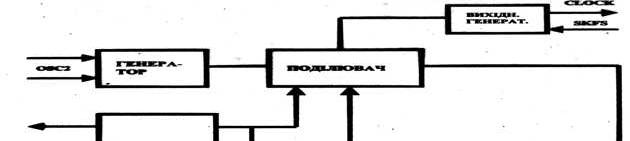
**Часы реального времени (RTC)**

Кроме интервального таймера в состав устройств, которые образуют реальный и системное время входит также часы реального времени RTC. Он необходим для того, чтобы сохранить дату и время, когда компьютер находится в выключенном состоянии. RTC энергонезависимый и питается от специального аккумулятора, по состоянию которого следит ОС.

RTC содержит календарь, а также ряд специальных ячеек памяти (CMOS), которые способны хранить различные параметры системы при выключенном компьютере. При включении компьютера и загрузке ОС, последняя устанавливает свой собственный часы (таймер ОС), используя данные с RTC.

Для построения RTC используется контроллер МС 46818, который используется в качестве RTC и памяти вашего компьютера конфигурации системы. Контроллер содержит в себе 64 байта памяти вашего компьютера CMOS, которые cкладаються из 10 байтов данных часов и календаря, 4 байта - регистры A, B, C, D (для управления работой RTC), а также 50 байтов, которые используются для размещения информации о конфигурацию системы.

*2009*



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OSC1** |  |  |  |  |  |
|  |  |  |  |  |  |
| **OSC2** |  | **ГЕНЕРАТОР** |  | **разделитель** |  |
|  |  |  |  |  |
|  |  |  |  |  |  |

**SQW** **ВЫХОДНОЙ**

**ГЕНЕРАТОР**



**IRQ**

**RESET**

**PS**

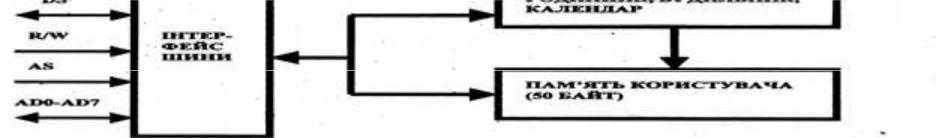


Таблица 1 - Адресное пространство памяти вашего компьютера КМОП-микросхемы

|  |  |  |  |
| --- | --- | --- | --- |
| Адрес | количество | назначение |  |
| поля | байт |  |
|  |  |
| 00h | 1 | Секунды в BCD |  |
|  |  |  |  |
| 01h | 1 | Секунды будильника в BCD |  |
| 02h | 1 | Минуты в BCD |  |
|  |  |  |  |
| 03h | 1 | Минуты будильника в BCD |  |
| 04h | 1 | Часы в BCD |  |
|  |  |  |  |
| 05h | 1 | Часы будильника в BCD |  |
| 06h | 1 | День недели (может отсутствовать) |  |
|  |  |  |  |
| 07h | 1 | Число в BCD |  |
|  |  |  |  |
| 08h | 1 | Месяц (январь - 1, февраль - 2 и т.д.) в BCD |  |
|  |  |  |  |
| 09h | 1 | Год моложе две цифры в BCD |  |
|  |  |  |  |
| 0Ah | 1 | **регистр A** |  |
|  |  |  |  |
| 0Bh | 1 | **регистр В** |  |
|  |  |  |  |
| 0Ch | 1 | **регистр С** |  |
| 0Dh | 1 | **регистр D** |  |
|  |  |  |  |
| 0Eh | 1 | байт диагностирования |  |
| 0Fh | 1 | Байт кода сброса процессора |  |
|  |  |  |  |
| 10h | 1 | Типы HDD (если менее 15) |  |
| 11h | 1 | зарезервировано |  |
|  |  |  |  |
| 12h | 1 | типы дискет |  |
| 13h | 1 | зарезервировано |  |
|  |  |  |  |
| 14h | 1 | Состав установленного оборудования |  |
| 15h | 2 | Об Объем базовой памяти вашего компьютера, Кбайт |  |
|  |  |  |  |
| 17h | 2 | Об Объем расширенной памяти вашего компьютера, Кбайт |  |
|  |  |  |  |
| 19h, 1Ah | 2 | Тип первого HDD (если более 15) |  |
|  |  |  |  |
| 1Bh | 14 | зарезервировано |  |
|  |  |  |  |
| 2Eh | 2 | Контрольная сумма байтов 10h-2Dh |  |
|  |  |  |  |
| 30h | 2 | Об Объем расширенной памяти вашего компьютера, Кбайт |  |
|  |  |  |  |
| 32h | 1 | Год, первые две цифры в BCD |  |
| 33h | 1 | системная информация |  |
|  |  |  |  |
| 34h | 12 | зарезервировано |  |
|  |  |  |  |

14 15

Доступ к ячейкам памяти вашего компьютера CMOS осуществляется с помощью двух портов - 70h и 71h.

В порт 70h сначала заносится номер требуемой ячейки памяти вашего компьютера CMOS, к которой добавляется 80h (D7 = 1) для запрещение немаскируемых прерываний NMI. Далее для занесения нужной информации в ячейку 71h выполняется запись или чтение нужной информации.

RTC в компьютере взаимодействует вместе с CPU через PIC, формируя аппаратное прерывание IRQ8 (int70h). RTC имеет один выход, который соединен с входом PIC\_ф и обеспечивает прерывание 3х видов:

1. ***Сигнальное прерывания (будильник)*** - может быть запрограммировано от 1

раз в секунду до 1 раза в сутки;

1. ***периодическое прерывание*** - может биты с периодом 30,517мкс ... 0,5 с (для фоновых программ);
2. ***Программирование корректировки времени***. Нужный режим программируется с помощью регистра В. Каждое из 3с прерываний.

**сигнальное прерывания**

Прграмме CPU получает значение будильника и каледаря путем чтения байтов через порты 70h и 71h.

В 70h указываем адрес необходимого байта + 80h (d7 = 1) и затем из порта 70h читаем / записываем нужную информацию.

Содержание 10-и байтов 00h-09h - это часы (календарь, будильник) должен быть двоичным или двоично-десятичной.

При инициализации данных ячеек памяти надо запретить коррекцию, то есть d7 = 1 в регистре В.

Байта часов, календаря и будильника не всегда доступны программисту. 1 раз

* секунду все 10 байтов переключаются на корректировку, во время которой невозможно прочитать истинное значение байтов.

Три байта будильника могут быть использованы двумя способами:

1. Когда программа устанавливает время будильника в соответствующих байтах часов,

минут, секунд, каждый день в указанное время производится прерывания будильника;

1. Если установлен разряд разрешает прерывание D5 регистра В;
2. Если 2 старших байта установить в единицу, то пе6реривання происходить каждые 2:00.

**периодическое прерывание**

Позволяет включить вид IRQ через определенный интервал от 30 до 517 мкс 30517мкдк до 0,5 сек. Это прерывание имеет ничего общего с сигнальным.

Частота периодического прерывания определяется разрядами d0, d1, d2, d3 регистра А.

Периодическое прерывание реализуется во всех системах реального времени, а также используется для обработки входных сигналов корреляционной функции статистической характиристики и позволяет работать в фоновом режиме

**Прерывание конца коррекции**

Цикл коррекции проводится каждую секунду, если бит d7 регистра В установлен.

Во коррекции программа не имеет доступа к байтов времени, календаря и будильника. Это достигается за счет отключения части памяти вашего компьютера от системной шины.

RTC имеет 4 изолированные ячейки памяти вашего компьютера (порты) (служебные регистры A, B, C, D), которые предназначены для программиста для управления работой RTC.

***Формат регистра А***

**0Ah** - регистр состояния RTC:

7654321 0

 0-3: скорость отсчета

 4-6: делитель

 7: флаг обновления

16 17

D0, D1, D2, D3 - для выбора первого из 15 выходов разделяемая частоты.

D4, D5, D6 - 3 разряда, которые позволяют выбрать необходимую частоту CLK,

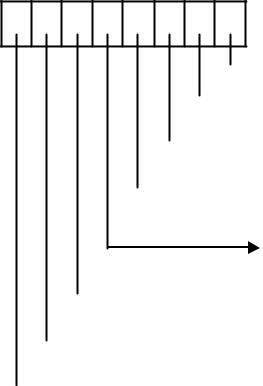
нужную для работы CPU (экономия кварцевых кристаллов)

D7 - разряд контроля текущего цикла коррекции. 1- происходит коррекция (данные часов, календаря, будильника недоступны), 0- данные доступны. D7 доступен только для чтения: не сбрасывается выводом RESEТ.

***Формат регистра В***

**0Bh** - регистр состояния RTC:

76543210



 0: тип времени

 1: 12- или 24-часовой формат времени

 2: формат данных: 0 = BCD,

1 = двоичный;

 3: 1 = разрешить прямоугольный импульс

1. 1 = разрешить прерывание от конца обновления

 5: 1 = разрешить прерывание

будильника (сигнальное прерывание)

 6: 1 = разрешить периодическое

прерывания

 7: флаг обновления

D0 - тип времени (0 - зимний, 1 - летний)

D1 - продолжительность суток (1-24 ч, 0-12 ч)

D2 - формат данных (1 - bin 0 - 2/10);

D3 - разряд доступа прямоугольного сигнала SQR;

D4 - разряд прерывания конца коррекции, доступный для чтения и записи, позволяет установить аппаратное прерывание IRQ;

D5 - разряд разрешения сигнального прерывания с частотой, заданной в регистре А, заданной битами D0, D1, D2, D3;

D6 - разряд разрешения периодического прерывания;

D7 - разряд разрешения коррекции

Регистр В доступен для записи и чтения не меняется RESEТ.

***Формат регистра С***

D7D6D5D4D3D2 D1 D0

D0, D1, D2, D3 - нули;

D4 - флажок конца коррекции

D5 - флажок сигнального прерывания (1 - совпадет сигнала и

времени);

D6 - флажок периодического прерывания

D7 - управление IRQ (1 - IRQ запрещено)

***Формат регистра D***

Используется только бит D7: 1 - CMOS получает питание от аккумулятора, 0 - от источника.

***Формат регистра Е***

**0Eh** - байт байт результатов начального тестирования.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | 0 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  | 0 | 0 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 2: 1 = формат текущего времени |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 3: 1 = HDD не загружается |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4: 1 = размер ОЗУ неправильный |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5: 1 = устройство соответствует |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | конфигурации CMOS |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 6: 1 = неправильная контрольная сумма CMOS |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | 7: 1 = не работает автономный источник питания |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | CMOS |  |



**0Fh** - байт состояния перезагрузки. Этот байт считывается после сброса ЦБ, чтобы определить, не было сброса, вызванного выводом 80286 с защищенного режима. Он может иметь следующие значения:

0 - горячий рестарт (Ctrl-Alt-Del) или неожиданная остановка;

1 - остановка после определения размера ОЗУ;

2 - остановка после тестирования памяти вашего компьютера;

3 - остановка после обнаружения ошибки паритета памяти вашего компьютера;

18 19

4 - рестарт по запросу начального загрузчика;

5 - рестарт за сбросом контроллера прерываний и JMP FAR PTR [0: 467h];

6, 7, 8 - остановка после теста защищенного режима;

9 - рестарт за JMP FAR PTR [0: 467h].

*Пример.* **Программа перехода на****2000и****год**

**Си**

#include <conio.h>

#include <dos.h>

int main (void)

{

char i, rez;

printf ( "Переход на 2000 год \ n");

i = 0x32;

outportb (0x70,0x80 + i)

outportb (0x71,0x20)

rez = inportb (0x71)

if (rez == 0x20)

printf ( "Переход выполнен успешно! \ n");

else

printf ( "Переход не выполнен! \ n");

return 0;

|  |  |
| --- | --- |
| } |  |
| **ASM 86** |  |
| MOV DX, 70h |  |
| MOV AL, B2h | ; (80h + 32h) |
| OUT DX, AL |  |

INC DX

MOV AL, 20h

OUT DX, AL

*Тема 3*

**Таймер операционной системы (таймер BIOS)**

Для ОС таймер представляет собой 32 разрядную ячейку памяти вашего компьютера, которая начинается по адресу 046ch-0470h. Эти ячейки памяти вашего компьютера инкрементуються прерыванием int8h. При достижении значения времени этих ячеек, ОС сбрасывает таймер ячеек в 0 и устанавливает флажок ячейки 0470h, которая служит для программного прерывания и коррекции таймера.

Для работы с таймером ОС содержит функции 00h-07h прерывания int1Аh. Эти функции позволяют установить начало значение таймера при включении компьютера, а также позволяет прочитать текущее содержимое таймера, а также считать и установить время и день RTC.

|  |  |
| --- | --- |
|  | ***Функции прерывания int1Аh*** |
|  |  |
| № ф-й | действие |
|  |  |
| 00h | Считывания таймера ОС |
|  |  |
| 01h | Установка таймера ОС |
| 02h | Считывания времени RTC |
|  |  |
| 03h | Установка времени RTC |
|  |  |
| 04h | Считывания даты RTC |
| 05h | Установка даты RTC |
|  |  |
| 06h | Установка сигнализации RTC |
| 07h | Отмена сигнализации RTC |
|  |  |

*Пример применения функции int1Ah*

Функция 00h - считывание таймера ОС на входе Аh = 0 ни выходе Сх - старшее слово счетчика; Dx - младшее слово счетчика; Al = 1, если с момента запуска прошло 24 часа. Al (0470h) - флажок признаки переполнения.

20 21

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Mov Ah, 00h | | | Сх - старшее слово времени работы счетчика (мксек) | |
|  | Int 15h | | | Dx - | младшее слово (мксек) |
|  | Dx ... | | | Int15h | |
|  | Cx ... | | |  |  |
|  | Al ... | | | Средства MS DOS для работы с таймером MS DOS использует 4 функции | |
|  |  |  |  | прерывания Int21h с системным таймером: | |
|  | Пользование функциями Int15h: 01h, 02h, 03h, 04h, 05h, 06h, 07h аналогично | | |  |  |
| использования функции 01h. | | | | 2ah - | определение текущей даты; |
|  | Таймер ОС позволяет также создавать функцию задержки, а также функцию | | | 2bh - | установление даты; |
| отсчета нужного интервала времени. | | | | 2ch - определение текущего времени; | |
|  | Для образования временных интервалов и функций задержек времени | | | 2dh - | установление времени. |
| используют функции 83h и 86h прерывания Int15h. | | | |  |  |
|  | ***Рассмотрим эти функции:*** Функция 83h позволяет запустить таймер ОС на счет, | | |  |  |
|  |  | |  |  |  |
| указав некоторую адрес байта в оперативной памяти. Программа, которая запустила | | | |  |  |
| таймер сразу после запуска, получит управление. По окончании заданного времени | | | |  |  |
| функция устанавливает старший бит заданного байта в 0, символизирующий завершение | | | |  |  |
| указаний временного интервала. Эту функцию удобно использовать для | | | |  |  |
| организации выполнения каких-либо действий, параллельно с отчисления времени. К примеру | | | |  |  |
| ограничить время на ввод пароля. | | | |  |  |
|  | На входе Аh в 83h: | | |  |  |
|  | 1 - | запустить таймер; | |  |  |
|  | 0 - | остановить таймер; | |  |  |
|  | Сх - старшее слово счетчика (в микросекундах), Dx - младшее слово; | | |  |  |
| Es: Bx - | | адрес байта, в котором по окончании интервала времени меняется бит D7. | |  |  |

***Формирование задержки: функция 86h***

Функция предназначена для формирования задержек. Позволяет определить время

задержки в микросекундах

На входе Аh в 86h:

22 23

*Тема 4*

**Средства Windows для работы с таймером ОС**

Windows для работы с устройствами кроме прерывания использует функции

расширенного программного интерфейса (API).

Рассмотрим применение этих функций для работы с таймером ОС.

Для работы с таймером ОС Windows использует функции для чтения

данных:

GetSystemTime

SetSystemTime

Функция GetSystemTime осуществляет выбор данных из структуры SystemTime, в

которой содержатся временные данные.

синтаксис:

Void GetSystemTime (LPSystem.lpst), где LP - long point - указатель длины сегмента, который позволяет выход программы за пределы указанного сегмента; lpst - long point System Time - указатель на структуру SystemTime, в которую поступает текущее время.

Включаем файл <widows.h>

Структура SystemTime:

Typedef struct\_SYSTETIME

{

Word wYear;

Word wMonth;

Word wDayOfWeek;

Word wDay;

Word wHour;

Word wMinute;

Word wSecond;

Word wMillisecond;

}

***пример***

/ \* Программа вывода системного времени ОС windows \* /

* include <stdio.h>
* include <windows.h>

void main ()

{

SYSTIME st;

// чтение системного времени ОС Windows GetlocalTime; // вывод времени в формат час: мин: сек: милисек

printf ( "Системный время ОС:% d% d:% d", wHour, wMinute, wSecond, wMillisecond)

}

24 25

*Тема 5*

**контроллер прерываний**

* каждым прерывания свя связана или иное событие. Система должна распознать, какое прерывание по какому номеру произошло и соответствующем подпрограмму надо выполнить.

Известно два вида прерываний: аппаратное и программное.

Программное прерывание удобно использовать для организации доступа для отдельных общих для всех программ модуля. Приложения могут сами устанавливать свои обработчики прерываний для их последующего использования другими программами.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | Общая структура | | | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | аппаратные | | |  |  |  |  | программные | | | |  |  |  | исключительные | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ситуации | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |
| NI |  |  | NMI |  |  | ОС |  |  |  |  | пользо | |  |  | Ошибка |  |
|  |  |  |  |  |  |  |  |  |  |  | тель | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |



***Ошибка*** - представляет собой исключительные ситуации, которые оказываются и обслужуються после выборки к выполнению команды, которая содержит ошибку.

***ловушки*** - представляют собой исключительные ситуации, о которых сообщают сразу после выполнения команды, которая привела к данной ситуации. CS и IP будут указывать на другую команду, в случае нарушения норм выполнения команд непосредственно на команду, которая нарушила этот порядок.

Прерывания, которые определяются пользователем является примером ловушки.

***Аварийное завершение -*** это исключительная ситуация, возникающая при невозможности точно локализовать источник ошибки и использовать при обнаружении ошибок.

***Аппаратное прерывание IRQ*** - использует маскированные прерывания МИ и немаскируемые NMI ((прерывание интерфейсов и памяти вашего компьютера компьютера, которые всегда проходят через РИС, который имеет регистр маски, маскирует необходимые прерывания.

***программные прерывания*** - удобно использовать для организации доступа к отдельным и общих для всех программных модулей.

Приложения могут устанавливать собственные обработчики прерываний. Для этого прерывания должны быть резедентнимы в памяти. Используя прерывания с медленными устройствами позволяю совместить / О информации с обработкой данных в ЦП.

Некоторые прерывания (первые 5) зарезервированы для использования самим процессором для выполнения некоторых событий. Составление личных программ обработки прерывания и замена стандартных обработчиков является ответственной и сложной работой. Необходимо учесть все тонкости работы аппаратуры и взаимодействия программного и аппаратного обеспечения.

**маскировки прерывания**

Для выполнения необходимой последовательности команд при наличии нескольких прерываний необходимо прерывание с более высоким приоритетом замаскировать. Это можно сделать с помощью команды CLI. Эта команда запрещает только маскируемые прерывания, а немаскируемые всегда обрабатываются процессором. Если используется запрет прерываний с помощью CLI, то в конце обязательные надо поставить команду STI.

**Изменение таблицы векторов прерываний**

Если вашей программе нужно изменить обработку некоторых прерываний, то программа должна пере назначить вектор прерывания на свой обработчик. Одним из путей решения является изменение в таблице соответствующего из векторов прерываний.

26 27

**Последовательность действий для нерезедентних программ обработки прерываний:**

* прочитать содержание элемента таблицы векторов прерываний для вектора с нужным вам номером;
* запомнить закладку это содержание (адрес старого обработчика) в область данных программы;
* установить новый адрес в таблице векторов прерываний так, чтобы она соответствовала начала вашей;
* перед завершением работы программы прочитать с области данных адрес старого обработчика прерывания и записать в таблицу обработчика прерывания.

Для облегчения работы по замене прерывания DOS имеет специальные функции для чтения прерываний и для записи в нее нового адреса. DOS гарантирует, что операцию замены будет выполнено верно.

Аппаратное прерывание. Для управления схемами приоритетов необходимо знать внутреннее устройство РОС (И8259). Поступая, прерывания запомнить "ятовуються в регистр S запросов IRR.

Наиболее интересным по программированию является регистр маски IMR и регистр обслуживания ISR.

* машинах класса ХТ регистр маски прерывания 21h; управляющий регистр прерывания = 20h.

Для машины АО первый контроллер имеет такие же адреса, а второй IMR (регистр маски) = А1h. Можно запретить также немаскируемое прерывания. В ячейках АО и Хт предусмотрены регистры, с помощью которых программист может блокировать вход немаскируемого прерывания.

Для ХТ маскировки немаскируемого прерывания управляет регистр (порт) по адресу 0А0h. Если записать в него 0, немаскируемое прерывание будет запрещено, а если 80h- разрешено.

Для Ат запретом немаскируемого прерывания руковожу бут 7 порта 070h

28

**Типы команд РОС:**

Существует 2 типа команд, которые программа посылает в РОС:

1. Команда выбора режима (ICW)
2. Команда управления (OCW), с помощью которых можно выполнить следующие операции:

- индивидуальное маскировки запросов прерывания; - специальное маскировки обслуживающих запросов;

- установление статуса уровней приоритета обслуживания;

- операции конца прерывания (обычный, специальный и автоматический) - чтение регистра IMR, ISR, IRR

**Команды бывают 3х типов:**

1. Маскировки запросов прерываний;
2. Команды обработки конца прерывания;
3. Опрос регистров и специальное маскировки;

Байты команды маскировки запросов прерывания выводятся в порты 20h и 21h (для первого и второго РОС АО - соответственно).

Команды операций конца прерывания для 2го и 3го типа (команды обработки конца прерывания, опрос регистров) использует порты с адресами 20h и A0h, для 1го и 2го контроллера соответственно. Для маскировки какого-либо уровня прерывания надо записать в регистр масок IMR по адресу 21h или A1h единицу.

**Команды обработки конца прерывания**

По команде обычного конца прерывания программистом устанавливается в нулевое состояние разряд IRS последнего обслуживаемого запроса.

29

Команда специального конца прерывания устанавливает в нулевое состояние то ISR, номер которого указан в D0, D1, D2 команды.

Команды циклического сдвига с обычным концом прерывания устанавливает в 0 разряд ISR последнего обслуживаемого запроса и этому же номеру запроса придает низкий уровень приоритета.

Аналогично работает команда циклического сдвига уровней приоритета со специальным концом прерывания. Только низкий уровень приоритета присваивается том входа IRQ, номер которого указан в разрядах b0, b1, b2 команды.

Команда циклического сдвига уровней приоритета со специальным концом прерывания устанавливает статус уровней приоритета без выполнения конца прерывания.

Разряды b0, b1, b2 указывают действие приоритетного кольца.

**Команды опроса регистров и специальное маскировки**

Для получения содержания IMR необходимо выполнить чтение портов по адресам 21h и A1h соответственно.

Команда установки битов специального маскировки блокирует действие тех разрядов ISR, которые замаскированы командой маскировки индивидуальных приоритетных уровней запроса прерывания

Специальное маскировки используется для такого запроса, который блокируется старшим или равным по уровню приоритета обслуживаемым запросу, который хранится в ISR.

**Режимы работы РОС:**

1. Режим фиксированных приоритетов (ЭР) - 0 выше;
2. Автоматическое смещение приоритетов;
3. Программно-управляющий содержание прерывания - программист задает дно прерывания;
4. Автоматическое завершение обработки прерывания - в обычном режиме сбросить свой бит в ISR. В этом редими это осуществляется автоматически и не надо выдавать команду завершения ЭОИ. Недостаток этого прерывания состоит в том,

что в процессе обработки появление повторного прерывания теряется;

1. Режим специальной маски - данный режим позволяет отменить обычный режим прерывания;
2. Режим опроса - в этом режиме аппаратное прерывание не происходит автоматически. Появление запросов прерывания ввизначаеться чтением IRR.

**программирование РОС**

Для программирования Рис используется 2 порта прерывания 20h и 21h.

Через эти порты передаются 4 команды инициализации: IСW1 ... ICW4 которые определяют

режим рооботы РОС и 3 управляющих команды: ОСW1 ... ОCW3

В порт 21h (с парной адресу) вводится IСW1, IСW2, IСW3 (бит D4

показуэ ICW или OCW).

В порт 21h (с нечетным адресу) вводится IСW1, IСW2, IСW3, OCW1. Следует памяти закладку: слова инициализации должны следить порядок ввода

команд, которых указано для портов 20 и 21 h.

Для 1го РОС IСW3 нужно. Наличие IСW4 определяет каскадирования

**Формат команды ICW1**

D7D6D5D4D3D2 D1 D0

D0 = 1 - ICW4 будет

0 - ICW4 отсутствует

D1 = 1 - 1 РОС

0 - каскад

30 31

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D2 Адресный интервал команды call | | | | | | | | | | | | - устанавливать / сбрасывать режим специальной маски; |
| 1 | - 4 | байты | | | | | | | | | | - Переводить PIC в режим опроса и считывать состояние регистра ISR; |
| 0 | - 8 | байт | | | | | | | | | |  |
| D3 = 1 - Запись или сброс битов IRR специальной командой | | | | | | | | | | | | **Режим опроса PIC** |
| 0 | - бит в IRR сбрасывается сигналом INTA при установке в ISR | | | | | | | | | | |  |
| D4 = 1 - определяет, что вводится ICW1 | | | | | | | | | | | | Предназначен для использования в случае, когда количество внешних |
| 0 | - определяет, что вводится ICW2 и ICW3 | | | | | | | | | | | приспособлений велика. |
| D5, D6, D7 - разряды младшего байта адреса А0 = 0 | | | | | | | | | | | | В этом случае командой OCW3 PIC перводиться в режим опроса и CPU |
|  |  |  |  |  |  |  |  |  |  |  |  | последовательно, с помощью команды READ опрашивает состояние регистров ISR или IRR и |
|  |  |  |  | **Формат команды ICW2** | | | | | | | | определяет необходимость робот из устройством, пригласил прерывания. В режиме |
|  |  |  |  |  |  |  |  |  |  |  |  | опроса сигнал INT и INTA не используются. |
|  |  |  | A15 A14 A13 A12 A11 A10 A9 A8 | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

A0 = 1;

**Формат команды ICW3M:**

A0 = 1 - Подключен каскад

0 - не подключен

ICW3M определяет, к я кого вывода подключена подчинена PIC

**Формат команды ICW3S:**

D0, D1, D2 - определяют идентификатор подчиненной PIC

**Работа PIC без пере инициализации**

* компьютер IBM PC возможны следующие операции: - Маскировать / розмаскуваты аппаратные прерывания; - изменять приоритеты уровней;

- задавать команду завершения аппаратных прерываний;

32 33

*Тема 6*

**И / О информации компьютера.**

**Программируемый порт последовательной передачи данных**

Данные передаются и принимаются в виде последовательных импульсов с использованием одной линии. Для распознавания группы символов служит стартовый бит. Затем передаются биты данных в виде сигналов высоких и низких уровней. Последний бит данных может сопровождаться битом паритета (битом четности), который используется для обнаружения ошибок. Для выявления конца символа включается 1 или более стоповых битов. Информационные биты данных, бит четности, стартовый и стоповый биты - определяются протоком для обмена информацией. Приемник и передвавч можуться использовать один и тот же протокол.

Другая важная характеристика - скорость передачи. Она должна быть одинаковой для приемника и передатчика. Измеряется в бодах.

БОД - это количество битов в секунду, учитывая стартовые и стоповые биты, а также бит четности.

**Основные функции, которые выполняет UART (УСАПП)**

1. Обеспечивает преобразование параллельного кода в последовательный при передаче данных и обратное преобразование при их приеме.
2. Формирование стартового, контрольного (паритета) и стопового битов при передаче данных.
3. Контроль верности при приеме стартового, контрольного и стопового битов.
4. Прием-передача знака на фиксированных скоростях
5. Формирование и контроль состояния сигналов в интерфейсе RS232S
6. Организация диагностической проверки при использования дополнительного оборудования.

**Аппаратная реализация СОМ**

Компютеры IBM PC имеют 2 порта последовательной передачи данных, так как ОС поддерживает только 2 порта.

Внешние устройства подключены к ввода / вывода через разъем объемы DB-9p-i DB-

25B

**Программирование UART 8250**

Адаптер имеет 10 программируемых однобайтных регистров, с помощью которых управляется и контролируется обмен информацией.

Доступ к этим 10 регистров осуществляется с помощью 7 адресов: это порты:

3F8h, 3F9h (COM1), 2F8h, 2F9h (COM2).

Асинхронный адаптер вырабатывает прерывание COM1 IRQ (INT0Ch) и COM2 IRQ3 (INT0Bh)

На этапе инициализации системы модуль POST BIOS Тетсуи синхронный адаптер; Их базовые адреса размещены по адресу 0000: 0400h Рассмотрим алгоритм и форматы портов

**порт 3F8h**

Этот порт соответствует регистру данных, передаваемых. Для передачи в порт 3F8h необходимо записать байт, который передается. При приеме даны от внешнего устройства данные могут быть прочитаны из этого порта. Таким образом, порт 3F8h используется для приема и передачи данных.

Прием и передача данных осуществляется при условии, что в регистре управления 3FВh бит D7 = 0. Если D7 = 1, то порт 3F8h используется для ввода значения младшего байта делителя частоты тактового генератора.

34 35

|  |  |  |
| --- | --- | --- |
|  | по | его содержания программист сможет определить причину прерываний. формат: |
| **Порт 3FВh- управляющий регистр** | D0 | = 1 - нет прерываний, ожидающих обслуживания |
|  | D1, D2 = 00 - прерывание по линии состояния приемника (ошибка возникает при | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | D7 | | D6 D5 D4 D3 D2 D1 D0 | | | | | | | переполнении буфера приемника, ошибка четности или формата данных). | | |  |
|  |  |  |  |  |  |  |  |  |  |  | Похоже после чтения данных 3F9h | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  | = 01 - | данные приняты и доступны для чтения. похоже после |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| D0, D1 - длина слова. | | | |  |  |  |  |  |  |  | чтение порта состояния линии 3FDh | | |  |
| D2 - | количество стоповых битов (0 - 1, 1 - 2;) | | | | | | | | | |  | = 10 - | буфер передатчика пуст, похоже после записи данных в |  |
| D3, D4 контроль четности (00 - не используется | | | | | | | | | | | порт 3F8h | |  |  |
| 01- |  |  |  | контроль на нечетность | | | | | | |  | = 11 - | состояние модема. Устанавливается при изменении состояния входных |  |
| 11 - контроль четности) | | | | | | | | | | | сигналов линии CTS, DSR, R | | |  |
|  |  |  |  |  |  |  |  |  |  |  | D3-D7 - не используются | | |  |
| D5 - фиксация четности 0 - контроль четности | | | | | | | | | | |  |  |  |  |
|  |  |  | 1 - контроль на нечетность | | | | | | | |  |  | **Порт 3FСh - регистр управления модемом** |  |
| D6 - установление прерывания | | | | | | | | | | |  |  |  |  |
| D7 - | если 1 - порты | | | 3F8h - 3F9h используются для загрузки | | | | | | | Руководит состоянию выходных линий DTR, RTS линией модема OUT1, OUT2, запускает | | |  |
| константы разделитель частоты | | | | | | | | | | | диагностики. формат: | | |  |
|  | 0 - порты используются обычные порты 3F8h - прием- | | | | | | | | | | D0 | - линия DTR |  |  |
| пердач, 3F9h - установление разрешения обработки прерывания. | | | | | | | | | | | D1 | - линия RTS |  |  |
|  |  |  |  |  |  |  |  |  |  |  | D2 | - OUT1: 0 - | единица на выходе |  |
|  | **Регистр управления прерываниями адаптера порт 3F9h** | | | | | | | | | | D3 | - OUT2: 1 - | запрет выхода прерываний |  |
|  |  |  |  |  |  |  |  |  |  |  |  | 0 - | разрешение выхода прерываний |  |
| Регистр разрешения прерывания используется для управления | | | | | | | | | | | D4 | - запуск диагностики (вход асинхронного адаптера замкнут на выход) | |  |
| прерываниями от асинхронного адаптера или для ввода значения старшего | | | | | | | | | | | D5- D7 - не используются | | |  |
| байта делителя частоты тактового генератора | | | | | | | | | | |  |  |  |  |
| Если программист не использует прерывания, то все равно необходимо | | | | | | | | | | |  |  | **Порт 3Fdh - регистр состояния линий** |  |
| произвести запись в регистр | | | |  |  |  |  |  |  |  |  |  |  |  |
| Регистр управления прерываниями имеет формат: | | | | | | | | | | | Предназначен для контроля модема при приеме и передаче информации. | | |  |
|  |  |  |  |  |  |  |  |  |  |  | Формат регистра: | |  |  |
| Регистр идентификации прерываний порт 3FАh | | | | | | | | | | | D0 | = 0- буфер приема регистра пустой | |  |
|  |  |  |  |  |  |  |  |  |  |  | 1- | загружен | |  |

36 37

|  |  |  |
| --- | --- | --- |
| D1 | - сигнал переполнения при приеме | |
|  | 0 - | переполнения отсутствует |
|  | 1 - | переполнения |
| D2 | - ошибка паритета | |
| D3 | - стоп биты | |
| 0- |  | нет |
| 1- |  | есть |
| D4 | - прерывание обрыва линий | |
| 0- |  | нет |
| 1- |  | есть |
| D5 | - буфер передатчика | |
| 0-заватажено | | |
| 1 - | пустой | |
| D6 | - регистр сдвига | |
| 0- |  | загружен |
| 1- |  | пустой |
| D7 | - не используется | |

***Алгоритм программирования COM-порта***

1 - в порт 3FBh послать 80h

2 - в порт 3F8h послать младший байт, а в 3F9h старший байт делителя частоты

3 - в порт 3FBh послать константу нужного режима D7 = 0

4 - разрешение всех прерываний в порт 3F9h посылаем 0Fh

5 - в порт управления модемом 3FСh посылаем 0

Пункты 1-5 являются общими для приема - передачи

**Поддержка синхронного адаптера в BIOS**

Существует 4 процедуры последовательной передачи данных, общих для всех IBM PC. Эти процедуры доступны через прерывания INT 14h. Они имеют номера от 0 до 3 и выбираются с помощью регистров Аh

|  |  |  |
| --- | --- | --- |
| № | значение | функция |
| ф-й | Ah |  |
|  |  |  |
| 0 | Ah = 0 | Инициализация параметров последовательного порта |
|  |  |  |
| 1 | Ah = 1 | Передать один символ |
|  |  |  |
| 2 | Ah = 2 | Получить один символ |
|  |  |  |
| 3 | Ah = 3 | Получить состояние последовательного порта и модема |
|  |  |  |

***Рассмотрим процедуры:***

*Процедура 0:*

*Устанавливает 4 параметра последовательного порта;*

*скорость передачи данных, четность, количество*

*стоповых битов, довжину4 сримвола.*

*Эти параметриобеднуються в 8-битный код,*

*помещается в регистр Al. Формат Al (на входе*

*АН = 0, DX = 0- COM1, DX = 1-COM2)*

*Процедура 1:*

*Передать 1 символ (байт). Эта процедура передает внешнему устройству через последовательный порт 1 символ. Символ размещается в регистре Al. Для передачи байта на входе:*

*АН = 01h DX (номер порта) 0 - СОМ1,1 - СОМ2*

*Al- байт, который передается.*

*На выходе, после выполнения Al - сохраняется, АН - состояние порта асинхронного адаптера. Якзо бит D7 регистра AH*

38 39

*установлен в единицу, то возникла ошибка.*

*Процедура 2:*

*Сообщение об ошибке является отсылкой от нормы.*

*Так как бит D7 указывает, что была ошибка, то нет*

*возможности распознать TIMЕOUT. Для этого следует использовать сообщения о состоянии процедуры 2.*

На рисунке показана блок-схема DMA. Каждый из каналов К0-К3 имеет регистры:

16-битный регистр адреса, 14-битный счетчик и 2-битный регистр режима канала. Последние объединяются в 1 16-битный регистр, который называется счетчиком.

К выполнению операции КПДП надо инициализировать. В регистровые адреса загружается начальный адрес области памяти. В счетчик завантижиты количество передаваемых байт. В регистр режима канала нужный режим его работы (00 - проверка; 01 - запись 10 - считывания 11-запрещенная комбинация).

***Обмен информацией может осуществляться в одном из 4-х режимов:***

1. Режим одиночной передачи (Single Transfer Mode) - КПДП освобождает шину и начинает проверку сигналов запросов DRQ и при наличии запросов инициализирует следующий цикл передачи.
2. Режим блочной передачи (Block Transfer Mode) - наличие сигнала нужна только до момента выдачи контроллером сигнала Dack. После этого шина освобождается после передачи всего блока.
3. Режим передачи по требованию - передача осуществляется дока активный сигнал запроса DRQ, состояние которого проверяется после каждого цикла передачи (с медленными устройствами).
4. Каскадный режим. В этом режиме 1-ке из выходов канала главного контролера используются для каскадирования с подчиненным контроллером.

**Назначение основных каналов**

1. I / OR - считывание I / O;
2. I / OW - запись I / O;
3. А3..А0 - 4 двунаправленный линии адреса.

Если I / OR = 0, байт шины данных загружается в регистр КПДП по адресу А3..А0.

Если I / OW = 0 - содержание адресует моего регистра КПДП передается на шину данных.

I / OR-MEMW - эта пара генерирует сигнал для передачи в основную память.

I / OR-MEMR - эта пара генерирует сигнал для передачи с основной внешней памяти.

Программная модель КПДП городе 18 8-битных регистров.

Эти регистры делятся на регистры, которые являются общими для этих каналов и на регистры для каждого канала отдельно.

1. Регистр начального адреса (Base Adress Register) - в этом регистре задается начало адресов ОЗУ, с которого начинается передача. Регистры содержит 16

разрядов и определяет адрес страницы. И смещение внутри страницы.

1. Регистр начала счетчика циклов - в этом регистре задается начальное число циклов для программируемого канала.
2. Регистр текущего адреса. Начальное значение заносится в этот регистр одновременно с регистром начале адреса. Далее следует передать значение текущего регистра за каждый цикл.
3. Регистр текущего значения счетчика циклов. Регистр содержит текущее число циклов передачи, которые остались.
4. Регистр режима (Mode Register)

D7D6D5D4D3 D2 D1 D0

40 41

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| D0- D1 - 00 - проверка | | | D0 | - | 0 - | | запретить память в память |
| 11 | - | запись в память |  |  | 1 | - | позволить |
| 10 | - | чтение по памяти | D1 | - | 0 | - | запретить фиксацию адреса каналов |
| 11 | - | запрещенная комбинация |  |  | 1 | - | позволить |
| D2 - 0 - |  | автоинициализация запрещена | D2 | - | 0 | - разблокировать DMA | |
| 1 - автоинициализация разрешена | | |  |  | 1 | - заблокировать DMA | |
| D3 - Изменение потомные адреса при обмене | | | D3 | - | 0 | - нормальная передача | |
| 0 - |  | увеличение (инкрементация) |  |  | 1 | - | сжатие передачи (сжатие времени передачи) |
| 1 - |  | уменьшение (декрементация) | D4 | - | 0 | - | режим фиксированных приоритетов |
| D4- D5 - Тип передачи | | |  |  | 1 | - режим циклического сдвига приоритетов | |
| 00 | - | режим передачи по требованию | D5 | - | 0 | - | задержка при записи |
| 01 | - режим одиночной передачи | | 1 - | режим расширенного записи (если бит D3 = 1, то D5 игнорируем) | | | |
| 10 | - режим блочной передачи | | D6 | - | 0 | - активным является высокий уровень DRQ | |
| 11 | - | каскадный режим | 1 - | | низкий | | |
| D6- D7 - не используются | | | D6 | - | 0 | - активным является высокий уровень сигнала Dack | |
| Каждый из 4-х каналов КПДП имеет свой набор регистров, которые описаны выше. | | | 1 - | | низкий | | |
| Кроме того DMA содержит общий регистр для всех каналов | | |  |  |  |  |  |
| Регистр управления: | | |  |  | **регистр состояния** | | |
| формат: |  |  | Отражает текущее состояние запросов и передач по всем четвёртого каналам | | | | |
|  |  | D7 D6 D5 D4 D3 D2 D1 D0 | биты | | D0- D3 -1-эта указывает канал, который работает. | | |
|  |  | 42 |  |  |  |  | 43 |

**регистр маски**

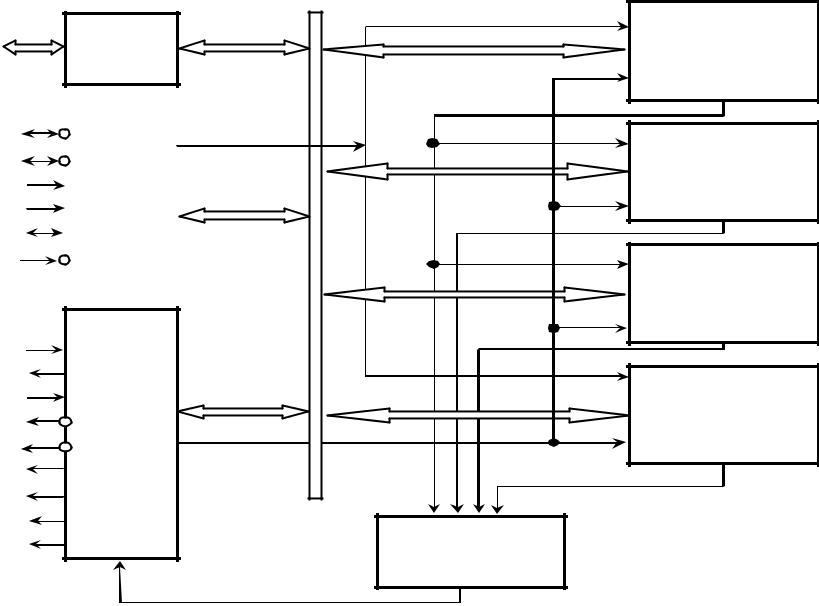
4-битный регистр. Маскирует / демаскирует свой канал контроллера - 1- маскирует канал 0

- позволяет прием запроса по этому каналу.

Регистр запросов (Request Register)

*Тема 7*

**КПДП (DMA) в компьютерах IBM PC**



буфер

D7-0

ШД

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | управление | вводом-выводом |  |
|  | I / OR | | | |  |
|  |  |  |  |
|  |  |  |  |  |  |  |  |
| I / OW | | |  |  |  |
|  | CLK | | | |  |  |  |
| RESET | | | | |  |  |  |
|  |  | A3-0 | | |  |  |  |
|  |  | CS |  | |  |  |  |
|  |  |  |  |  |  |  |  |

1. 7-4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| READY | | | | | Пристрийкерування | тарегистррежима |  |
| ADSTB | | | | |  |
|  |  | HRQ | | |  |  |  |
|  |  | HLDA | | |  |  |  |
|  |  |  |  |  |  |  |  |
| MEMR | |  |  |  |
|  | MEMW | |  | |  |  |  |
|  |  | AEN | | |  |  |  |

|  |  |
| --- | --- |
| TC |  |
| MARK | управление |
|  | приоритетом |

**K0** (Адрес, DRQ0 счетчик,

режим DACK0

**K1** (Адрес,  DRQ1 счетчик,

режим DACK1

**K2** (Адрес, DRQ2 счетчик,

режим DACK2

**K3** (Адрес, DRQ3 счетчик,

режим DACK3

CLK - вход для подключения тактового генератора;

*CS - выбор кристалла;*

*RESET - сброс микросхемы;*

*READY - сигнал готовности;*

*HLDA - подтверждение захвата шины;*

*DREQ3 - DREQ0 - входы запросов на ПДП от внешних устройств;*

*DB7 - DB0 - двунаправленная шина данных с буфером;*

*IOR - сигнал чтения;*

*IOW - сигнал записи;*

*EOP - окончания процесса;*

*А3 - А0 - адресные входы / выходы*

*А7 - А4 - адресные выходы, на которые в режиме ПДП передаются соответствующие*

*разряды адреса ОЗУ;*

44 45

*HRQ - выход запроса захвата шин;*

*DACK3 - DACK0 - подтверждение ПДП;*

*AEN - разрешение адреса.*

*ADSTB - строб (Импульс) адреса.*

*MEMR - чтение из памяти.*

*MEMW - запись в память.*

*Ucc - шина питания (+5 В).*

*GND - общий.*

**Регистры каналов DMA для IBM PC XT**

***Каждый канал содержит 16-разрядные регистры:***

- Регистр текущего значения адреса CAR (Count Adress Key) - содержит текущий адрес ячейки памяти вашего компьютера, при использования операций по обмену данными с использованием DMA;

- Регистр циклов CWR (Count Word Register) - содержит число слов - 1

предназначенных для передачи;

- Регистр хранения базового адреса BAR (Base Adress Register) - используется для хранения базовой (начальной) адреса памяти, в процессе обмена информацией, содержание данного регистра не изменяется;

- Регистр хранения базового числа циклов прямого доступа к памяти вашего компьютера;

- Регистр режима MR (Mode Register) определяет работу канала.

Рассмотрим адреса указанных регистров, которые используются при программирования DMA.

Порты 00h-07h: Эти регистры (порты) содержат базовые адреса и содержание счетчиков передаваемых слов для каналов 0-3. Их назначение:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *№ Порта* | *запись* | *чтение* | *№ Порта* | *запись* | *чтение* |
|  |  |  |  |  |  |
| 00h | Базовая | Текущая | 01h | счетчик | текущее |
|  | адрес | адрес |  | канала 0 | значение |
|  | канала 0 | соответствующих |  |  | К-0 |
|  |  | каналов |  |  |  |
|  |  |  |  |  |  |
| 02h | Базовая |  | 03h | счетчик | текущее |
|  | адрес |  |  | канала 1 | значение |
|  | канала 1 |  |  |  | К-1 |
| 04h | Базовая |  | 05h | счетчик | текущее |
|  | адрес |  |  | канала 2 | значение |
|  | канала 2 |  |  |  | К-2 |
|  |  |  |  |  |  |
| 06h | Базовая |  | 07h | счетчик | текущее |
|  | адрес |  |  | канала 3 | значение |
|  | канала 3 |  |  |  | К-3 |
|  |  |  |  |  |  |

**порт 08h**

Этот порт используется в качестве управляющего регистра и при чтение как регистр состояния. Формат управляющего регистра рассмотрено в предыдущей лекции.

**Формат регистра состояния**

D7D6D5D4D3D2 D1 D0

D0- D3 - устанавливаются в 1 при достижении счетчиков соответствующих каналов конечных значений.

D4- D7 - устанавливаются в 1, если есть разрешение для работы соответствующего канала.

46 47

**порт 09h**

Регистр запроса. Предназначен для организации программного запроса на DMA. Для использования программного запроса канал должен быть запрограммирован в режиме битной передачи. формат:

D7D6D5D4 D3 D2 D1 D0

D0- D1 - номер канала

00 - канал 0

01 - канал 1

10 - канал 2

11 - канал 3

D2 - 0 - установить программный запрос

1 - сбросить программный запрос

D3- D7 - не используются

**Порт 0Аh.** Регистр маски.

D7D6D5D4D3 D2 D1 D0

D0- D1 - номер канала

00 - канал 0

01 - канал 1

10 - канал 2

11 - канал 3

D2 - 0 - установить маску

1 - сбросить маску

D3- D7 - не используются

**порт 0Вh**. Регистр режима. Формат в предыдущей лекции.

**порт 0Сh**. Регистр сброса триггера байтов. Для загрузки внутренних 16-разрядных регистров контроллера используется последовательный вывод младшего, затем старшего байтов слова. После сброса байтов, можно начинать загрузку 16-разрядных регистров.

**порт 0Dh**. Регистр состав контроллера. После записи в этот регистр возникает сброс контроллера. Для дальнейшего использования контроллер должен быть заново проинициализированы.

**порт 0Еh**. Регистр сброса маски. После записи в этот регистр любого значения, разрешается работа всех 4-х каналов без масок.

**порт 0Fh**. Регистр маскировки / розмаскування нужных каналов. формат:

D7D6D5D4 D3 D2 D1 D0

D0 - 0 - розмаскування канала 0

1 - маскировка канала 0

D1 - 0 - розмаскування канала 1

1 - маскировка канала 1

D2 - 0 - розмаскування канала 2

1 - маскировка канала 2

D3 - 0 - розмаскування канала 3

1 - маскировка канала 3

D4- D7 - не используются

Регистры страниц (порты 81h-8fh)

Для работы с памятью 'ятю контроллер DMA использует 20-разрядные физические адреса. 16 младших битов адресов необходимо записать в регистр базового адреса, а другие использовать для адресации 16 страниц. Таким образом реализуется 20-битная базовый адрес.

*Для адресации регистров страниц для машин IBM PC XT используются порты:*

48 49

81h - регистр страниц канала 2;

82h - регистр страниц канала 3,

83h - регистр страниц канала 1;

87h - регистр страниц канала 0;

**Инициализация канала DMA**

**Для инициализации канала программа должна выполнить следующие действия:**

1. Сбросить триггер байтов командой записи в регистр 0Сh любого

значения;

1. Задать режим работы канала, выполнив запись по адресу 0Вh (в регистр режима MR)
2. Записать младшие 16 бит 20-битной адреса области памяти вашего компьютера, которая будет использована для передачи данных в регистр базового адреса;
3. Записать № страницы (старшие 4 бита 20-битной адреса) в регистр

страниц,

1. Скачать регистр циклов ПДП XWR значением на 1 меньше нужного количества передаваемых байтов;
2. Разрешить работу канала, выполнив запись в регистр маски (0Аh)

Сразу после разрешения канал начнет передачу данных.

**Алгоритм программирования канала DMA**

1. В регистр маски 0Аh внести 1 и замаскировать выбранный канал;
2. Сброс триггера байтов - в порт 0Сh послать нули;
3. Выбрать необходимый режим канала - регистр 0Вh;
4. С помощью управляющего регистра 08h осуществить выбранный режим работы;
5. Вводим базовый адрес соответствующего канала (соответствующий байт)
6. Вводим в регистр базового адреса старший байт;
7. Задать регистр страниц (в D0 заносим 1, все остальные - нули)
8. Задать количество циклов обмена в регистр 01h младший байт;
9. Задать в регистр количества циклов старший байт;

10. Сброс регистра маски 0Аh, после чего контроллер DMA начинает работу.

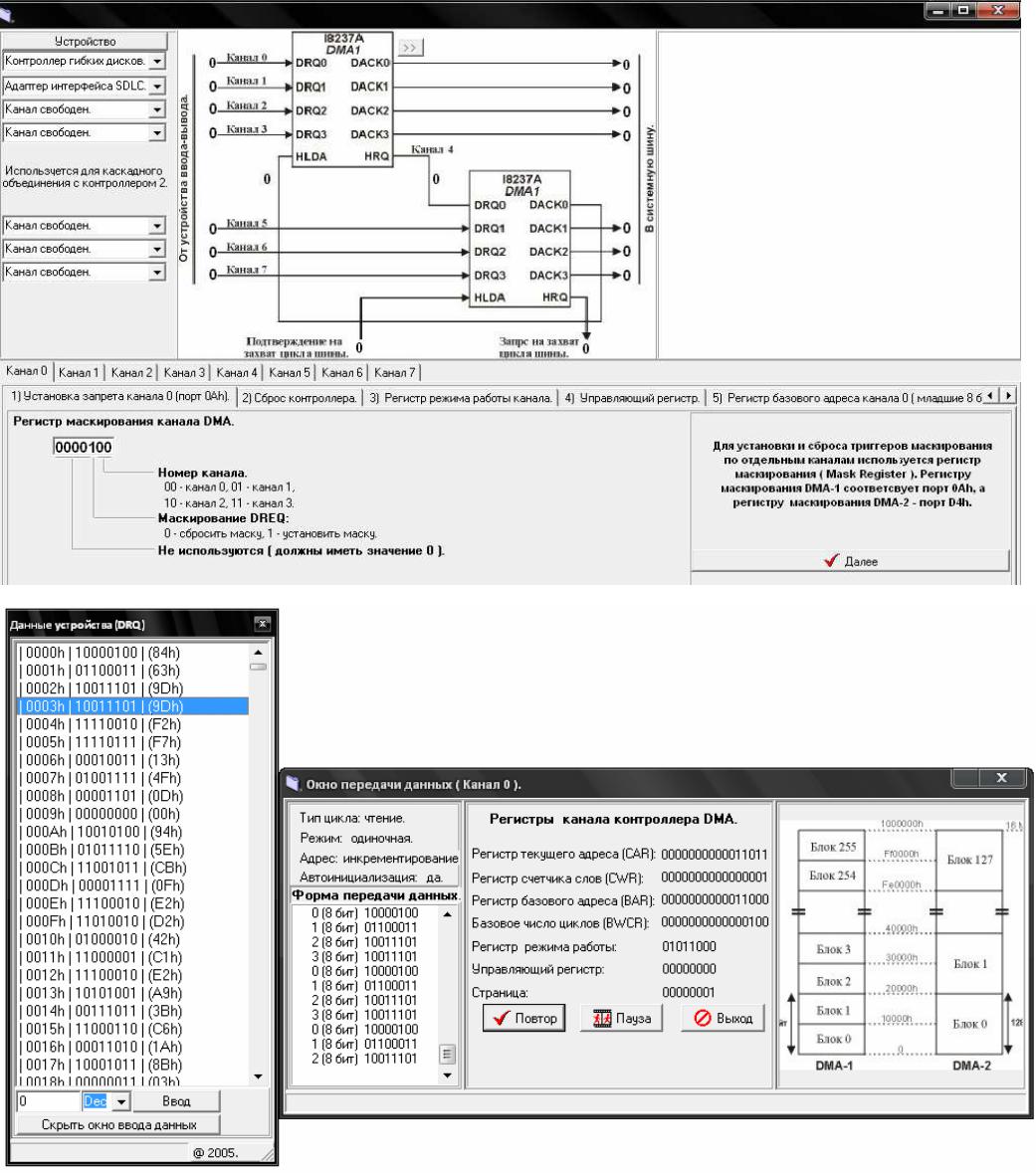
**программирование эмулятора DMA**

Для программирования DMA на эмуляторе рассмотрим 0-вой канал.

Алгоритм программирования эмулятора:

1. Регистр маски 0Аh (00000000)
2. Сброс триггера байтов 0Сh (00000000)
3. Регистр режима канала 0Вh (01011000)
4. Управляющий регистр 08h
5. Регистр базового адреса канала 0h (00011000) (младший байт)
6. Регистр базового адреса канала 0h (00000000) (старший байт)
7. В регистр страницы посылаем 1 (00000001)
8. Регистр базового количества циклов 01h (00000100) (младший байт)
9. Регистр базового количества циклов 01h (00000000) (старший байт)
10. Сброс маски 0Аh (00000000)

50 51



**КПДП в IBM АT**

DMA IBM АT состоит из 2-х каскадно-включенных микросхем 8237А.

Второй контроллер обслуживает каналы DMA с № 4-7, которые обеспечивают 16-байтовую передачу данных.

Используется 8 бит регистров страниц. Таким образом формируется 24-битная

адрес.

81h - регистр страниц канала 2

82h - регистр страниц канала 3

83h - регистр страниц канала 1

87h - регистр страниц канала 0

89h - регистр страниц канала 6

8Вh - регистр страниц канала 5

8Аh - регистр страниц канала 7

Порты 0С / 0h - порты 0С / 0Еh. Эти регистры содержат базовые адреса и содержание счетчиков передавемих данных каналов 4-7.

0С / 0h - запись базового адреса канала 4 / читиння базового адреса канала 4

0С / 2h - запись счетчика канала 4 / читиння счетчика канала 4

0С / 4h - запись базового адреса канала 5 / читиння базового адреса канала 5

0С / 6h - запись счетчика канала 5 / читиння счетчика канала 5

0С / 8h - запись базового адреса канала 6 / читиння базового адреса канала 6

0С / Аh - запись счетчика канала 6 / читиння счетчика канала 6

0С / Сh - запись базового адреса канала 7 / читиння базового адреса канала 7 0С / Еh - запись счетчика канала 7 / читиння счетчика канала 7 Порты 0D0-0DEh

0D0h - управляющий регистр записи / управляющий регистр считывания

0D2h - регистр запроса

0D4h - регистр маски

0D6h - регистр режима

0D8h - регистр сбрасываемый триггера байтов

0DАh - сброс контроллера

0DСh - сброс регистра маски

0DEh - маскировка / розмаскування каналов

Регистр состояния 008-0D0h:

52 53

Временный регистр 008-0D0h для хранения данных во время передачи память-память всегда содержит последний байт.

***пример***: Программирование канала 0 DMA на режим однократной передачи 100

байт по адресу 37856h

#include <dos.h>

void main ()

{

long adr = 0x37856;

outportb (0x0a, 4); // маскировка

outportb (0x8,0) // управляющий регистр

outportb (0x0c 0) // сброс триггера

ouportb (0x0b, 0x54) // 01010100bh - 54h

outportb (0x00, adr & 0xffff)

outportb (0x83, adr >> 16);

outportb (0x01,99) // 100 байт

outportb (0x0a, 0x0) // сброс маски

}

литература

1. Таненбаум Э.С. Архитектура компьютера. Пятый изд. - СПб .: Изд-во "Питер", 2006. - 848 с.
2. Фролов А., Фролов Г. Аппаратное обеспечение IBM PC. / Т. 2, книга 1. - М .:

Диалог-МИФИ, 1992. - 208 с.

1. Голенкова Ж.К., Заблоцкий А.В., Мархасин Н.Л, Порошин С.С., Цесин Б.В., Шугаев А.Н. Руководство по архитектуре IBM РС АО. - Минск: "Консул", 1992. - 949 с.
2. Гольденберг Л.М., Малев В.А., Малько Б. Цифровые устройства и микропроцессорные системы. Задачи и упражнения: Учебн. пособие для вузов. - М .: "Радио и связь", 1992. - 256 с.
3. Майоров В., Гаврилов А.И. Практический курс программирования микропроцессорных систем. - М .: "Машиностроение", 1989. - 263 с.
4. А. Фролов, Г. Фролов. Аппаратное обеспечение IBM PC. / Т. 2, книга 2. - М .:

Диалог-МИФИ, 1992. - 200 с.

1. И.В. Хмелевский, В.П. Битюцкий. Организация ЭВМ и систем. Однопроцессорные ЭВМ. Часть 3 .: Конспект лекций / второй изд., Испр. и допол. Екатеринбург: ГОУ ВПО УГТУ-УПИ, 2005. 100 с.
2. Журден Р. Справочник программиста на персональном компьютере фирмы IBM. - М .: "Радио и связь", 1992. - 356 с.
3. Нортон П. Программно-аппаратная организация компьютера IBM РС. Пер. с

англ. Писарев С.Е. - М .: 1987. - 200 с.

1. Пей А. Сопряжение ПК с внешними устройствами. Пер. с англ. - М .: Изд-во

"ГМК", 2003. - 316 с.

Internet-Поддержка курса АЭВМ - **www.aeom.ho.ua**

54 55