Саратовский государственный университет им. Н.Г.Чернышевского

А.А.Семёнов

ИЗУЧЕНИЕ БИС ПРОГРАММИРУЕМОГО ИНТЕРВАЛЬНОГО ТАЙМЕРА

Учебное пособие
для студентов факультета компьютерных наук
и информационных технологий
и факультета нано- и биомедицинских технологий

Издательство Саратовского университета
2006

УДК 621.374.32 (075.8)
ББК 32.847.6 я73
С30

Семёнов А.А.

С30 Изучение БИС программируемого интервального таймера: Учеб. пособие для студ. фак. компьютерных наук и информационных технологий и фак. нано- и биомедицинских технологий – Саратов: Изд-во Сарат. ун-та, 2006. – 36 с.: ил.

ISBN _-___-_____-_

Учебное пособие представляет собой руководство к практическим занятиям по курсам "Устройство и применение микропроцессоров" и "Микропроцессорные системы". Содержит теоретическое описание материала, знание которого необходимо при выполнении лабораторных и практических работы по изучению БИС программируемого интервального таймера КР580ВИ53.

Для студентов, обучающихся по направлениям “Электроника и микроэлектроника”, “Физика полупроводников и диэлектриков” специальностям “Вычислительные машины, комплексы, системы и сети”, “Микроэлектроника и полупроводниковые приборы”.

Рекомендуют к печати:

Кафедра физики твердого тела факультета
нано- и биомедицинских технологий
Саратовского государственного университета
Доктор физико-математических наук В.Б. Байбурин,
доктор физико-математических наук А.В. Скрипаль

УДК 621.374.32(075.8)
ББК 32.847.6 я73

ISBN _-___-_____-_

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Решение задач реального времени, управление периферийными устройствами часто требуют от микропроцессорной системы (МПС) точного задания временных интервалов между управляющими сигналами. Осуществить это программно достаточно трудоёмко, а зачастую и невозможно из-за сложности временных процессов обмена данными внутри микропроцессорной системы. К тому же вставка тактов ожидания в программу непроизводительно загружает центральный процессор. Подобные задачи в микропроцессорной системе обычно возлагаются на программируемый таймер (ПТ). С помощью ПТ микропроцессор (МП) может формировать временные интервалы произвольной длительности, производить синхронизацию внешних устройств, организовывать счетчики событий, вести счет текущего времени.

В составе микропроцессорного комплекта К580 существует большая интегральная схема (БИС) ***программируемого интервального таймера КР580ВИ53*** – трёхканальное программируемое устройство, предназначенное для организации работы микропроцессорных систем в режиме реального времени. Микросхема формирует сигналы с различными временными параметрами, задаваемыми программно.

Таймер КР580ВИ53 в составе микропроцессорной системы может выполнять достаточно разнообразные функции:

– программируемый тактовый генератор;

– источник опорной частоты при работе других устройств компьютера;

– генератор различных звуковых эффектов;

– счетчик событий;

– цифровой одновибратор;

– часы или датчик реального времени;

– вывод процессора в защищенном режиме из тупиковых ситуаций.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. АРХИТЕКТУРА ПРОГРАММИРУЕМОГО ТАЙМЕРА КР580ВИ53

Структурная схема программируемого таймера представлена на рисунке 1. В состав ПТ входит ***блок логики чтения-записи***, определяющий характер операции и канал, к которому обращается процессор, ***буфер канала данных*** и ***три независимых канала***. ***Блок логики чтения-записи*** (устройство управления) управляет обменом данными между тремя счетчиками

и шиной данных. Обмен информацией между отдельными каналами таймера и микропроцессором осуществляется посредством ***внутренней 8-разрядной магистрали*** через 8-разрядный двунаправленный трехстабильный ***буфер канала данных*** согласно сигналам выборки и управления.

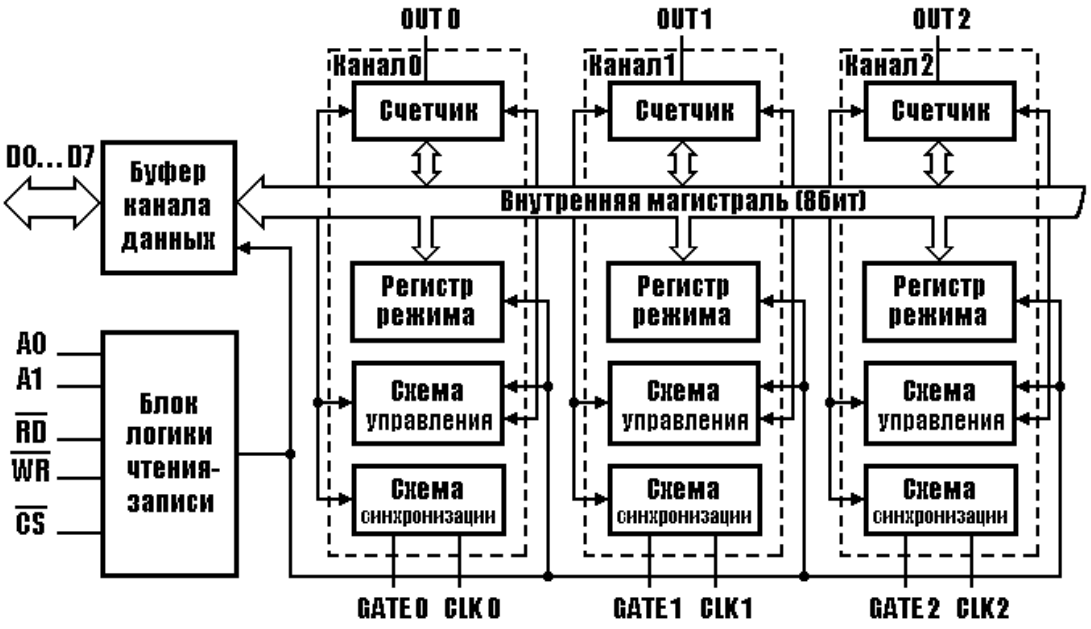


Рис. 1. Структура программируемого таймера КР580ВИ53

Каждый канал включает в себя, ***регистр режима***, ***схему управления, схему синхронизации и 16-разрядный счетчик***.

В ***регистр режима*** канала записывается ***управляющее слово***, определяющее режим работы канала, что позволяет программно настроить любой из трех счетчиков в необходимый режим работы независимо от двух других. ***Схема управления*** организует работу отдельных систем канала в соответствии с запрограммированным режимом работы. ***Схема синхронизации*** формирует серию внутренних счетных импульсов с длительностью и периодом внешних тактовых импульсов, а также синхронизирует работу канала с работой процессора.

В состав отдельного счетчика входят ***регистр хранения***, ***буферный регистр*** и собственно сам ***счетчик***. ***Регистр хранения*** содержит значение константы счета. В начале цикла работы канала константа счета из регистра хранения переписывается в ***счетчик***, и затем по тактовым импульсам на входе CLK происходит декремент содержимого счетчика. Содержимое счетчика в любой момент времени может быть переписано в ***буферный регистр*** и прочитано процессором.

1.2. ПОДКЛЮЧЕНИЕ ТАЙМЕРА К МАГИСТРАЛЯМ МИКРОПРОЦЕССОРНОЙ СИСТЕМЫ

Условное графическое изображение таймера и схема подключения к магистралям микропроцессорной системы показаны на рисунке 2.

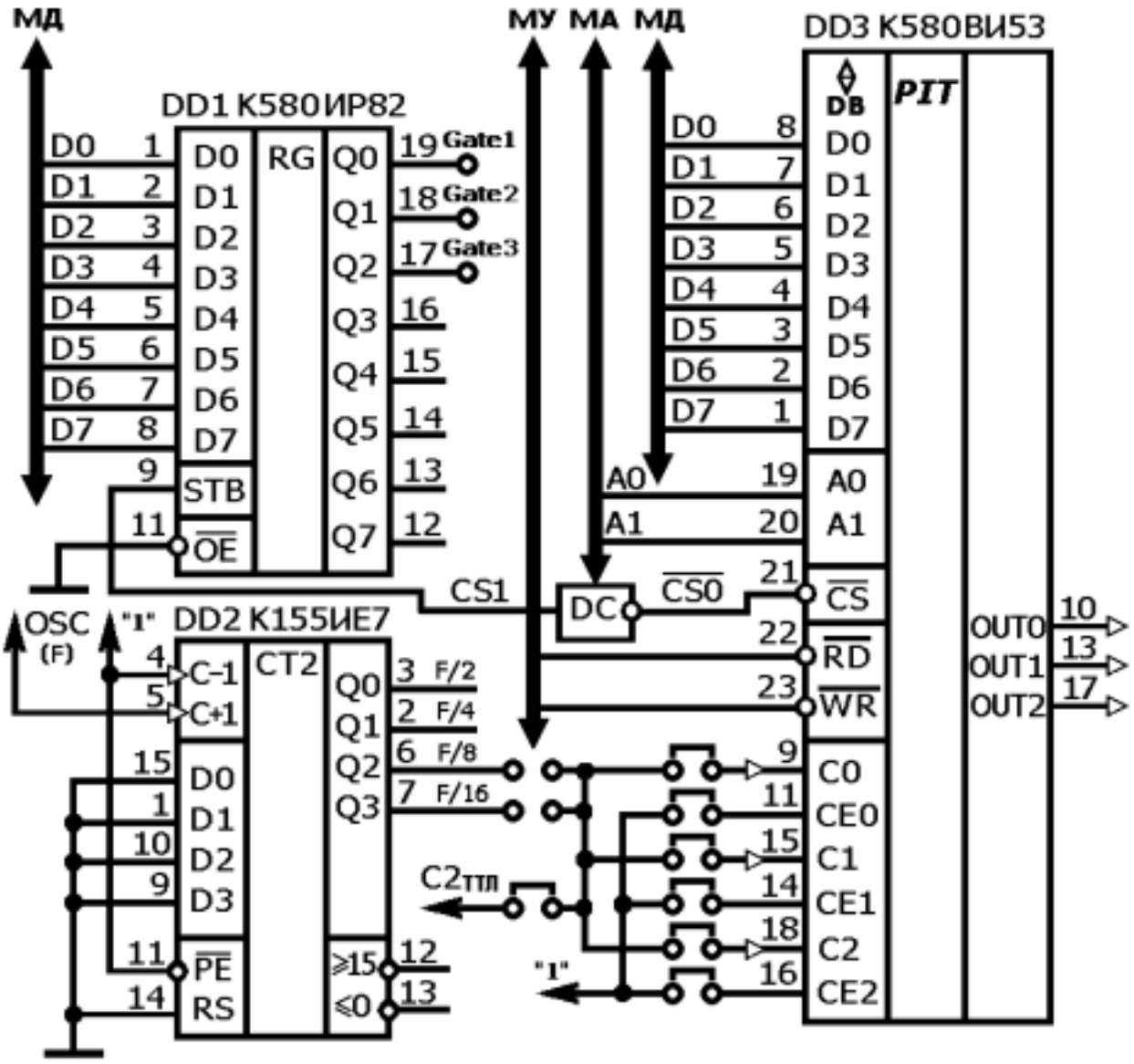


Рис. 2. Схема подключения таймера к магистралям микропроцессорной системы

Назначение входных, выходных и управляющих сигналов ПТ указано при описании выводов микросхемы в таблице 2.

Таблица 2. Функциональное назначение выводов ПТ

Обозначение	Функциональное назначение вывода
D0…D7	Двунаправленная шина данных
$\overline{\mathbf{WR}}$	***Write***. Запись. По низкому уровню на этом входе микропроцессор записывает данные в ПТ
$\overline{\mathbf{RD}}$	***Read***. Чтение. Низкий уровень на этом входе информирует ПТ, что микропроцессор хочет прочитать состояние счетчика
$\overline{\mathbf{CS}}$	***Chip Select***. Выбор микросхемы. Низкий уровень разрешает обмен между процессором и ПТ по активным сигналам шины управления
A0, A1	Адресные входы. Позволяют выбрать регистр режима или один из трех счетчиков для операции чтения/записи
CLK0... CLK2	(или **C** – ***Clock***) Тактовые входы каждого из трех счетчиков
GATE0… GATE2	(или **CE** – ***Clock Enable***) Входы разрешения счетчиков. Уровень "1" - разрешение счета
OUT0… OUT2	Выходы счетчиков

Через выводы **D0÷D7** буфера канала данных ПТ подключается к одноименным линиям магистрали данных (МД) непосредственно или через буферный элемент типа К580ВА86 в том случае, если нагрузочной способности буфера канала данных недостаточно для работы с шиной данных системы. Входы **A0** и **A1**, как правило, подключаются к младшим разрядам магистрали адреса (МА), а поступивший на них код задает следующий порядок подключение буфера канала данных: 00 – к счетчику 0 (СТ0), 01 – к счетчику 1 (СТ1), 10 – к счетчику 2 (СТ2), 11 – к регистрам режима каналов. Код на остальных адресных линиях определяет выбранную микросхему. Сигнал выборки **CS0** с низким активным уровнем формируется схемой дешифратора **DC**, который может быть выполнен на микросхемах комбинационной логики, компараторах кодов и дешифраторах в интегральном исполнении. В МПС, имеющих в своем составе контроллер прямого доступа к памяти (ПДП), работу дешифратора выборки следует блокировать на время циклов ПДП по активному уровню сигнала **AEN** (**A**ddress **En**able), поскольку в этот промежуток времени сигналы магистрали управления (МУ) формируются контроллером ПДП, а не центральным процессором. В МПС, построенных на МП К580ВМ80, дешифрация устройств ввода/вывода может быть упрощена в том случае, если их количество не превышает шести. Поскольку МП К580ВМ80 при обращении к внешним уст-

ройствам командами IN и OUT дублирует на линиях **A15÷A8** код, выставляемый по линиям **A7÷A0**, входы $\overline{\mathbf{CS}}$ микросхем внешних устройств могут подключаться непосредственно к линиям **A15÷A10** МА без дополнительной дешифрации и однозначно адресоваться кодами 111110XXb, 111101XXb, 111011XXb, 110111XXb , 101111XXb, 011111XXb.

Входы $\overline{\mathbf{WR}}$ и $\overline{\mathbf{RD}}$ определяют направление передачи данных и подключаются к линиям $\overline{\mathbf{IOWR}}$, $\overline{\mathbf{IORD}}$ МУ в том случае, если ПТ подключен как устройство ввода/вывода, или к линиям $\overline{\mathbf{MEMWR}}$, $\overline{\mathbf{MEMRD}}$ – если регистры ПТ включены как ячейки памяти. В простых микропроцессорных системах, в которых пространство ячеек памяти и пространство устройств ввода/вывода не разделены, входы $\overline{\mathbf{WR}}$ и $\overline{\mathbf{RD}}$ подключаются непосредственно к соответствующим выводам микропроцессора.

При включении ПТ в пространство устройств ввода/вывода программирование и обмен данными с ним осуществляется командами ассемблера IN и OUT. При работе с регистрам ПТ как с ячейкам оперативного запоминающего устройства (ОЗУ) в полном объеме применимы команды обращения к памяти. В том случае, если пространство ячеек памяти и пространство устройств ввода/вывода не разделены, доступ к регистрам ПТ возможен как с помощью команд обращения к памяти, так и с помощью команд ввода/вывода с учетом специфики работы последних применительно к конкретному типу процессора. Так команды IN 61h и OUT 61h МП К580ВМ80 выполняются в этом случае эквивалентно командам LDA 6161h и STA 6161h.

На тактовые входы **C0÷C2** подается внешний счетный или тактирующий сигнал, частота которого не превышает 2,5 МГц. Превышение этого значения приводит к сбоям в работе ПТ, хотя отдельные экземпляры демонстрируют устойчивую работу и на завышенной частоте. В МПС, построенных на МП К580ВМ80, в качестве тактирующего обычно используют сигнал $С2_{ттл}$ задающего генератора К580ГФ24 (как показано на рис. 2). Возможно также применение отдельного тактирующего генератора или использование подходящего сигнала счетчика регенерации динамического ОЗУ.

В персональных компьютерах IBM PC системный таймер, расположенный на материнской плате, тактируется сигналом частотой 1,19 МГц (по некоторым данным — 1,193182 МГц) независимо от источника тактового сигнала или особенностей конструкции платы, что определяет стандартную для всех компьютеров длительность одного периода сигнала **CLK** — 0,840336 мкс.

При подключении дополнительного таймера через системную шину расширения ISA, в качестве опорного сигнала удобно использовать стандартный сигнал OSC — меандр с частотой F=14,31818 МГц. В этом случае

тактирующий сигнал допустимой частоты формируется интегральным счетчиком-делителем (DD2, на рис. 2). Сигналы с выходов Q2 и Q3 счетчика, частотой соответственно F/8=1,7897725 МГц и F/16=0,89488625 МГц, могут быть использованы в качестве тактирующих.

Если отдельный канал таймера используется для подсчета внешних событий, на его вход **CLK** подается соответствующий внешний счетный сигнал. В ситуации, когда максимального коэффициента пересчета отдельного канала таймера недостаточно для получения выходного сигнала с заданными параметрами, каналы можно включать последовательно. При этом выход **OUT** одного из каналов является источником тактирующего сигнала для **CLK** другого.

Управляющие входы таймера **GATE0÷GATE2** в простейшем случае могут быть подключены к источнику питания +5 В через резистор 1÷10 кОм, что эквивалентно подаче на них уровня логической единицы. Такое включение накладывает ограничение на использование отдельных режимов ПТ, поскольку каналы таймера начинают работу непосредственно после программирования режима и занесения константы счета.

Для программно–аппаратного управления таймером может быть применен отдельный регистр (DD1, на рис. 2), включенный в пространство устройств ввода/вывода и адресуемый сигналом выборки **CS1** с высоким активным уровнем, формируемым схемой дешифратора **DC**. При подключении входов **GATE0÷GATE2** к выходам **Q1÷Q3** управляющего регистра, разрешение и запрещение работы соответствующих каналов таймера осуществляется установкой логических "1" и "0" в разрядах 0 ÷ 2 байта, записываемого в регистр. Программно–аппаратное управление ПТ позволяет использовать все доступные режимы работы каналов без ограничений.

Вход **GATE** одного из каналов может подключаться к выходу **OUT** другого канала, что позволяет при соответствующей настройке каналов формировать пакеты импульсов и сложные звуковые эффекты.

В персональных компьютерах IBM PC входы **GATE** каналов 0 и 1 всегда установлены в высокий уровень "1", в силу чего эти каналы недоступны для программно–аппаратного управления. При этом канал 0 используется для отсчета текущего времени, а канал 1 выступает в роли генератора для схемы регенерации памяти. Вход **GATE** канала 2 управляется битом 0 порта 61h. Выход канала 2 подключен динамику и используется для генерации звука.

1.3. НАЧАЛЬНАЯ ИНИЦИАЛИЗАЦИЯ ТАЙМЕРА

Каналы таймера полностью независимы друг от друга – каждый может быть настроен в свой режим работы. Счетчик каждого канала представляет собой 16–разрядный счетчик с предустановкой, работающий на вычитание в двоичном или двоично–десятичном коде. Таким образом, максимальное значение счета – 2^{16} = 65536 (при работе в двоичном коде) или 10^4 = 10000 (при работе в двоично-десятичном коде) достигается при загрузке нулевой константы в счетчик канала.

Режимы работы ПТ определяются в процессе начальной установки и программируются с помощью простых операций вывода или записи в память. Для приведения каналов ПТ в необходимое рабочее состояние центральный процессор должен задать каждому каналу в указанной последовательности:

режим работы (определяется записью ***управляющего слова*** в ***регистр режима*** ПТ по адресу A0=1, A1=1);

константу (число) для счетчика (один или два байта в зависимости от управляющего слова для данного канала).

Формат управляющего слова приведен в табл. 3. Порядок программирования ПТ произвольный, т. е. можно сначала запрограммировать режимы работы всех каналов, а затем – загрузить счетчики. Счетчик канала должен быть обязательно загружен именно тем количеством байтов, которое было запрограммировано в управляющем слове значениями разрядов D5 и D4. **При загрузке содержимого счетчика двухбайтовым числом первым записывается *младший байт*, затем – *старший*.**

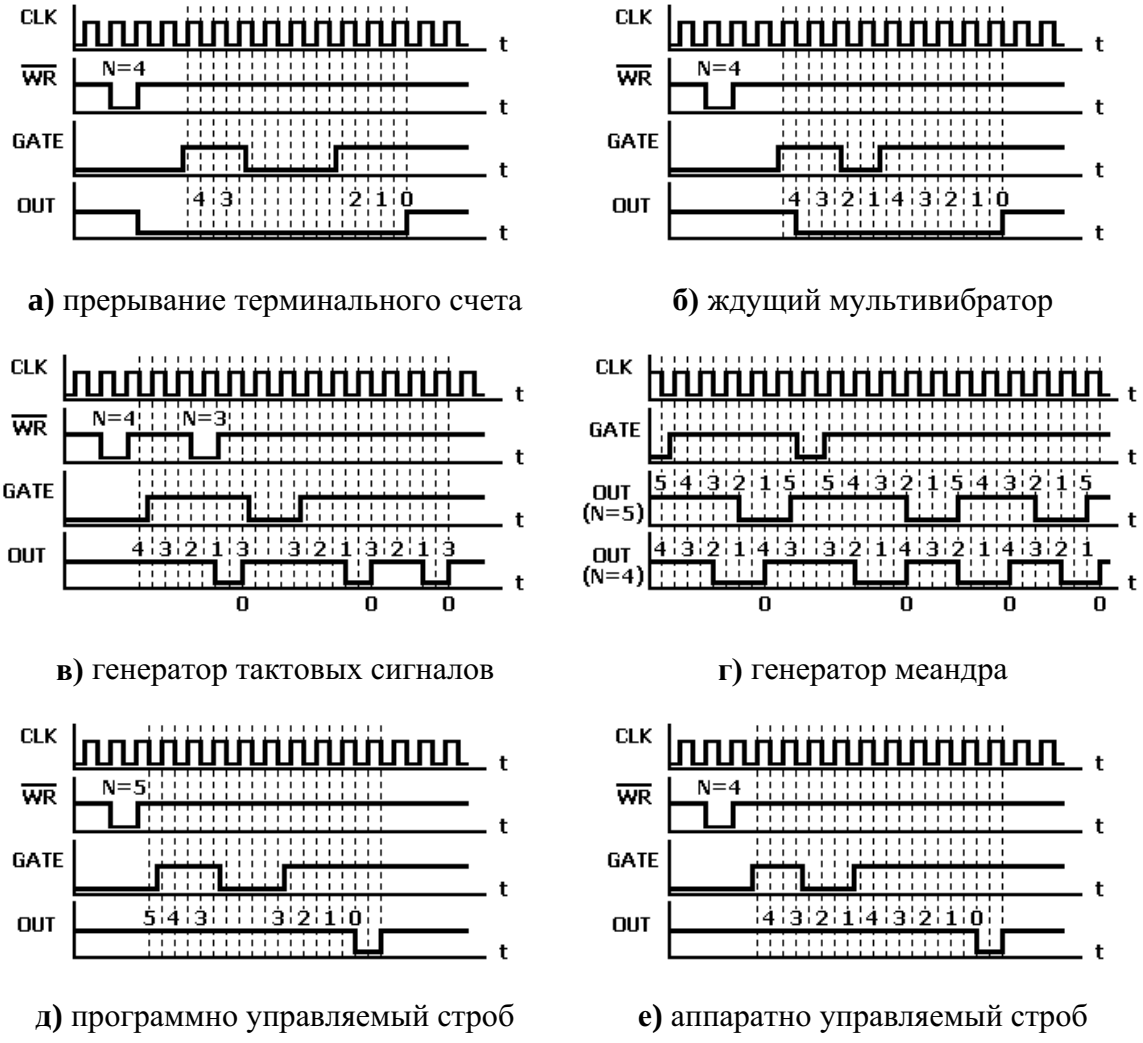
Таблица 3. Формат управляющего слова (побитно) для выбора режима работы

D7	D6	D5	D4	D3	D2	D1	D0
Выбор канала:		Число байтов для загрузки счетчика:		Режим работы канала:			Тип счета:
00 – канал 0 01 – канал 1 10 – канал 2		00 – чтение на лету; 01 – чтение/запись младшего байта; 10 – чтение/запись старшего байта; 11 – чтение/запись слова		000 – режим 0 001 – режим 1 010 – режим 2 011 – режим 3 100 – режим 4 101 – режим 5			0 – двоичный счет; 1 – двоично–десятичный счет.

Поскольку ПТ не имеет отдельного вывода аппаратного сброса (“Reset”, “Начальная установка”), в БИС предусмотрен внутренний программный сброс отдельно по каналам. Сигнал внутреннего сброса формируется при записи управляющего слова в регистр режима выбранного канала.

1.4. РЕЖИМЫ РАБОТЫ ПРОГРАММИРУЕМОГО ТАЙМЕРА

Существует шесть режимов работы каждого канала таймера. Временные диаграммы соответствующих режимов иллюстрирует рисунок 3.



а) прерывание терминального счета

б) ждущий мультивибратор

в) генератор тактовых сигналов

г) генератор меандра

д) программно управляемый строб

е) аппаратно управляемый строб

Рис. 3. Временные диаграммы режимов работы таймера КР580ВИ53

<u>Режим 0</u> – ***прерывание терминального счета*** (выдача сигнала прерывания по конечному числу, рис. 3, а). При работе в этом режиме на выходе **OUT** канала появляется уровень "0" сразу же после установления режима работы. После загрузки числа в счетчик канала выход остается в "0" и счетчик начинает считать, если на входе разрешения **GATE** установлен уровень "1". После того как достигается конечное число, на выходе **OUT**

устанавливается уровень "1" и остается до тех пор, пока канал не будет перезагружен управляющим словом режима работы или новым числом.

Режим 1 – ***ждущий мультивибратор*** с программно – устанавливаемой длительностью сигнала (рис. 3, б). В этом режиме выход **OUT** канала после загрузки числа в счетчик канала устанавливается в уровень "0" после первого тактового сигнала, следующего за передним фронтом на управляющем входе **GATE**. Одновременно начинается счет, а при достижении конечного числа на выходе **OUT** устанавливается уровень "1".

Режим 2 – ***генератор тактовых сигналов*** (рис. 3, в). В этом режиме на выходе канала через число периодов тактовой частоты, на единицу меньшее записанного в счетчике канала (N-1), появляется уровень "0" длительностью в один период тактовой частоты.

Режим 3 – ***генератор меандра*** (прямоугольных импульсов со скважностью 2, рис. 3, г). В этом режиме на выходе канала будет уровень "1" в течение первой половины интервала времени, определяемого числом в счетчике, и уровень "0" в течение второй половины. При нечетном числе длительность сигнала уровня "1" на один такт больше чем для сигнала уровня "0".

Режим 4 – ***программно управляемый строб*** (рис. 3, д). После установки режима 4 на выходе канала появляется уровень "1". Когда число полностью загружено в счетчик канала и на управляющий вход **GATE** подан уровень "1", начинается счет, и при достижении конечного числа на выходе появляется импульс уровня "0" длительностью в один период тактовой частоты.

Режим 5 – ***аппаратно управляемый строб*** (рис. 3, е). Работа канала в этом случае аналогична работе в режиме 4 с той лишь разницей, что счетчик канала после загрузки начинает счет только по переднему фронту на управляющем входе **GATE**. Кроме того, если во время счета на управляющем входе **GATE** снова появится передний фронт сигнала, то счет будет начат сначала.

Результат воздействия сигнала **GATE** на соответствующий счетчик зависит от режима работы канала. Функции выполняемые сигналом **GATE** для различных режимов приведены в таблице 4.

Существуют задачи, в которых процессор должен периодически считывать содержимое счетчиков каналов. Примером такой задачи может служить счет событий. Во время работы счетчика его текущее содержимое может быть передано в буферный и прочитано двумя способами:

- при помощи обычной операции чтения;
- с помощью специальной операции "***чтения на лету***", активизируемой вводом специального управляющего слова.

Таблица 4. Функции сигнала GATE

Режим	Состояние сигнала GATE		
	Низкий уровень или отрицательный фронт (спад сигнала)	**Положительный фронт (нарастание сигнала)**	**Высокий уровень**
0	Запрещает счет	—	Разрешает счет
1	—	Начинает счет; устанавливает уровень"0" сигнала OUT со следующего такта CLK	—
2	Запрещает счет; устанавливает уровень "1" сигнала на выходе	Начинает счет	Разрешает счет
3	Запрещает счет; устанавливает уровень "1" сигнала на выходе	Начинает счет	Разрешает счет
4	Запрещает счет	—	Разрешает счет
5	—	Начинает счет	—

Первый способ заключается в непосредственном чтении показаний счета, причем для обеспечения стабильных показаний работа счетчика должна быть в этот момент приостановлена подачей на управляющий вход **GATE** ***схемы синхронизации*** канала уровня "0" (в режимах 0, 2 – 4) или с помощью внешней логической схемы, приостанавливающей подачу импульсов на тактовый вход канала.

Второй способ позволяет процессору считывать содержимое счетчика, не прерывая процесса счета. Для этого в ПТ по адресу A0=1, A1=1 осуществляется запись управляющего слово, определяющего в соответствии с табл. 3 ***режим чтения на лету*** (D4=0, D5=0). Разряды D6 и D7 определяют номер канала, состояние разрядов D0 – D3 произвольно. По этой команде в ***буферном регистре*** защелкивается текущая величина счета, далее следует обычная операция чтения содержимого счетчика. Данная команда не изменяет содержимого регистра режима. При этом способе чтения нельзя предварительно по каждому каналу производить запись управляющих слов и лишь затем производить чтение счетчиков каналов.

Особенность построения внутренней схемы таймера требует, чтобы операция чтения содержимого счетчика была выполнена до конца, т.е. если запрограммирована загрузка счетчика канала двумя байтами, то нельзя,

прочитав только один младший байт, перезагружать счетчик новой величиной.

Операции обмена информацией между ПТ и МП задаваемые сигналами управления и адресными входами приведены в таблице 5.

Таблица 5. Операции обмена информацией между ПТ и микропроцессором

Операция	Сигналы на входах				
	$\overline{WR}$	$\overline{RD}$	$\overline{CS}$	A1	A0
Запись управляющего слова в регистр режима канала 0 (1, 2)	**0**	**1**	**0**	**1**	**1**
Загрузка счетчика канала 0: D(7-0) → CT0	**0**	**1**	**0**	**0**	**0**
Загрузка счетчика канала 1: D(7-0) → CT1	**0**	**1**	**0**	**0**	**1**
Загрузка счетчика канала 2: D(7-0) → CT2	**0**	**1**	**0**	**1**	**0**
Чтение счетчика канала 0: CT0 → D(7-0)	**1**	**0**	**0**	**0**	**0**
Чтение счетчика канала 1: CT1 → D(7-0)	**1**	**0**	**0**	**0**	**1**
Чтение счетчика канала 2: CT2 → D(7-0)	**1**	**0**	**0**	**1**	**0**
Нет операции. Отключение ПТ от магистрали данных МП.	**1**	**0**	**0**	**1**	**1**
Нет операции. Отключение ПТ от магистрали данных МП.	**1**	**1**	**0**	**x**	**x**
Запрет операций. Отключение ПТ от магистрали данных МП.	**x**	**x**	**1**	**x**	**x**

Примечание. x - безразличное состояние сигнала

1.5. ОСНОВЫ ПРОГРАММИРОВАНИЯ МИКРОПРОЦЕССОРНЫХ СИСТЕМ НА ЯЗЫКЕ АССЕМБЛЕРА

Ассемблерные программы обладают преимуществом компактности кода и максимальным быстродействием, вследствие этого они чаще всего используются при работе микропроцессорных систем в режиме реального времени, а также при необходимости управления внешними устройствами (ВУ) или обмена данными с ними. Операционные системы МПС обычно содержат низкоуровневые подпрограммы (драйверы) для работы со стандартными устройствами ввода/вывода. Потребность в написании ассемблерной подпрограммы может быть вызвана заменой стандартного устройства, оптимизацией кода существующих драйверов, необходимостью обслуживания нестандартной периферии.

Интересным примером использования ПТ и программирования на языке ассемблера является задача последовательного ввода в МПС информации внешнего аналого-цифрового преобразователя (АЦП), работающего по принципу двойного интегрирования.

1.5.1. Формулировка задачи

Первым этапом решения задачи является составление алгоритма, включающего в себя как словесное описание, так и любые виды формализации – таблицы, формулы, графики, принципиальные и блок-схемы. Этап формулирования задачи является одним из самых важных, поскольку ошибки, допущенные в процессе составления алгоритма, сводят на нет все дальнейшие усилия по составлению программы и кодированию.

Временная диаграмма сигналов обмена данными измерения, выведенных на интерфейсный разъем АЦП в рассматриваемой задаче приведена на рисунке 4.

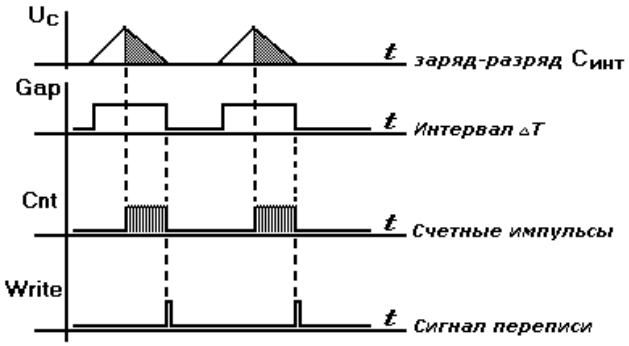


Рис. 4. Временная диаграмма интерфейсных сигналов АЦП двойного интегрирования

В соответствии с принципом двойного интегрирования, разряженный интегрирующий конденсатор $\mathbf{C_{инт}}$ за один интервал измерений **Gap** заряжают определенное время током, пропорциональным измеряемому напряжению, а затем разряжают определенным током до нуля. Время, в течение которого происходит разряд конденсатора, пропорционально измеряемому напряжению. Это время измеряют с помощью счетчика, подсчитывающего пакет счетных импульсов **Cnt**, количество которых определяет измеряемую величину. Поступление сигнала переписи **Write** завершает цикл измерений.

Большинство задач допускает альтернативные равносильные решения. На этапе формулировки задачи следует провести их оценку и определить возможность и целесообразность практической реализации. Так в рассматриваемой задаче сравнительно высокая частота поступления счетных импульсов (128 кГц) не позволяет осуществить их счет в МПС методом программного опроса параллельного порта (что аппаратно более просто), в связи с чем приемлемым и целесообразным решением является применение интервального таймера для подсчета пакета импульсов **Cnt**.

В качестве счетчика импульсов можно использовать любой из каналов таймера, поскольку максимальное количество импульсов при использовании распространенных интегральных АЦП не превышает 20000 еди-

ниц, что существенно меньше максимального числа счета отдельного канала таймера. Сигнал **Gap** удобно использовать, как разрешающий для входа **GATE** таймера, сигнал переписи **Write** – как запрещающий продолжение счета. В силу асинхронности процессов в АЦП и МПС необходимо либо тактировать АЦП тактовым сигналом МП, либо применить устройство, синхронизирующее процесс считывания данных АЦП с временными процессами МПС. Для синхронизации может быть использована схема (рис. 5), использующая свойство приоритета асинхронных входов RS-триггера над синхронными, осуществляющая временную привязку сигналов обмена данными к процессу программного опроса.

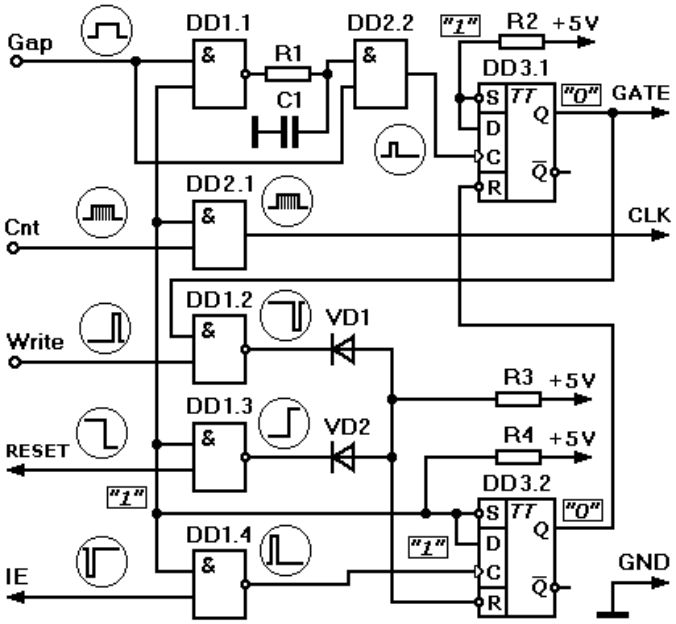


Рис. 5. Принципиальная схема устройства синхронизации

Устройство синхронизации работает следующим образом. При включении питания компьютера сигнал **RESET** через инвертор DD1.3 и диод VD2, образующий вместе с диодом VD1 и резистором R3 схему логического "И", низким уровнем на входе $\overline{\mathbf{R}}$ триггера DD3.2 устанавливает уровень логического нуля на его выходе *Q*. В свою очередь триггер DD3.2, воздействуя низким уровнем на выходе *Q* на вход $\overline{\mathbf{R}}$ триггера DD3.1, переключает его выход *Q*, управляющий входом разрешения счета **GATE** канала 2 интервального таймера КР580ВИ53, в "0", тем самым запрещая прохождение счетных импульсов **Cnt** через вход **CLK** на схему счетчика таймера. Этот же сигнал с выхода *Q* триггера DD3.1 блокирует прохождение сигнала окончания счета **Write** через вентиль DD1.2 и диод VD1 на вход $\overline{\mathbf{R}}$ сброса триггера DD3.2. В силу приоритета асинхронных входов триггеров над синхронными, триггер DD3.2 удерживает через вход $\overline{\mathbf{R}}$ элемент DD3.1 от переключений при воздействии фронта сигнала **Gap** на его тактовый вход *C*.

Два расположенных на плате параллельных порта (шинный формирователь К580ВА86, работающий на ввод и буферный регистр К580ИР82, используемый, как устройство вывода), обеспечивают программный обмен сигналами управления с устройством синхронизации и таймером.

Сформированный программной записью управляющего байта в регистр К580ИР82 импульс **IE** (***I**nput **E**nable* - разрешение ввода) через инвертор DD1.4 своим фронтом переписывает уровень логической единицы,

удерживаемый на входе *D* триггера DD3.2 резистором R4, на выход *Q*, тем самым разрешая работу триггера DD3.1 по тактовому входу *C*. Первый, пришедший после этого, фронт сигнала **Gap** через схему формирования короткого импульса на элементах DD1.1, DD2.2, R1, C1 переписывает единичный логический уровень со входа *D* триггера DD3.1 на его выход *Q*, разрешая тем самым прохождение счетных импульсов по входу **CLK** канала 2 таймера активным уровнем на выводе **GATE**, а также открывает логический вентиль DD1.2 для прохождения сигнала окончания счета **Write**. После прохождения пакета счетных импульсов, сигнал **Write**, воздействуя активным уровнем на вход сброса $\overline{\mathbf{R}}$ триггера DD3.2 через вентиль DD1.2 и диод VD1, возвращает всю схему в состояние, идентичное установившемуся после воздействия сигнала **RESET**. Состояние окончания счета отслеживается программно в цикле опроса сигнала **GATE** через параллельный порт К580ВА86. Весь цикл подсчета импульсов повторяется с поступлением нового разрешающего импульса **IE**, сформированного программно в произвольный момент времени.

Определение задачи: последовательный ввод в МПС данных от внешнего аналого-цифрового преобразователя (АЦП), работающего по принципу двойного интегрирования;

Устройства: АЦП двойного интегрирования, интервальный таймер, параллельные порты ввода и вывода, клавиатура.

Датчики и сигналы: матрица клавиш, опрашиваемая через порт клавиатуры; линии интерфейса АЦП, обслуживаемые таймером через устройство синхронизации.

Суть задачи: Настроить канал 2 таймера в режим прерывания терминального счета. Сформировать через регистр К580ИР82 импульс разрешения ввода **IE** для устройства синхронизации. В цикле программного опроса отслеживать состояние окончания счета и матрицы клавиш. Обеспечить выход из процедуры по нажатию клавиши [ESC]. По окончанию счета прочитать полученное значение из канала 2 таймера и преобразовать его к виду, удобному для обработки и индикации. Восстановить константу счета канала 2.

Задача поставлена. Она содержит работу с внешними устройствами в реальном времени, программирование периферии и портов ввода/вывода.

1.5.2. Решение задачи

Следует определить для себя ответы на ряд вопросов следующего характера:

а) какие данные вводятся или выводятся;

б) какие порты используются и в какие режимы их запрограммировать;

в) в каком виде осуществить диалог с оператором (если диалог нужен);

г) какая информация и в какой форме необходима для вывода;

д) возможное развитие задачи.

Возможные ответы: сигнал разрешения ввода **IE** формируется записью в параллельный порт К580ИР82 управляющего байта, младший бит которого программно устанавливается в "1" и сбрасывается в "0". Первый, пришедший после этого, фронт сигнала **Gap** по входу **GATE** разрешает прохождение пакета импульсов в счетчик таймера. Количество импульсов преобразуется таймером в численное значение. Сигнал **Write** завершает цикл, устанавливая логический "0" на входе **GATE**. Параллельный порт К580ВА86 через младший бит отслеживает логическое состояние входа GATE. Численное значение в виде двухбайтового числа извлекается из счетчика 2 таймера двумя операциями чтения данных.

Параллельные порты осуществляют функции ввода/вывода аппаратно и в настройке не нуждаются. Канал 2 ПТ настраивается в режим прерывания терминального счета с последующей загрузкой константы счета, равной 0FFFFh. Поскольку счетчик таймера работает на вычитание, загрузка его такой константой упрощает программное вычисление значения количества импульсов в пакете.

Для опроса клавиатуры будем использовать стандартную подпрограмму опроса консоли, возвращающую код клавиши в случае её нажатия и код 0FFh – в противном случае. Клавиатура может опрашиваться разными программами, поэтому для совместимости лучше не использовать опрос матрицы клавиш своей подпрограммой. В качестве диалога с оператором или пользователем можно предусмотреть вывод на экран сообщения типа: “Для выхода из программы нажмите клавишу [ESC]” (“Press ESC to terminate…”), но поскольку мы пишем процедуру низкого уровня, а ESC – стандартная клавиша для отмены процесса, подобное предупреждение обычно выводится вызывающей программой.

Информация для дальнейшей обработки или вывода на консоль возвращается программой в регистровой паре микропроцессора в виде шестнадцатиричного двухбайтового числа.

В качестве дальнейшего развития задачи каналы 0 и 1 таймера можно использовать для самотестирования устройства в отсутствие АЦП. Наличие порта вывода и двух независимых каналов позволяет сформировать эталонные последовательности импульсов **Gap**, **Cnt** и **Write** программно–аппаратным методом. С этой целью на входы **CLK** этих каналов подается тактирующий сигнал $\mathbf{C2_{ттл}}$ задающего генератора с частотой 2 МГц, вход **GATE** канала 0 управляется разрядом **Q1** регистра К580ИР82 (рис. 2), а вход **GATE** канала 1 управляется выходом **OUT** канала 0. Такое включение счетчиков таймера позволяет сформировать эталонный пакет импульсов заданной частоты следующей настройкой каналов:

канал 0 – в режим 1 (программируемый ждущий мультивибратор);
канал 1 – в режим 3 (генератор меандра).

1.5.3 Проектирование программы

Существует множество различных методов проектирования программ. Метод “Сверху вниз” или “Нисходящее проектирование” является одним из наиболее простых и понятных. Суть метода заключается в том, что решаемая задача последовательно разбивается на части, каждая из которых – на более мелкие (называемые модулями). Разбиение прекращается, когда содержание модуля не вызывает видимых затруднений для программной реализации. Вполне вероятно, что отдельные модули могут в дальнейшем потребовать дополнительно и более мелкого разбиения.

Детализация разбиения и размер модулей величина субъективная, но предпочтительно, чтобы модули были небольшими, что связано с особенностями зрительного восприятия человека, и, по возможности – независимыми, для оперативной замены при внесении изменений в программу.

1.5.4. Проектирование модулей

При правильном проектировании программы простота функций и логики работы каждого модуля позволяют сразу же записать код на языке ассемблера. В противном случае можно описать алгоритм работы модуля на “псевдокоде” (то есть, на языке, близком к естественному). Указатель комментария “ **;** ” позволяет реализовать такой подход в программах на ассемблере.

Правила “хорошего тона” в программировании предъявляют к оформлению и написанию отдельного модуля следующие требования:

а) наличие заголовка, включающего имя вызова модуля, краткое описание выполняемой функции, а также входных и выходных данных;
б) небольшой размер (для отладки в экранном режиме – не более 40 строк);
в) возвращать управление по месту вызова;
г) наличие одного входа и одного выхода;
д) возможность обращения к другим модулям (универсальность отдельных модулей);
е) по возможности меньшая взаимная зависимость (для оперативной замены);
ж) наличие необходимых комментариев по тексту программы.

Выполнение последнего требования безусловно необходимо при коммерческой разработке программ и передаче программного обеспечения заказчику. Тем не менее, выполнение этого требования облегчает работу и с личными программами, позволяя всякий раз не затрачивать время на чтение кода и выяснение алгоритмов.

Перечисленные требования несут в себе некоторые элементы структурного программирования и являются в большей мере желательными, но не обязательными. Программа, написанная без учета данных требований,

может быть вполне работоспособна, но трудоёмка в сопровождении и отладке.

1.5.5. Логическая структура ассемблерной программы

Элементы подпрограммы на языке ассемблера легче понять, если они расположены в той последовательности, в которой встречаются. Логической структуру ассемблерной программы в наиболее общем случае удобно представить в виде пяти вложенных в друг друга частей:

Уровень 1: общая надстройка ассемблера.
Уровень 2: надстройка ассемблерной подпрограммы.
Уровень 3: входной код.
Уровень 4: получение значений параметров вызывающей программы.
Уровень 5: выполнение кода программы, вызов служб ROM BIOS;
Уровень 4: передача результатов обратно вызывающей программе.
Уровень 3: выходной код.
Уровень 2: завершение надстройки ассемблерной подпрограммы.
Уровень 1: завершение общей надстройки ассемблера.

Данной базовой структуры можно придерживаться в большинстве интерфейсных программ, написанных для обращения к системным службам или просто в виде стандартных подпрограмм на ассемблере, однако следует иметь в виду, что конкретное кодирование будет видоизменяться в зависимости от языка программирования, используемого в вызывающей программе.

Для программ, написанных на языке ассемблера для процессора К580ВМ80 общий вид программы может быть примерно следующим:

```
;+-------------------------------------------------------+
;¦          ОСНОВНЫЕ ИНСТРУКЦИИ ПРОГРАММЕ АССЕМБЛЕР      ¦
;+-------------------------------------------------------+
;
;Col.1 _______________ - Поле меток и имен переменных
; ¦    Col.2 ___________ - Поле (псевдо)оператора (директивы)
; ¦     ¦    Col.3 _________ - Поле операндов
; ¦     ¦     ¦      Col.4 _____ - Поле комментариев
;+---+ +---+ +----+ +-----------------------------------+
        ORG  100H
;        ¦   +---+              256 (100H или другое количест-
;        ¦     +--------------- во байт)  резервируются  перед
;        ¦                      началом программы...
;        +--------------------- этой директивой, она же связы-
;                               вает  начало программы с абсо-
;                               лютным адресом 100H
```

```
;+------------------------------------------------------------+
;¦              Начало основной процедуры                     ¦
;+------------------------------------------------------------+
;
begin:
; ¦                              Метка (произвольна), сообщает
; +--------------------------- ассемблеру о начале выполнения
;                                программы с данного адреса или
;                                просто фиксирует начальный ад-
;                                рес (может отсутствовать).

;+--------------- Тело основной процедуры -----------------+
;¦
   ...
;¦
;+------------------------------------------------------------+
;
;+------------------------------------------------------------+
;¦           Объявление и инициализация данных                ¦
;+------------------------------------------------------------+
         DB  32H, 5BH; (Размер: 1 байт, байты)
         DW  0FFE1H;   (Размер: 1 слово)
;        ¦_____________________ псевдооператор резервирования
;
;+------------------------------------------------------------+
;--------------- Внешние метки и константы -----------------
;
;     +------------------------- псевдооператор присваивания
;     ¦
SYST:EQU 0F800H;____ выход в систему
KLAV:EQU 0F81BH;____ опрос клавиатуры без остановки программы
CNT0:EQU 0FFECH;____ адрес счетчика 0 ПТ
CNT1:EQU 0FFEDH;____ адрес счетчика 1 ПТ
CNT2:EQU 0FFEEH;____ адрес счетчика 2 ПТ
RGRJ:EQU 0FFEFH;____ адрес регистра режима ПТ

;+------------------------------------------------------------+
;¦                  Секция конца программы                    ¦
;+------------------------------------------------------------+
;
     END
;     ¦
;     +---------------------------- Директива конца основной
;                                    процедуры
```

Оформим рассматриваемую нами программу ввода в МПС данных от внешнего АЦП в виде ассемблерной программы с элементами псевдокода. Строки псевдокода становятся комментариями при переводе алгоритма на язык ассемблера.

```
;-------------------------------------------------------------;
;       Ввод данных АЦП в МПС с помощью таймера КР580ВИ53     ;
;-------------------------------------------------------------;
        ORG  0000H
START:
;______ Начальная инициализация ____________________________
        LXI  SP,7FFFH;    укажем стек программы
        MVI  A,0F0H;      байт управления: 1111.0000b
        STA  PRG;         выключаем таймер, запрещаем ввод
        CALL SET;         настроим таймер
        MVI  A,01FH;      код очистки экрана
        CALL SCR;         очистить экран, курсор в начало экр.
        XRA  A;           обнулим регистр A и [BC]
        MOV  C,A;         загрузим 0 в регистр C
        MOV  B,A;         загрузим 0 в регистр B
        CALL CON;         выводим индикацию, опрашиваем кнопки
        JZ   SYS;         выход, если нажата кнопка [Esc]

;______ Основной цикл чтения таймера и индикации ____________
CLK:
        CALL CNT;         считаем импульсы АЦП
        JZ   FIN;         ввод прерван нажатием кнопки [Esc]
        CALL CON;         выводим индикацию, опрашиваем кнопки
        JNZ  CLK;         если нет команды выхода -> в цикл
FIN:
        JMP  SYS;         выход в систему

;______ Считаем импульсы в регистровой паре [BC] ____________
CNT:
        MVI  A,0F1H;      байт управления: 1111.0001b
        STA  PRG;         фронт импульса разрешения ввода IE
        MVI  A,0F0H;      байт управления: 1111.0000b
        STA  PRG;         спад импульса разрешения ввода IE


;______ Цикл ожидания конца счета __________________________
WAIT:
        CALL KBD;         опрос клавиатуры без остановки
        CPI  1BH;         нажата ли кнопка [Esc]?
        JZ   EXIT;        да, нажата – выйти из подпрограммы
        LDA  PRG;         читаем порт ввода, анализируем
        RRC;              бит 0 – состояние GATE
        JC   WAIT;        если GATE = 1, ждем в цикле GATE = 0
READ:   LXI  H,RGR;       укажем на регистр режима таймера
```

```
        DCX  H;           укажем на канал_2
        MOV  A,M;         читаем младший байт остатка
        CMA;              инвертируем его и
        MOV  C,A;         загрузим в C
        MOV  A,M;         читаем старший байт остатка
        CMA;              инвертируем его и
        MOV  B,A;         загрузим в B
        XRA  A;           обнулим регистр A, флаг Z = 1
        DCR  A;           A = 0FFH, флаг Z = 0
        MOV  M,A;         восстановим константу счета
        MOV  M,A;         канала_0 = 0FFFFH
EXIT:
        RET

;_______Настройка таймера_____________________________________________
SET:
        PUSH H;       сохраним [HL]
        DI;           запретим прерывания
        LXI  H,RGR;   укажем на регистр режима таймера
;_____________________ Канал_2: прерывание терминального счета
        MVI  M,0B0H;  УС  К.2 = 0B0h = 10_11.000_0
;_____________________ Канал_1 настроим на меандр 01_11.011_0
        MVI  M,76H;   УС  К.1 = 76h - генератор меандра
;_____________________ Канал_0 стробирует канал_1 импульсом 1 мс
        MVI  M,36H;   УС  К.0 = 32h = 00_11.001_0 одновибратор

;--------------------------------------------------------------
        DCX  H;       укажем на канал_2 (константа 65535D)
        MVI  M,0FFH;  загрузим младший байт константы
        MVI  M,0FFH;  загрузим старший байт константы
;--------------------------------------------------------------
        DCX  H;       укажем на канал_1, вычислим делитель:
;-------------------- 2000000/128000 = 15,625 ~ 16
        MVI  M,10H;   загрузим младший байт делителя
        MVI  M,00H;   загрузим старший байт делителя
;--------------------------------------------------------------
        DCX  H;       укажем на канал_0, вычислим делитель:
;-------------------- 1/2000000 = 0.5 мкс - период внешнего сигнала
        MVI  M,0D0H;  делитель Кан.1: 1000 мкс / 0.5 мкс = 2000
        MVI  M,07H;   загрузим старший байт делителя
        POP  H;       восстановим [HL]
        EI;           разрешим все прерывания
        RET
;--------------------------------------------------------------
```

```
;______ Обработка консоли, [BC]- код для вывода_____________________
        MVI  A,0CH;   код позиционирования курсора
        CALL SCR;     курсор - в начало экрана
        MOV  A,B;     старший байт счетчика
        CALL HEX;     выводим на экран
        MOV  A,C;     младший байт счетчика
        CALL HEX;     выводим на экран
        CALL KBD;     опрос клавиатуры без остановки
        CPI  1BH;     нажата ли кнопка [Esc]?
        RET
TEST:  ;_______Подпрограмма самотестирования адаптера_________
        MVI  A,0F1H;      байт управления: 1111.0001b
        STA  PRG;         фронт импульса разрешения ввода IE
        MVI  A,0F0H;      байт управления: 1111.0000b
        STA  PRG;         спад импульса разрешения ввода IE
        MVI  A,0F4H;      байт управления: 1111.0100b
        STA  PRG;         имитация фронта сигнала GAP
        MVI  A,0F6H;      байт управления: 1111.0110b
        STA  PRG;         разрешение GATE канала 0 (бит 2)
        LXI  H,RGR-3;     укажем на счетчик канала 0
TST1:   MOV  B,M;         читаем младший байт счетчика 0
        MOV  A,M;         читаем старший байт счетчика 0
        ORA  A;           проверяем на равенство 0
        JNZ  TST1;        если не 0 – читаем счетчика снова
        MOV  A,B;         младший байт счетчика 0
        ORA  A;           проверяем на равенство 0
        JNZ  TST1;        если не 0 – читаем счетчика снова
        MVI  A,0F8H;      байт управления: 1111.1000b
        STA  PRG;         имитация фронта Write, сброс GATE
        MVI  A,0F0H;      байт управления: 1111.0000b
        STA  PRG;         имитация спада импульса Write
        JMP  READ;        читаем канал 2 (результат - 125)

;-- Внешние метки ------------------------------------
SYS:EQU  0F800H;___Холодный старт системы_______________
;___Код символа из A - в позицию курсора экрана_________
SCR:EQU  0F80FH;       в кодировке ASCII
;___Код символа из A - в позицию курсора экрана_________
HEX:EQU  0F815H;       в шестнадцатиричной форме
;___Опрос консоли без остановки программы_______________
KBD:EQU  0F81BH;       в A – код нажатой клавиши
PRG:EQU  0FFF9H;___Регистр управления: GATE0 и IE_______
RGR:EQU  0FFEFH;___Регистр режима таймера_______________
    END;_____________ укажем транслятору конец программы
```

2. ЭКСПЕРИМЕНТАЛЬНАЯ ЧАСТЬ

2.1. ПРОГРАММНО-АППАРАТНЫЕ СРЕДСТВА

Трехканальный счетчик–таймер применяется во всех моделях персональных компьютеров IBM PC. В ранних версиях XT в этом качестве использовалась микросхема i8253 (отечественный аналог – КР580ВИ53), в AT – более быстродействующая БИС i8254 (отечественный аналог – КР1810ВИ54), которая могла работать с процессорами i80286 без тактов ожидания. На современных системных платах функции счетчика–таймера берет на себя интегральный чипсет, сохраняя при этом полную программную совместимость как с i8253, так и с i8254. Поэтому большинство практических заданий легко реализуются в среде программного эмулятора при отображении портов таймера виртуальной машины К580ВМ80 на регистры системного таймера материнской платы.

В силу того что таймер на материнской плате компьютера имеет достаточно жесткую конфигурацию, для изучения более сложных схем включения таймера и режимов работы при тактировании его каналов сигналом внешнего источника может использоваться специальная плата расширения. Принципиальная схема интерфейсной платы, обеспечивающей подключение к системной шине компьютера IBM PC через слот расширения стандарта ISA программируемого таймера КР580ВИ53, а также двух или более параллельных периферийных адаптеров (ППА) представлена на рисунке 6.

Шинные формирователи DD2, DD3 увеличивает нагрузочную способность адресных (A0÷A11) и управляющих выводов шины ISA (**$\overline{\text{IOWR}}$**, **$\overline{\text{IORD}}$**, **AEN**, **RES**). На микросхемах DD5, DD9, на входы которых через шинные формирователи поданы сигналы A5÷A9 шины адреса компьютера, выполнен селектор адресов интерфейсной платы диапазона 300–31Fh, закрепленных стандартами IBM за устройствами пользователя или макетными платами. При выполнении компьютером команд чтения из портов с адресами от 300h до 31Fh или записи в эти же порты на выводе 8 DD9 формируется импульс низкого логического уровня, разрешающий работу микросхемы DD8. Дешифратор DD8, ко входам которого подключены буферированные сигналы A2÷A4 адресной шины, разбивает адресное пространство устройств пользователя на участки по 4 байта (300–303h ÷ 31C–31Fh), формируя на своих выходах сигналы выборки **$\overline{\text{CS}}$** низкого уровня при обращении по этим адресам.

Сигналы A0 и A1 подаются непосредственно на адресные входы программируемых БИС, осуществляя выборку внутренних регистров микросхем.

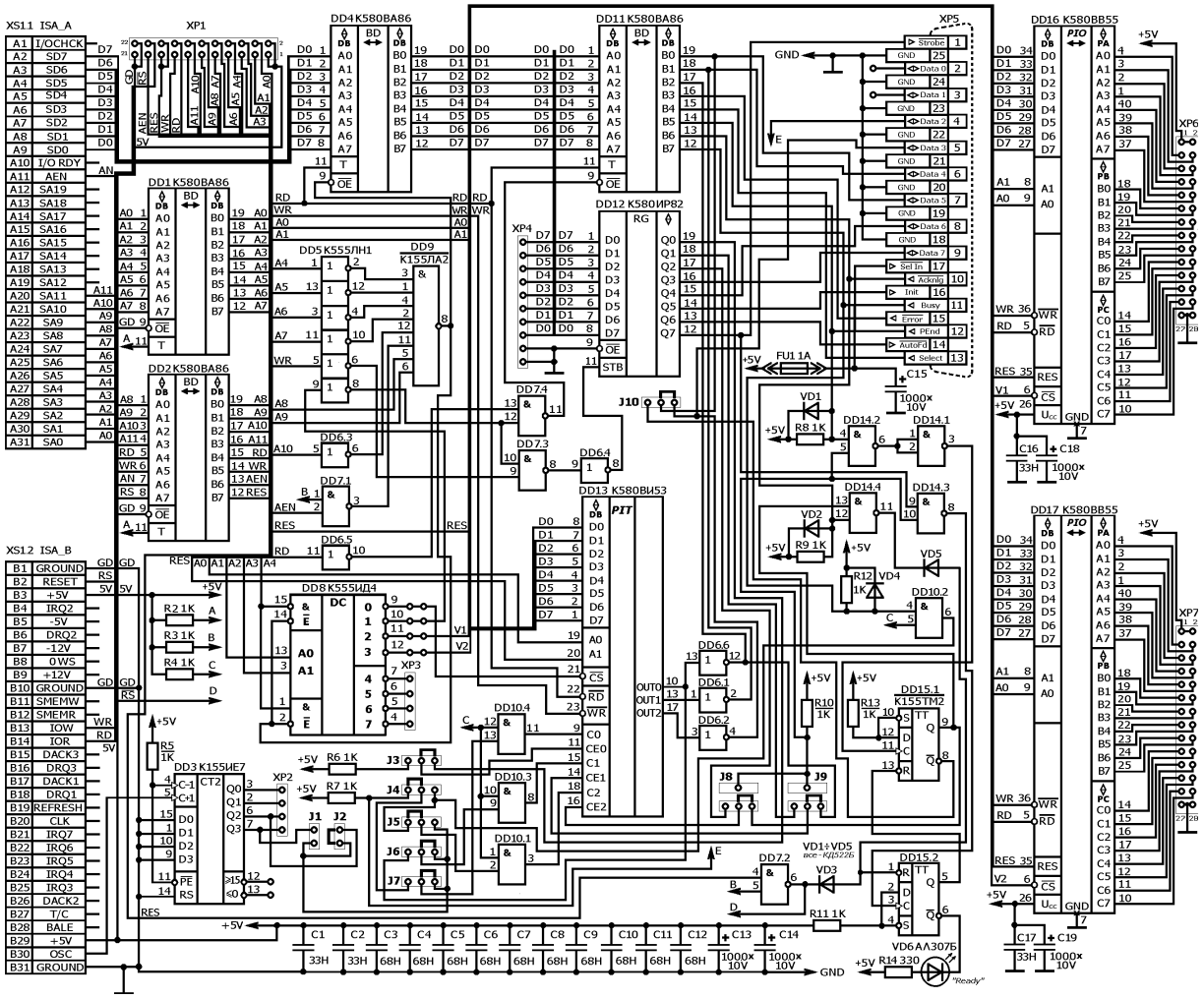


Рис. 6. Принципиальная схема платы расширения

Операции чтения или записи производятся по формируемым процессором компьютера сигналам **$\overline{\text{IOWR}}$** или **$\overline{\text{IORD}}$**. Для исключения сбоев на дешифратор DD5, DD9 подан сигнал AEN, блокирующий его при работе компьютера в режиме ПДП. Шинный формирователь DD4 увеличивает нагрузочную способность выводов данных (D0÷D7) программируемых БИС при работе на шину данных компьютера. Счетчик DD3 осуществляет деление тактовой частоты OSC, стандартной для всех системных плат IBM PC (14,31818 МГц), до величины, приемлемой для тактирования программируемого таймера (1,7897725 МГц – выв. 6 и 0,89488625 МГц – выв. 7). С помощью перемычек J1 и J2 можно выбрать необходимую частоту тактирования.

Буферный регистр DD12 позволяет программно формировать сигналы **GATE** для отдельных каналов таймера. Трехстабильный двунаправленный шинный формирователь DD11, доступный программно для чтения, используется для анализа состояния выходных сигналов таймера. Перемычки J3–J10 реализуют конфигурации включения БИС таймера в различные режимы: стробирование внутренним сигналом, стробирование внешним сигналом, последовательное включение каналов таймера, выбор источника

последовательное включение каналов таймера, выбор источника тактового сигнала и т.д.

Для соединения с внешними устройствами сигнальные цепи таймера и управляющих регистров выведены через разъём XP5 на заднюю панель компьютера, куда также выведен предохранитель цепи питания внешнего устройства и светодиодный датчик готовности схемы.

Согласно приведенной принципиальной схеме (рис. 6), внутренние регистры таймера DD13 доступны для записи/чтения в пространстве ввода/вывода компьютера IBM PC, как ячейки с адресами 300H–303H. Управляющий регистр DD12 доступен ***только для записи*** по любому из адресов 304H–307H, трехстабильный порт DD11 доступен ***только для чтения*** в этом же диапазоне адресов.

Программа — эмулятор, представляющая собой кросс–средство для выполнения команд процессора К580ВМ80 (i8080) на платформе x86, позволяет обратиться к регистрам таймера DD13 и управляющим портам DD11, DD12, как к ячейкам памяти по адресам 0FFECH ÷ 0FFEFH, 0FFF9H ÷ 0FFFBH соответственно. В пространстве адресов внешних устройств программа эмулятора рассматривает эти ячейки как порты таймера и резервные порты.

2.2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ

Задание 2.1

Реализовать генерацию пакетов импульсов с частотой 440 Гц, длительностью 0,5 сек и интервалом 0,5 сек между пакетами, используя программируемый интервальный таймер КР580ВИ53.

Регистры счетчиков таймера имеют следующие адреса в совмещенной карте памяти:

канал 0 – 0FFECH;
канал 1 – 0FFEDH;
канал 2 – 0FFEEH;
регистр режима каналов – 0FFEFH.

Доступное пользователю ОЗУ расположено по адресам 0000H÷7FFFH. Канал 1 настроить в режим генерации меандра с частотой 100 Гц.

На тактовые входы каналов 0 и 1 подаются импульсы с частотой 2 МГц (скважность 2), вход канала 2 подключен к выходу канала 1, выходы каналов 0 и 2 объединяются через логический элемент "или", выход которого управляет звуковым излучателем. На все входы разрешения счета поданы уровни логической единицы.

Для прекращения генерации использовать разряд 1 канала B ППА, установка которого в нулевое состояние через подключенную к нему замыкающую кнопку инициализирует выход в системный монитор по адресу 0F800H. Адрес канала B ППА в совмещенной карте памяти – 0FFE1H.

Задание 2.2

Реализовать электронный 4-х разрядный секундомер с подсчетом десятых и сотых долей секунды, используя программируемый интервальный таймер КР580ВИ53.

Регистры счетчиков таймера имеют следующие адреса в совмещенной карте памяти:

```
канал 0 – 0FFECH;
канал 1 – 0FFEDH;
канал 2 – 0FFEEH;
регистр режима каналов – 0FFEFH.
```

Доступное пользователю ОЗУ расположено по адресам 0000H÷7FFFH.

На тактовые входы каналов 0 и 1 подается импульсы с частотой 2 МГц (скважность 2), вход канала 2 подключен к выходу канала 1, выходы каналов 0 и 2 объединяются через логический элемент "или", выход которого управляет звуковым излучателем. На все входы разрешения счета поданы уровни логической единицы. Канал 1 настроить в режим генерации меандра с частотой 100 Гц.

Для начала и остановки счета использовать разряд 1 канала В ППА, установка которого в нулевое и единичное состояние осуществляется через подключенную к нему замыкающую кнопку. Адрес канала В ППА в совмещенной карте памяти – 0FFE1H.

Для вывода текущего состояния счетчика на дисплей использовать системную подпрограмму 0F818H, осуществляющую вывод строки символов на экран по следующему соглашению:

```
MT0:DB   1FH;  очистка экрана
MT1:DB   0CH;  курсор – в верхний левый угол экрана
    DB   20H;  пробел
MS3:DB   30H;  разряд 3 - десятки секунд,
MS2:DB   30H;  разряд 2 - секунды,
    DB   2EH;  разделительная точка,
MS1:DB   30H;  разряд 1 - десятые доли,
MS0:DB   30H;  разряд 0 - сотые доли,
    DB   00H;  признак конца строки,
```

где MT0 (MT1) – адрес, заносимый в регистровую пару [HL] перед вызовом подпрограммы 0F818H, MT0 – при начальном вызове и индикации 00.00d, MT1 – при обновлении показаний секундомера. Ячейки строки располагаются в произвольном месте ОЗУ пользователя. Числам от 0 до 9 соответствуют коды 30H – 39H.

Задание 2.3

Реализовать электронный метроном используя программируемый интервальный таймер К580ВИ53.

Регистры счетчиков таймера имеют следующие адреса в совмещенной карте памяти:

канал 0 – 0FFECH;
канал 1 – 0FFEDH;
канал 2 – 0FFEEH;
регистр режима каналов – 0FFEFH.

Доступное пользователю ОЗУ расположено по адресам 0000H÷7FFFH.

На тактовые входы каналов 0 и 1 подаются импульсы с частотой 2 МГц (скважность 2), вход канала 2 подключен к выходу канала 1, выходы каналов 0 и 2 объединяются через логический элемент "или", выход которого управляет звуковым излучателем. На все входы разрешения счета поданы уровни логической единицы.

Для запуска и остановки метронома использовать разряд 1 настроенного на ввод канала В порта ввода/вывода (ППА), установка которого в нулевое и единичное состояние осуществляется через подключенную к нему замыкающую кнопку.

Канал 0 использовать для задания звуковой частоты, канал 2 – для определения временного интервала. Канал 1 настроить в режим генерации меандра с частотой 100 Гц. Адрес канала В ППА в совмещенной карте памяти – 0FFE1H.

Временные интервалы выбрать произвольно, задавая их при помощи подпрограммы опроса клавиатуры 0C803H с ожиданием ввода, возвращающей код в регистре А микропроцессора. Коды 30H ÷ 39H соответствуют цифрам 0 ÷ 9, код 1BH использовать для выхода в системный монитор по адресу 0F800H.

Задание 2.4

Реализовать генерацию звука двухтональной сирены с частотами 440 Гц – 880 Гц и длительностью звучания каждого тона 0,5 сек, используя программируемый интервальный таймер КР580ВИ53.

Регистры счетчиков таймера имеют следующие адреса в совмещенной карте памяти:

канал 0 – 0FFECH;
канал 1 – 0FFEDH;
канал 2 – 0FFEEH;
регистр режима каналов – 0FFEFH.

Доступное пользователю ОЗУ расположено по адресам 0000H÷7FFFH.

На тактовые входы каналов 0 и 1 подаются импульсы с частотой 2 МГц (скважность 2), вход канала 2 подключен к выходу канала 1, выходы каналов 0 и 2 объединяются через логический элемент "или", выход которого управляет звуковым излучателем. На все входы разрешения счета поданы уровни логической единицы.

Канал 0 использовать для задания тона звуковой частоты, канал 2 –для задания временного интервала. Канал 1 настроить в режим генерации меандра с частотой 100 Гц.

Для прекращения генерации звука сирены использовать разряд 1 канала B ППА, установка которого в нулевое состояние через подключенную к нему замыкающую кнопку инициализирует выход в системный монитор по адресу 0F800H. Адрес канала B ППА в совмещенной карте памяти – 0FFE1H.

Задание 2.5

Двенадцати нотам второй октавы клавишных музыкальных инструментов соответствуют частоты: **261.6 Гц, 277.2 Гц, 293.6 Гц, 311.2 Гц, 329.6 Гц, 349.2 Гц, 370 Гц, 392 Гц, 415.2 Гц, 440 Гц, 466 Гц, 494 Гц**. Первой ноте следующей октавы соответствует частота **523.2 Гц**.

По нажатиям клавиш “Q”, “2”, “W”, “3”, “E”, “R”, “5”, “T”, “6”, “Y”, “7”, “U”, “I” имитировать работу клавишного музыкального инструмента, используя программируемый интервальный таймер КР580ВИ53.

Регистры счетчиков таймера имеют следующие адреса в совмещенной карте памяти:

канал 0 – 0FFECH;
канал 1 – 0FFEDH;
канал 2 – 0FFEEH;
регистр режима каналов – 0FFEFH.

Доступное пользователю ОЗУ расположено по адресам 0000H÷7FFFH.

На тактовые входы каналов 0 и 1 подаются импульсы с частотой 2 МГц (скважность 2), вход канала 2 подключен к выходу канала 1, выходы каналов 0 и 2 объединяются через логический элемент "или", выход которого управляет звуковым излучателем. На все входы разрешения счета поданы уровни логической единицы.

Канал 0 использовать для задания тона звуковой частоты, канал 2 – для задания временного интервала. Канал 1 настроить в режим генерации меандра с частотой 100 Гц.

В зависимости от выбранного алгоритма работы программы, для опроса клавиатуры можно использовать системные подпрограммы 0C803H и 0C81BH, возвращающие в регистре A микропроцессора код нажатой клавиши с ожиданием нажатия и без остановки программы соответственно. Если ни одна клавиша не нажата, подпрограмма 0C81BH возвращает в регистре A код 0FFH.

Клавишам “Q”, “2”, “W”, “3”, “E”, “R”, “5”, “T”, “6”, “Y”, “7”, “U”, “I” соответствуют коды: **51H, 32H, 57H, 33H, 45H, 52H, 35H 54H, 36H, 59H, 37H, 55H, 49H**.

По нажатию клавиши “Esc” (код – 1BH) осуществить выход в системный монитор по адресу 0F800H.

2.3. ПОРЯДОК ВЫПОЛНЕНИЯ ЗАДАНИЙ

1. Подготовить компьютер для выполнения соответствующего задания.
2. Ввести в редакторе подготовленные тексты программ, скомпилировать объектный код, отладить и выполнить программу. В общем случае правильность настройки программируемого таймера следует проверить при помощи осциллографа. Если он недоступен, следует в пошаговом режиме отладчика убедиться в правильности загрузки управляющих кодов и констант в соответствующие регистры таймера. В диапазоне звуковых частот ориентировочная оценка результата программирования осуществляется подключением к выходам таймера пьезоэлектрического излучателя или низкочастотного усилителя с высокоомным делителем на входе.
3. Оформить индивидуальный отчет, включающий необходимые иллюстративные материалы (функциональные схемы исследуемых подсистем, алгоритмы разработанных программ, временные диаграммы) и листинги отлаженных программ.

СПИСОК ЛИТЕРАТУРЫ

1. Гук М. Аппаратные средства IBM-PC. Энциклопедия, 2-е изд. – СПб.: Питер 2003. – 928 с.
2. Микропроцессоры и микропроцессорные комплекты интегральных микросхем. Под ред. В.А.Шахнова. – М.: "Радио и связь", 1988. Т.1, – 368 с.
3. Гутников В.С. Интегральная электроника в измерительных устройствах. – Ленинград.: "Энергоатомиздат", 1988. – 304 с.
4. Васильев Б.И., Гусев Ю.М., Миронов В.Н. Электронные промышленные устройства. – М.: "Высшая школа", 1988. – 304 с.
5. Сопряжение датчиков и устройств ввода данных с компьютерами IBM PC: Пер. с англ. / Под ред. У. Томпкинса и Дж. Уэбстера. – М.: Мир, 1992. – 592 с.
6. Нортон П., Соухэ Д. Язык ассемблера для IBM PC. – М.:"Компьютер", 1993. – 352 с.
7. Рудаков П.И., Финогенов К.Г. Программируем на языке ассемблера IBM PC: В 4-х частях. Ч.1. (Ч.2., Ч.3–4.) – М.:"Энтроп", 1995. – 164 с. (164 с., 320 с.).

ПРИЛОЖЕНИЕ

СИСТЕМА КОМАНД МИКРОПРОЦЕССОРА К580ВМ80

Флаги МП образуют регистр флагов (признаков) F: S.Z.0.AC.0.P.1.C
Условные обозначения: Z (ZERO) - флаг нулевого результата
S (SIGN) - флаг знака результата,
C (CARRY) - флаг переноса,
AC (AXIAL CARRY) - флаг осевого переноса,
P (PARITY) - флаг четности результата,
+ - команда изменяет флаг,
- - команда не изменяет флаг,
0 - команда устанавливает флаг в 0,
1 - команда устанавливает флаг в 1,
R - регистр общего назначения,
RP - регистровая пара,
 # - однобайтовый операнд,
- двухбайтовый операнд,
XX - адрес.

КОМАНДА	ДЛИНА (БАЙТ)	ФУНКЦИЯ	ПРИЗНАКИ				
1	2	3	Z	S	C	AC	P
MOV R,R	1	передача содержимого одного регистра в другой	-	-	-	-	-
MVI R,#	2	загрузка регистра вторым байтом команды	-	-	-	-	-
INR R	1	увеличить содержимое регистра на 1	+	+	-	+	+
DCR R	1	уменьшить содержимое регистра на 1	+	+	-	+	+
ADD R	1	к содержимому аккумулятора прибавить содержимое регистра	+	+	+	+	+
ADC R	1	к содержимому аккумулятора прибавить содержимое регистра с учетом флага переноса	+	+	+	+	+
SUB R	1	вычесть из аккумулятора содержимое регистра	+	+	+	+	+
SBB R	1	вычесть из аккумулятора содержимое регистра и флаг переноса	+	+	+	+	+
ANA R	1	логическое умножение аккумулятора и регистра	+	+	0	+	+
XRA R	1	выполнить операцию неравнозначности аккумулятора и регистра	+	+	0	0	+
ORA R	1	выполнить операцию логического сложения аккумулятора и регистра	+	+	0	0	+
CMP R	1	сравнить содержимое аккумулятора и регистра	+	+	+	+	+
ADI #	2	к содержимому аккумулятора прибавить второй байт команды	+	+	+	+	+
ACI #	2	к содержимому аккумулятора прибавить второй байт команды с учетом флага переноса	+	+	+	+	+

1	2	3	Z	S	C	AC	P
SUI #	2	из содержимого аккумулятора вычесть второй байт команды	+	+	+	+	+
SBI #	2	из содержимого аккумулятора вычесть второй байт команды с учетом флага переноса	+	+	+	+	+
ANI #	2	выполнить операцию логического умножения аккумулятора и второго байта команды	+	+	0	0	+
XRI #	2	выполнить операцию неравнозначности аккумулятора и второго байта команды	+	+	0	0	+
ORI #	2	выполнить логическое сложение аккумулятора и второго байта команды	+	+	0	0	+
CPI #	2	сравнить содержимое аккумулятора и второго байта команды	+	+	+	+	+
RLC	1	сдвинуть циклически влево содержимое аккумулятора	-	-	+	-	-
RRC	1	сдвинуть циклически вправо содержимое аккумулятора	-	-	+	-	-
RAL	1	сдвинуть циклически влево содержимое аккумулятора через флаг переноса	-	-	+	-	-
RAR	1	сдвинуть циклически вправо содержимое аккумулятора через флаг переноса	-	-	+	-	-
JMP XX	3	безусловный переход по указанному адресу	-	-	-	-	-
JC XX	3	условный переход при единичном состоянии флага переноса	-	-	-	-	-
JNC XX	3	условный переход при нулевом состоянии флага переноса	-	-	-	-	-
JZ XX	3	условный переход по нулевому значению результата	-	-	-	-	-
JNZ XX	3	условный переход по ненулевому значению результата	-	-	-	-	-
JP XX	3	условный переход по положительному значению результата	-	-	-	-	-
JM XX	3	условный переход по отрицательному значению результата	-	-	-	-	-
JPE XX	3	условный переход по четности кода результата	-	-	-	-	-
JPO XX	3	условный переход по нечетности кода результата	-	-	-	-	-
CALL XX	3	безусловный вызов подпрограммы по адресу XX	-	-	-	-	-
CC XX	3	вызов подпрограммы при единичном состоянии флага переноса	-	-	-	-	-
CP XX	3	вызов подпрограммы при положительном значении результата	-	-	-	-	-
CZ XX	3	вызов подпрограммы при нулевом значении результата	-	-	-	-	-

1	2	3	Z	S	C	AC	P
CNZ XX	3	вызов подпрограммы при ненулевом значении результата	-	-	-	-	-
CM XX	3	вызов подпрограммы при отрицательном значении рез-та	-	-	-	-	-
CPE XX	3	вызов подпрограммы по четности кода результата	-	-	-	-	-
CPO XX	3	вызов подпрограммы по нечетости кода результата	-	-	-	-	-
RET	1	безусловный возврат из подпрограммы	-	-	-	-	-
RC	1	возврат из подпрограммы по единичному состоянию флага переноса	-	-	-	-	-
RP	1	возврат из подпрограммы по положительному результату	-	-	-	-	-
RM	1	возврат из подпрограммы по отрицательному результату	-	-	-	-	-
RPE	1	возврат из подпрограммы по четности кода результата	-	-	-	-	-
RNC	1	возврат из подпрограммы по ненулевому состоянию флага переноса	-	-	-	-	-
RZ	1	возврат из подпрограммы по нулевому результату	-	-	-	-	-
RNZ	1	возврат из подпрограммы по ненулевому результату	-	-	-	-	-
RPO	1	возврат из подпрограммы по нечетности кода результата	-	-	-	-	-
RST N	1	безусловный вызов подпрограммы по фиксированному адресу (N*8)H	-	-	-	-	-
LXI RP,##	3	загрузка регистровой пары вторым и третьим байтами команды	-	-	-	-	-
PUSH RP	1	запись регистровой пары в стек	-	-	-	-	-
POP RP	1	загрузка регистровой пары из стека	-	-	-	-	-
POP PSW	1	загрузка аккумулятора и регистра признаков из стека	+	+	+	+	+
STA XX	3	запись содержимого аккумулятора по указанному адресу	-	-	-	-	-
LDA XX	3	загрузка аккумулятора содержимым ячейки с указанным адресом	-	-	-	-	-
XCHG	1	поменять местами содержимое регистровых пар [HL] и [DE]	-	-	-	-	-
XTHL	1	поменять местами содержимое верхней ячейки стека и [HL]	-	-	-	-	-
SPHL	1	загрузить в указатель стека [SP] содержимым [HL]	-	-	-	-	-
PCHL	1	загрузить программный счетчик [PC] содержимым [HL]	-	-	-	-	-

1	2	3	Z	S	C	AC	P
DAD RP	1	к содержимому [HL] прибавить содержимое другой регистровой пары или указателя стека и результат поместить в [HL]	-	-	+	-	-
STAX RP	1	записать содержимое аккумулятора по адресу, указанному в регистровой паре	-	-	-	-	-
LDAX RP	1	загрузить аккумулятор содержимым ячейки, адрес которой указан в регистровой паре	-	-	-	-	-
INX RP	1	увеличить на 1 содержимое регистровой пары или указателя стека	-	-	-	-	-
DCX RP	1	уменьшить на 1 содержимое регистровой пары или указателя стека	-	-	-	-	-
CMA	1	инвертировать содержимое аккумулятора	–	–	–	–	–
STC	1	установить флаг переноса в 1	-	-	1	-	-
CMC	1	инвертировать флаг переноса	-	-	+	-	-
DAA	1	десятичная коррекция результата	+	+	+	+	+
SHLD XX	3	записать содержимое [HL] в две ячейки последовательно с указанного адреса: в первую- L, во вторую - H	-	-	-	-	-
LHLD XX	3	загрузить пару [HL] содержимым двух последовательных ячеек с указанного адреса,из младшей - L, из старшей - H	-	-	-	-	-
NOP	1	пустая операция	-	-	-	-	-
HLT	1	останов процессора	-	-	-	-	-
IN #	2	загрузка аккумулятора содержимым порта с номером #	-	-	-	-	-
OUT #	2	запись содержимого аккумулятора в порт с номером #	-	-	-	-	-
DI	1	запретить прерывания	-	-	-	-	-
EI	1	разрешить прерывания	-	-	-	-	-

Учебное издание

Семёнов Андрей Андреевич

**ИЗУЧЕНИЕ БИС
ПРОГРАММИРУЕМОГО
ИНТЕРВАЛЬНОГО
ТАЙМЕРА**

Учебное пособие
для студентов факультета компьютерных наук
и информационных технологий
и факультета нано- и биомедицинских технологий

Редактор В.А. Т р у ш и н а
Технический редактор Л.В. А г а л ь ц о в а
Корректор Ю.И. А с т а х о в а

Подписано в печать _.__.2006.
Формат 60×84 1/16. Бумага офсетная. Гарнитура Times. Печать офсетная.
Усл.печ.л.1,16(1,25). Уч.-изд.л.0,9. Тираж 150 экз. Заказ

Издательство Саратовского университета.
410012, Саратов, Астраханская, 83.
Типография Издательства Саратовского университета.
410012, Саратов, Астраханская, 83.