**Частина 2**

**ВСТУП**

При изучении систем ввода-вывода и интерфейсов необходимо представлять основные принципы построения средств вычислительной техники, которые в основном определяются той элементной базой, на которой строятся компьютеры. С этой точки зрения весь период развития вычислительной техники от первого компьютера до современных вычислительных машин можно разбить на два этапа. Первый – это этап до появления современных интегральных схем и микропроцессоров. Второй – после их появления и начала выпуска персональных компьютеров (ПК).

На первом этапе компьютеры разрабатывались и изготовлялись на своей собственной элементной базе, их устройства (процессор, ОЗУ, устройства управления и т.п.) имели архитектуру и структуру, присущую только данному компьютеру, связь между устройствами и узлами осуществлялась с помощью интерфейсов, используемых только этим типом вычислительной машины. Стандарты применялись, но в основном касались ПУ, а не внутренних устройств. Поэтому машины разных фирм были не совместимы по элементной базе, устройствам и конструктиву. Это относилось как к большим, так и к малым вычислительным машинам. Это машины типа IBM 360

(370) , DEC PDP-11, БЭСМ-1 (2,6), ЕС ЭВМ-1033 (1040, 1060), СМ-1 (2,3,4), «Урал», «Наири» и т.д.

Каждый тип компьютера был в определенной степени уникален.

На втором этапе изменился принцип построения вычислительной техники. Она стала основываться на правиле трех «М»: модульность, микропрограммируемость и магистральность. Модуль представляет из себя функционально полное и конструктивно законченное устройство, серийно выпускаемое и программно (микропрограммно) управляемое (настраиваемое). Компьютеры собираются на основе этих модулей с помощью стандартных каналов связи – интерфейсов. В этом случае модули (устройства) выпускают одни фирмы, а изготовляют компьютер – другие. Модули конструктивно и функционально разнообразны, но совместимы по своим интерфейсам. Модули выпускаются в виде микросхем различного типа (МП, память, ChipSet), съемных плат (материнская плата, платы расширения), различных типов ПУ. Функционально они соответствуют тем устройствам, которые реализуют преобразование, хранение и передачу информации: процессор, ОЗУ, схема управления, устройства ввода и вывода и т.п.

На втором этапе существенно возросла роль стандартизации, без которой разработка и выпуск компьютеров стал невозможен. Роль и значение систем ввода-вывода и интерфейсов в последнее время существенно возросла. Это связано, прежде всего, с быстрым ростом производительности микропроцессоров.

Частота их работы имеет значение в сотни мегагерц (Pentium III), увеличивается их разрядность: 32, 64, 128, 256 бита. Растет емкость оперативных запоминающих устройств, достигая сотен мегабайт и гигабайт. Уменьшается время доступа ОЗУ, составляя единицы и доли наносекунд. Увеличивается количество ПУ, подключаемых к компьютеру, растет их скорость работы. Емкость модуля жестких магнитных дисков уже составляет десяток гигабайт. Увеличивается объем графической информации, выводимый на монитор, широко используется 3- мерная графика, «живое видео».

Все это требует соответствующего увеличения скорости передачи информации до сотен мегабайт в секунду и более и такой организации взаимосвязи устройств в вычислительной системе, которая бы исключала потери процессорного времени и равномерно загружала бы работой все устройства компьютера.

Учебное пособие посвящено рассмотрению роли и места систем ввода-вывода и интерфейсов в вычислительных системах, изложению принципов их построения и функционирования. Оно содержит материалы о наиболее широко используемых интерфейсах ISA, PCI, AGP, SCSI,USB, а также компьютерные лабораторные работы, облегчающие и помогающие изучению этих материалов.

Учебное пособие предназначено для студентов, специализирующихся в области вычислительной техники и программирования. Знание интерфейсов позволит разработчику аппаратуры более грамотно подойти к выбору варианта, соответствующего поставленной задаче. Знания материалов этого пособия необходимы системным интеграторам. Без этих знаний они не смогут подобрать требуемый набор устройств и оптимально объединить их в систему. Сведенияучебного пособия помогут системным программистам при разработке собственных драйверов ПУ или адаптировать чужие разработки.

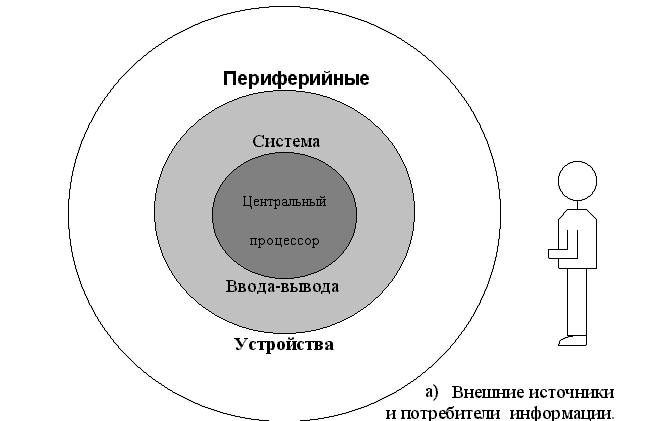
В основе учебного пособия лежат курсы лекций «Компьютерные коммуникации и периферия», «Система ввода-вывода и интерфейсы», читаемые студентам дневных и вечерних форм обучения МИФИ, а также материалы книги Гук М. «Интерфейсы ПК»: Справочник – С.-П.: Питер-

**1. ОСНОВНЫЕ ПРИНЦИПЫ ПОСТРОЕНИЯ СИСТЕМ ВВОДА-**

**ВЫВОДА И ИНТЕРФЕЙСОВ**

**1.1. РОЛЬ И МЕСТО СИСТЕМ ВВОДА-ВЫВОДА И ИНТЕРФЕЙСОВ В КОМПЬЮТЕРЕ**

Компьютер выполняет три основные функции: преобразование, хранение и передачу информации. В соответствии с этим компьютер условно делится на три части. Процессор, который предназначен для преобразования информации. Память, выполняющую функцию хранения информации. Процессор и оперативную память обычно называют центральным процессором (ЦПр). Средства передачи информации, которые целесообразно разделить на две части: внутренние – систему ввода-вывода и внешние – периферийные устройства (см. рисунок 1.1а).





Такое разделение средств передачи информации позволяет выделить ту часть вычислительной системы, которая реализует процессы передачи данных и в значительной степени определяет производительность работы компьютера, т.е. определяет его качество.

Система ввода-вывода предназначена для обеспечения передачи информации между электронными устройствами при реализации функций преобразования и хранения данных и, главное, для связи этих устройств с периферийными устройствами. Система ввода-вывода реализуется с помощью соответствующих аппаратных и программных средств.

Периферийные устройства (ПУ) предназначены для передачи информации между компьютером и внешним миром (см. рисунок 1.1а). Их основное назначение заключается в преобразовании многообразия форм представления информации во внешнем мире в электрические сигналы, воспринимаемые устройствами компьютера и обратного преобразования электронного представления информации в формы понятные объектам внешнего мира. ПУ связываются с центральным процессором посредством системы ввода-вывода.

Часто процессы и устройства ввода-вывода сводят к понятию «периферия», в результате из рассмотрения исчезают вопросы организации взаимосвязи устройств в компьютере и роль этой взаимосвязи при оценке его производительности.

Вопросам организации ввода-вывода информации в вычислительной системе иногда оказывают недостаточное внимание. Это приводит к тому, что при оценке производительности системы часто используют только оценку производительности центрального процессора, а оценкой системы ввода-вывода пренебрегают.

Это противоречит здравому смыслу. Очевидно, одной из наиболее правильных оценок производительности системы является время ответа (время между моментом ввода задания и получения результата), которое учитывает все накладные расходы, связанные с выполнением задания в системе, включая процессы передачи информации, т.е. ввод-вывод.

О важности системы ввода-вывода говорит также и то, что в настоящее время стоимость и производительность компьютеров различного ценового класса от персональных компьютеров до суперкомпьютеров (суперсерверов) определяется в основном организацией систем памяти и ввода-вывода (а также количеством процессоров).

Производительность процессоров растет со скоростью 50-100% в год. Если одновременно не улучшались бы характеристики систем ввода/вы - вода, то очевидно разработка новых систем зашла бы в тупик.

Функции компьютера, описанные выше, реализуются с помощью устройств, которые создаются на базе различных по своему конструктивному выполнению, стандартных, серийно выпускаемых изделий, которые будем для общности рассмотрения называть модулями. Таким образом, модуль – это функционально полное и конструктивно законченное устройство.

Примерами таких модулей могут быть следующие.

Интегральные микросхемы (микропроцессоры, микросхемы управления, адаптеры, микросхема памяти и т.п.).

Внешние запоминающие устройства (ВЗУ), выпускаемые в виде модулей жестких и гибких магнитных дисков, оптических и магнитооптических дисководов.

Модули периферийных устройств (см. рисунок 1.1б): мониторы, клавиатуры, печатающие устройства, сканеры и т.п.

В качестве отдельных модулей используются платы расширения, на которых располагаются специальные микросхемы и разъемы для сопряжения с другими модулями компьютера.



Основой компьютера является плата, на которой расположены МП, память, основная часть устройств системы ввода-вывода, такую плату называют системной или материнской платой. При изготовлении компьютера модули соединяются между собой с помощью стандартных конструктивных средств. Следует подчеркнуть, что модули по скорости работы очень разнородны, время выполнения ими операций лежит в очень широких пределах – от долей секунды до наносекунд, т.е. меняется на 8 порядков. Если эту скорость измерять временем обработки одного слова, т.е. временем его преобразования, временем доступа к памяти, временем его передачи, временем ввода или вывода, то получим следующие цифры: микропроцессор – от 1 нсек до нескольких наносекунд, ОЗУ – от нескольких до десятков наносекунд, ВЗУ – от сотен наносекунд до сотен микросекунд, Устройства ввода-вывода – от сотен микросекунд до долей секунд.

В процессе работы компьютера идет постоянная передача информации между модулями. И время передачи информации от модуля к модулю существенно влияет на производительность работы компьютера, а иногда определяет ее.

Огромная разница в скоростях работы взаимосоединяемых модулей создает серьезные трудности для систем ввода-вывода и требует выработки определенных оптимальных решений, обеспечивающих минимизацию простоев центрального процессора и полной загрузки работой других устройств при функционировании вычислительной системы.

Конструктивно компьютер, как правило, представляет из себя системный блок, выполненный в виде отдельного корпуса, где располагается системная плата, и нескольких периферийных устройств, подключенных к системному блоку с помощью специальных кабелей и шнуров.

Компоновка компьютера, т.е. соединение всех модулей между собой, производится с учетом выполнения трех главных требований:

 Обеспечения максимальной производительности работы компьютера в целом при малой

стоимости с учетом времени передачи информации.

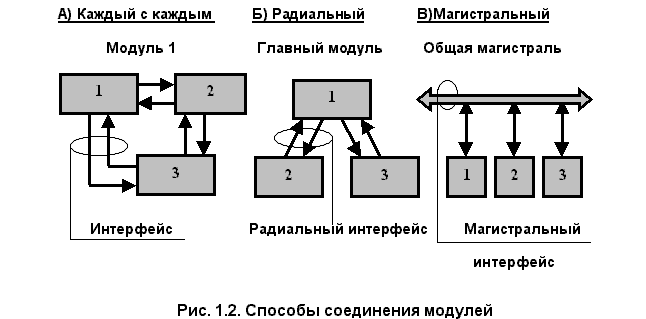
 Конструктивной простоты и минимальной стоимости средств взаимосоединений модулей.

 Возможности подключения максимального числа ПУ и простота изменения конфигурации

ПУ в период эксплуатации компьютера.

Рассмотрим, как реализуется взаимосвязь модулей, с учетом вышеперечисленных требований. Имеются три основных способа соединения модулей между собой (см. рисунок 1.2):

«каждый с каждым», радиальный и магистральный.



При соединении типа «каждый с каждым» взаимосвязь модулей ни чем не ограничена из-за огромного количества линий связи, поэтому легко выполняются требования высокой производительности, но остальные требования реализовать нельзя из-за сложности взаимных соединений. Поэтому этот способ на уровне модулей не используется.

Радиальный – при этом способе имеется главный модуль, с которым связаны все остальные и они могут взаимодействовать между собой только через главный модуль, что также снижает производительность, но меньше чем при магистральном способе. Такой способ соединения эффективен для соединения модулей, которые в основном работают только с главным. Достоинство – простота каналов связи каждого модуля, что позволяет увеличивать их длину на много больше, чем у магистральных. Радиальный способ позволяет к одному разъему подключать всего одно ПУ или, при соответствующей организации канала, несколько ПУ. Этот способ также нашел широкое применение в компьютерах.

Магистральный – это способ, когда все модули взаимодействуют через общую магистраль. Он является наиболее простым по реализации, но снижает производительность, т.к. все модули делят магистраль между собой. К недостатком этого способа можно отнести также и то, что высокоскоростные магистрали все же дороги и могут быть реализованы только при их ограниченной длине. При определенной организации взаимосвязи, магистральный способ позволяет к одному разъему подсоединить несколько ПУ. Этот способ наиболее широко используется в компьютерах.

Взаимосоединение модулей в компьютере, с помощью рассмотренных выше способов, возможно только при определенной унификации этих соединений. Средства и правила, обеспечивающие взаимосвязь модулей между собой, называются интерфейсом.

Главное требование к интерфейсам – их стандартизация. Только использование стандартных интерфейсов может обеспечить массовый выпуск компьютеров и их эффективное производство и эксплуатацию. Именно интерфейсы определяют те правила, по которым строятся каналы передачи информации между модулями вычислительной системы.

Физически интерфейс реализуется в виде электрических линий для передачи сигналов и набора микросхем, обеспечивающих выполнение основных функций интерфейса.

В интерфейсе стандартизируются следующие параметры:

 Структура интерфейса, т.е. количество и назначение линий интерфейса.

 Параметры электрических сигналов в линиях.

 Протоколы обмена информацией в интерфейсах, циклы (команды) интерфейса, реализуемые в виде временных диаграмм сигналов, зависящих от архитектуры и структуры интерфейса.

 Конструктивные параметры интерфейса.

 Таким образом, стандартный интерфейс – это совокупность унифицированных аппаратных, программных и конструктивных средств и правил, необходимых для реализации взаимодействия различных функциональных элементов в автоматических системах сбора и обработки информации.

1**.2. ОСНОВНЫЕ ПРИНЦИПЫ ОРГАНИЗАЦИИ ПЕРЕДАЧИ ИНФОРМАЦИИ В ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ**

В процессе работы компьютера передача информации по одному и тому же интерфейсу в один и тот же момент времени идет только между двумя устройствами (модулями) по принципу «точка-точка». При этом одно из устройств является активным (ведущим, задатчиком), другое – пассивным (исполнителем, ведомым). Активное устройство начинает процедуру обмена и управляет ею. Пассивное устройство выполняет предписания активного. В компьютере одни устройства всегда является задатчиками (активными), другие только исполнителями (пассивными), третьи в разные моменты времени могут быть как задатчиками, так и исполнителями. Процессор всегда активное устройство, оперативная память (ОЗУ) – пассивное устройство. Периферийные устройства при работе с процессором являются исполнителями, а при работе с ОЗУ (прямой доступ к памяти) – задатчиками. Возможна передача между двумя ПУ, тогда одно – задатчик, другое – исполнитель (см. рисунок 1.3а).



Таким образом, пассивными устройствами являются либо ОЗУ, либо ПУ. Со стороны процессора средства управления этими двумя типами устройств существенно различны. Это обусловлено тем, что для передач процессор-ОЗУ заранее известны все типы и параметры устройств, которые должны соединяться между собой, т.к. эти устройства однотипны, в то время как ПУ существенно различаются, как по задержке, так и по пропускной способности. Кроме того, процесс управления ПУ намного сложнее и требует большего времени и учета специфики работы.

Поэтому управление передачей процессор-ОЗУ реализуется в рамках одной компьютерной команды на уровне микрокоманд (см. рисунок 1.3б), а управление процессом ввода-вывода с учетом специфики ПУ с помощью специальной подпрограммы, которая называется драйвером и содержит как команды компьютера, так и команды управления, специфичные для каждого типа ПУ (см. рисунок 1.3в). Поэтому интерфейс, связывающий устройства при передаче данных должен учитывать эти особенности.

Передача информации от задатчика к исполнителю реализуется операцией записи, а обратная – операцией чтения. Процесс передачи между ПУ и ЦПр называют вводом-выводом информации.

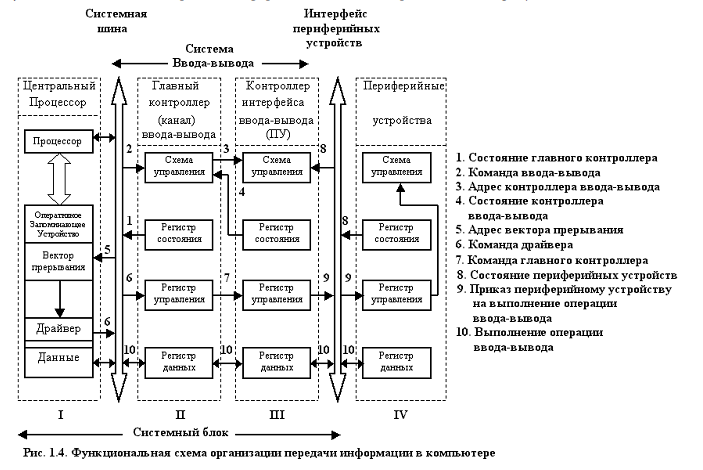
Ввод реализуется с помощью операции чтения, а вывод – операцией записи. Если на процессор возложить функции управления вводом-выводом, то у него не хватит времени для выполнения своей главной функции – преобразования информации. Это обусловлено широким диапазоном скоростей работы ПУ, сложностью их управления и большим разнообразием и количеством разных ПУ (см. рис.1.1б).

Для минимизации загрузки процессора при выполнении процессов ввода-вывода функции управления распределяются на несколько устройств: процессор, главный контроллер (канал) ввода-вывода, контроллер ПУ и блок управления ПУ.

Главный контроллер и контроллеры различного типа ПУ вместе с соответствующим программным обеспечением образуют Систему ввода-вывода.

Процессор, главный контроллер, контроллеры ПУ находятся в системном блоке, а блок управления ПУ – в самом периферийном устройстве, в ряде случаев контроллер ПУ может находиться в самом ПУ. Такое распределение позволяет оптимизировать загрузку каждого вида устройств присущими ему функциями при мультипрограммном режиме работы компьютера. Это происходит за счет совмещения во времени работы различных по функциям устройств.

Функциональная схема передачи информации в компьютере показана на рисунке 1.4.



Операцию ввода-вывода начинает процессор, выдавая в главный контроллер ввода-вывода соответствующую команду (2). Предварительно, с помощью слова состояния канала, процессор должен убедиться, что он готов к работе (1).

Канал приступает к управлению вводом-выводом с помощью специальной программы канала. После чего процессор отключается от процесса ввода-вывода и функции управления принимает на себя канал.

Канал, после анализа слова состояния (4) соответствующего контроллера о его готовности, передает ему управляющий приказ (команду канала) (6,7) и может переходить к работе с контроллером другого ПУ. Контроллер ПУ принимает эстафету управления вводом-выводом,формирует и передает в блок управления ПУ необходимый управляющий код (сигнал) (8,9), получив который ПУ преступает к процессу передачи данных (10). Как следует из рассмотрения (см. рисунок 1.4), каждый уровень управления вводом-выводом должен иметь регистры управления, состояния и данных для реализации своих действий по передаче информации.

При реализации такой многоуровневой системы управления вводом-выводом для связи устройств между собой используется два вида интерфейсов: системные интерфейсы (внутренние) для взаимосвязи модулей в системном блоке и внешние интерфейсы ПУ (интерфейс периферийных устройств) для подсоединения ПУ к системному блоку (см. рисунок 1.4).

Для учета особенностей реализации процессов ввода-вывода и специфики различного типа ПУ используются три режима ввода-вывода информации: программный ввод-вывод, ввод-вывод в режиме прерываний и с прямым доступом к памяти.

Интерфейсы должны учитывать возможность реализации всех 3-х режимов ввода-вывода.

**1.3. КОМПЬЮТЕРНЫЕ КОММУНИКАЦИИ И ИНТЕРФЕЙСЫ**

Интерфейсы, используемые при построении вычислительных систем, весьма разнообразны и различаются между собой по определенным критериям и характеризуются рядом свойств и параметров. В области интерфейсов существует своя терминология, которая задается теми или иными определениями и понятиями, которые с развитием компьютерной техники постоянно трансформируются, что вызывает необходимость рассматривать их в определенной ретроспективе. Кратко остановимся на некоторых понятиях и определениях.

Канал – среда передачи информации, представляемой в виде определенных сигналов. Канал реализуется с помощью тех или иных средств, зависящих от физической природы сигналов (ток, напряжение, радиосигнал, оптический сигнал и т.п.). Компьютерный интерфейс является каналом передачи информации с помощью электрических сигналов (импульсов и потенциалов).

Линия интерфейса – это электрический проводник (провод, линия печатного монтажа, контакт разъема платы), по которому распространяется электрический сигнал. Магистраль – это совокупность всех линий интерфейса.

Шина – группа линий интерфейса, соответствующая определенному функциональному назначению (шина данных, шина адреса и т.п.)

В дальнейшем термин «магистраль» стали заменять термином «шина». В свою очередь понятие «шина» стало в определенной мере синонимом термина «интерфейс», хотя понятие шины более узкое, чем общее понятие интерфейса.

В компьютерах, по причинам, изложенным выше, система ввода-вывода строится на основе магистральных и радиальных способов соединения модулей (устройств), оформленных в виде стандартных интерфейсов, называемых соответственно магистральными и радиальными. По способу передачи информации интерфейсы делятся на параллельные и последовательные.

Разряды данных могут передаваться в интерфейсах одновременно, т.е. параллельно. Такие интерфейсы называются параллельными, и они имеют шину данных из стольких линий, сколько разрядов передается одновременно. При передаче данных по одной линии последовательно разряд за разрядом, интерфейс называют последовательным.

Кажется очевидным, что при одной и той же скорости работы линий интерфейса, пропускная способность параллельного интерфейса выше, чем у последовательного. Однако повышение производительности за счет увеличения тактовой частоты передачи и количества линий данных упирается в волновые свойства соединительных кабелей. Задержка сигналов в различных линиях не одинакова, и это особенно сказывается при увеличении длины линий, что требует для надежной передачи данных дополнительных временных и аппаратных затрат, сдерживая этим рост пропускной способности параллельного интерфейса. Кроме того, в параллельных интерфейсах с увеличением числа параллельных линий и их длины труднее реализовать компенсацию помех, наводимых за счет электрического взаимодействия линий между собой.

В последовательных интерфейсах есть свои проблемы повышения производительности, но т.к. в них используется меньшее число линий, повышение пропускной способности канала связи обходится дешевле.

Поэтому важным параметром интерфейсов является допустимое удаление соединяемых устройств. Оно определяется как частотными свойствами, так и помехозащищенностью используемых каналов связи.

Для интерфейса, соединяющего два устройства (модуля), различаются три возможных режима обмена: дуплексный, полудуплексный и симплексный. Дуплексный режим позволяет по одному каналу связи, но имеющему две группы линий «туда» и «обратно», одновременно передавать информацию в обоих направлениях. Он может быть асимметричным, если пропускная способность в направлении «туда» и «обратно» имеет существенно различающееся значения, или симметричным. Полудуплексный режим позволяет передавать информацию по одним и тем же линиям «туда» и «обратно» поочередно в разные моменты времени, при этом интерфейс имеет средства переключения направлений канала.

Симплексный (односторонний) режим предусматривает только одно направление передачи информации (во встречном направлении могут передаваться только вспомогательные сигналы интерфейса).

Все модули внутри системного блока, а это электронные модули (микропроцессор, оперативная память, контроллеры ввода-вывода и т.д.), соединяются между собой с помощью так называемого внутреннего интерфейса, являющегося параллельным и построенного по магистральному способу. Внутренние интерфейсы предназначены для быстрой связи на короткие расстояния. Причем важна не только высокая пропускная способность, но и малое время доступа активного устройства к исполнителю.

ПУ соединяются с системным блоком по радиальным или магистральным интерфейсам, последовательным и параллельным, называемым внешними. Разъем системного блока, предназначенный для подсоединения к нему внешнего интерфейса и соответствующие ему электронные схемы, называют портом ввода-вывода.

Так как через внутренний интерфейс соединяются в систему все модули компьютера, этот интерфейс стали называть системным интерфейсом (системной шиной). Хотя в дальнейшем это понятие несколько изменилось, о чем речь пойдет ниже. Внешние интерфейсы стали называть малыми интерфейсами или интерфейсами периферийных устройств (см. рисунок 1.4).

Важным отличительным свойством интерфейсов является топология соединения модулей (устройств) с шиной интерфейса.

Шины внутренних интерфейсов, являясь параллельными и магистральными, размещаются на системной (материнской) плате, а взаимосоединяемые модули являются электронными, изготовленными в виде микросхем или плат расширения (процессор, ОЗУ, ROMBIOS и т.п.), поэтому каждый модуль имеет свое собственное соединение с шиной. Это соединение реализуется разными способами: печатным монтажем, с помощью слотов, сокетов и разъемов. Причем допустимое удаление модулей друг от друга мало и определяется размерами материнской платы.

Слот (Slot) представляет собой щелевой разъем, в который вставляется какая-либо печатная плата. Слот расширения (Expansion Slot) - это разъем системной шины, куда вставляется плата или карта расширения (Expansion Card).

Платой (картой) расширения называют печатную плату с краевым разъемом, устанавливаемую в слот расширения, платы, обеспечивающие дополнительный интерфейс, называют интерфейсными картами (Interface Card)

Сокет (Socket) – гнездо, в которое устанавливается микросхема.

Сокет (Socket) – гнездо, в которое устанавливается микросхема.

Внешние интерфейсы ПУ имеют более разнообразную топологию. Для радиальных интерфейсов – это двухточечная топология: порт системного блока – разъем ПУ (шины RS-232,Centronics) или топология типа дерева (шины USB). Двухточечная топология позволяет к порту подключать только одно ПУ. При древовидной топологии основой (корнем) дерева является порт ввода-вывода компьютера, а ПУ могут быть либо конечным устройством, либо новым центром ветвления. Эта топология позволяет подключать множество устройств к одному порту компьютера.

Для магистральных интерфейсов используется шлейфовая топология (шины SCSI). При шлейфовой топологии шина подключается к одному порту компьютера, а от него последовательно проходит по всем ПУ. В ПУ один разъем подключается к предыдущему ПУ (первое – к порту ввода-вывода), а другой разъем шины используется для подключения следующего ПУ, если его нет, то к этому разъему подключается терминатор (согласующая схема). В этом случае, как правило, контроллер внешнего интерфейса размещается в самом ПУ, а шиной SCSI управляет главный контроллер (адаптер), который фактически является каналом ввода-вывода (см. рисунок 1.4).

**1.4. СИСТЕМНЫЕ ИНТЕРФЕЙСЫ И ШИНЫ РАСШИРЕНИЯ**

Основой высокой производительности вычислительной машины любого типа является центральный процессор (ЦПр), состоящий из микропроцессора (МП) и оперативной памяти, связанных шиной передачи информации. Из этих трех составляющих определяющим является МП, т.к. он преобразует информацию и управляет передачей данных. Процессор – активное устройство, а ОЗУ и шина – пассивные. Если обозначить скорость работы МП – UМП, ОЗУ – UОЗУ, а шины – UШ, то наилучшим (оптимальным) соотношением скоростей будет UМП =UОЗУ=UШ. Пропускная способность шины зависит от ее разрядности, частоты работы и числа синхроимпульсов на передачу одного слова. Поэтому внутренние шины делят на шины, обеспечивающие связь процессора с памятью и шины ввода-вывода. Шины процессор-память сравнительно короткие, обычно высокоскоростные и соответствуют организации системы памяти для обеспечения максимальной пропускной способности канала память-процессор. Такие шины стали называть системными шинами, а бывшие системные - шинами ввода-вывода. Шины ввода-вывода длиннее и могут поддерживать подсоединение многих типов устройств и обычно следуют одному из стандартов. Шины ввода-вывода стали называть шинами расширения, подразумевая то, что эта шина как бы расширяет системную шину с точки зрения количества подключаемых устройств. ПУ подсоединяют к шине ввода-вывода посредством интерфейсов ПУ, которые, как правило, стандартизированы.

Шинам процессор-память заранее известны все типы и параметры устройств, которые должны соединяться между собой, шина ввода-вывода имеет дело с устройствами, различающимися по задержке и пропускной способности.

Под пропускной способностью понимают скорость передачи по шине, измеряемой количеством байт в секунду.

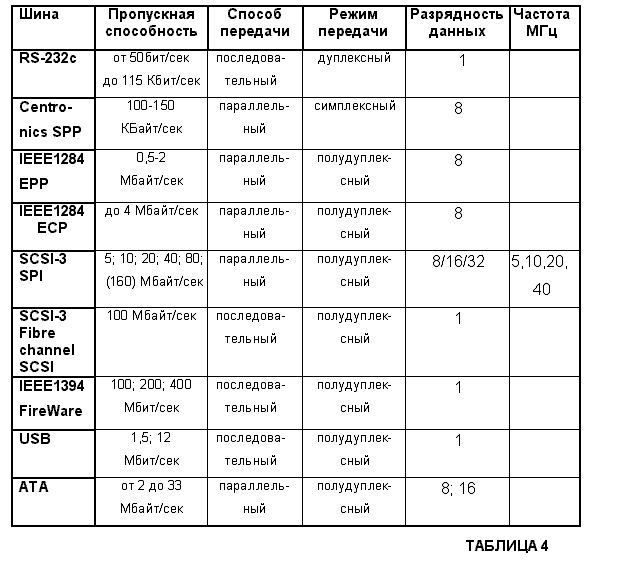
.

**1.5. ИНТЕРФЕЙСЫ ПЕРИФЕРИЙНЫХ УСТРОЙСТВ**

Шины ввода-вывода взаимодействуют с ПУ не непосредственно, а с помощью внешних интерфейсов (интерфейсов периферийных устройств). В отличие от шин расширения, шины внешних интерфейсов имеют большую длину и позволяют подключать к системному блоку компьютера ПУ, находящиеся на расстоянии до нескольких метров. Внешние интерфейсы ПУ можно разделить на две группы: специализированные интерфейсы и универсальные интерфейсы.

Специализированные интерфейсы обслуживают только один тип ПУ. К ним можно отнести: интерфейсы клавиатуры, интерфейс манипуляторов, аудиоинтерфейсы, интерфейсы мониторов, интерфейс игрового адаптера, интерфейс флоппи-дисков. Универсальные интерфейсы позволяют подключать различные типы ПУ (печать, сканер, графопостроитель, видеокамера и т.д.). К этим интерфейсам относятся: последовательные интерфейсы RS-232c, USB, Fire Wire и др., параллельные интерфейсы Centronics (стандарт IEEE1284), SCSI, ATA и др.

С развитием шин ввода-вывода и совершенствования самих ПУ изменяются и внешние интерфейсы. Модифицируются старые, появляются новые. Но эволюция интерфейсов ПУ идет меньшими темпами, чем системных интерфейсов и шин расширения. В таблице 4 приведены характеристики наиболее распространенных универсальных интерфейсов ПУ в их ретроспективе, начиная с интерфейсов, использовавшимися в первых ПК.



Практически не изменился за весь период развития ПК интерфейс RS-232, который появился в 1969г. Это наиболее простой и дешевый интерфейс. Он является дуплексным интерфейсом с последовательной передачей данных в асинхронном и синхронном режимах со скоростью до 115 Кбит/сек и топологией "точка-точка".

Интерфейс RS-232 чаще всего используется для подключения различного типа манипуляторов, для связи двух компьютеров, подключения принтеров и плоттеров, а также электронных ключей (Security Devices), предназначенных для защиты от нелицензированного использования программного обеспечения. Этот интерфейс позволяет эмулировать специальные терминалы (UT-52, UT-100 и т.д.). Он используется для беспроводных коммуникаций с применением излучателей и приемников инфракрасного диапазона - IR Connection.

Centronics – параллельный, симплексный интерфейс с побайтной передачей данных со скоростью до 150 КБайт/сек, предназначенный для подсоединения механических печатающих устройств, имеющий топологию "точка-точка". Он был разработан фирмой Epson в начале 80-х годов. С появлением новых видов ПУ и повышения их быстродействия произошло развитие Centronics. В 1994 году появился стандарт IEEE1284, где определялось три вида портов. SPP – начальный вариант Centronics.

EPP (Enchanted Parallel Port) – улучшенный параллельный порт. ECP (Extended Capability Port) – порт с расширенными возможностями. Последние варианты портов повысили быстродействие до 4-х Мбайт/сек, сделали интерфейс полудуплексным, что дало возможность использовать его для других видов ПУ, например, плоттеров, сканеров, видеокамер и т.п., а также использовать режим прямого доступа к памяти.

Наиболее быстродействующим, но и дорогим, является интерфейс SCSI (Small Computer System Interface), который был стандартизирован в 1986г. Этот интерфейс предназначен для подключений ПУ различных классов: жестких дисков, стримеров, CD-ROM, принтеров, сканеров и т.п. Это параллельный полудуплексный интерфейс со шлейфовой топологией соединения ПУ. С момента его появления прошло три его модификации. В 1994 году появились SCSI-2, а 1997 – SCSI-3. Скорость его возросла с 5 до 80 (160) Мбайт/сек, разрядность стала 8,16, (32) бита. 32-х разрядная шина практически не применяется. SCSI сейчас самый быстрый внешний интерфейс.

В последние годы появилась тенденция по созданию последовательных интерфейсов, не уступающих по скорости и количеству подключаемых ПУ SCSI, но имеющий более низкую стоимость.

В 1996г. появился последовательный интерфейс USB (Universal Serial Bus), работающий на скорости до 12 Мбит/сек и позволяющий подключать до 128 ПУ. Шина USB имеет древовидную структуру и требует специальные разветвители – хабы.

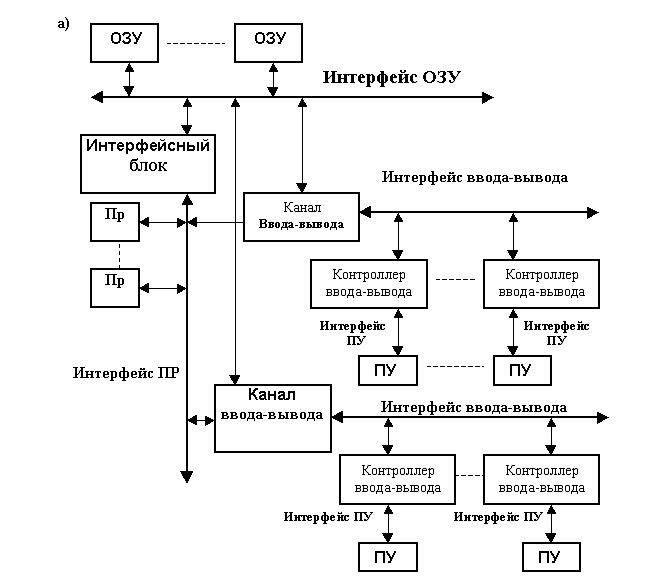
В 1995 был принят стандарт IEEE 1394, основанный на шине Fire Wire. В этом интерфейсе нет хабов и возможно подключение до 63 ПУ, скорость обмена 100 - 400 Мбит/сек.

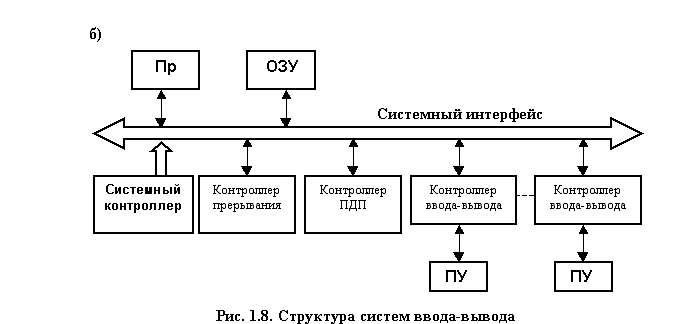
**1.6. СТРУКТУРА СИСТЕМ ВВОДА-ВЫВОДА**

Структура систем ввода-вывода представляет из себя совокупность взаимосвязанных внутренних и внешних интерфейсов (шин), посредством которых все устройства (модули) объединены в единую систему, называемую компьютером. Причем каждая шина имеет определенную скорость передачи информации, и к ней подсоединяются устройства с соответствующим быстродействием. Все шины, как правило, могут работать параллельно, обеспечивая высокую производительность вычислительной системы. Шины соединяются между собой с помощью специальных устройств – мостов. Кроме того, в структуру систем ввода-вывода входят устройства управления шинами и схемы организации процессов передачи информации при различных режимах ввода-вывода.

В процессе развития вычислительной техники формировалась структура самого компьютера и его систем ввода-вывода, разрабатывались и внедрялись различные типы интерфейсов.

Из поколения в поколение менялась элементная база и архитектура компьютеров. К моменту появления микропроцессоров (МП), больших интегральных схем (БИС) и персональных компьютеров уже сформировались определенные принципы построения и структуры систем ввода-вывода (см. рисунок 1.8).





В больших компьютерах (Мейнфрейм) типа IBM-360(370), ЕС ЭВМ-1033 (1040,1060), работающих в мультипрограммном режиме и имеющих мощный процессор с сопроцессором, большой емкости ОЗУ и много разнообразных ПУ, использовалась многомагистральная структура с выделенными каналами ввода-вывода и каскадно-магистральным подключением ПУ (см. рисунок 1.8а). В таких машинах, как правило, использовались два типа аппаратно реализованных каналовввода-вывода: мультиплексный, работающий с медленными ПУ и селекторный, обслуживающий быстродействующие ПУ. Такая структура позволила максимально использовать вычислительную мощность компьютера за счет одновременного решения нескольких задач и параллельной работы процессора и каналов ввода-вывода.

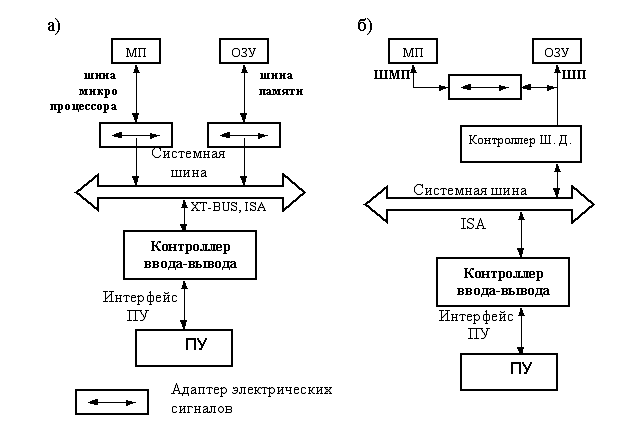
В таких компьютерах аппаратно реализовывались все функции, показанные на рисунке 1.4. В них система ввода-вывода содержала оптимальный набор из нескольких типов интерфейсов. Высокоскоростные интерфейсы процессора и ОЗУ, через которые взаимодействовали основной процессор, специализированные процессоры, блоки оперативной памяти обеспечивали максимальное использование процессорного времени. Интерфейсы ввода-вывода, аппаратно реализованные каналы ввода-вывода и контроллеры ПУ, освобождали центральный процессор от процедур управления вводом-выводом. Интерфейсы ПУ предназначались для подключения ПУ к компьютеру.

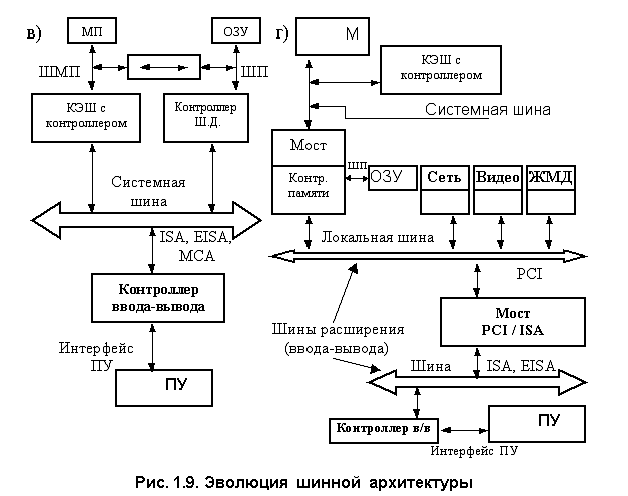
В малых вычислительных машинах типа DEC PDP-11, СM-1(2,3,4), которые были намного дешевле больших, система ввода-вывода строилась по одномагистральной структуре с распределенным каналом ввода-вывода и радиально-магистральным подключением ПУ (см. рисунок 1.8б). Мультиплексный канал, как отдельное устройство, отсутствовал, его функции выполнял процессор. В качестве селекторного канала применялся контроллер прямого доступа к памяти с ограниченными функциями канала. Существовали стандарты на используемые внутримашинные системные шины, например, Unibus фирмы DEC (общая шина).

С появлением МП и БИС наступил новый этап развития структур систем ввода-вывода, обусловленный новым принципом построения вычислительных машин на основе модульности, микропрограммируемости и магистральности, о чем говорилось во введении. Новый этап повторял стадии развития предыдущего, но на качественно новой элементной базе и других подходах к компоновке компьютеров.

С развитием элементной базы компьютеров, повышением скорости работы микропроцессоров и микросхем памяти, увеличением емкости ОЗУ совершенствовалась и изменилась структура системы ввода-вывода информации, повышалась скорость работы интерфейсов. Развитие интерфейсов и систем ввода-вывода было направлено на минимизацию потерь в производительности компьютера, вызванных задержками в передаче информации между его модулями (устройствами), т.е. передач процессор-ОЗУ, процессор-ПУ, ПУ-ОЗУ. Особенно остро проблемы по увеличению скорости работы интерфейсов и систем ввода-вывода в целом встала в связи с появлением нового поколения МП типа Pentium и с использованием в компьютерах средств мультимедиа. В частности узким местом стало обеспечение качественной 3-х мерной графики и «живого» видео, требующих скоростей передачи более 500 Мбайт/сек.

Наиболее наглядно и полно можно проследить и прочувствовать проблемы и тенденции развития систем ввода-вывода при рассмотрении ретроспективы эволюции интерфейсов и структур систем ввода-вывода на примере персональных компьютеров типа IBM PC (см. рисунок 1.9).





В начале эры персональных компьютеров частота работы процессора составляла 10 МГц, при этом на выполнение даже самых простейших операций процессор затрачивал несколько тактов. В таких условиях для обеспечения бесперебойной работы процессора было достаточно всего 4 миллионов обращений к памяти в секунду, что соответствовало циклу работы в 250 нсек. Этимусловиям удовлетворяла одношинная структура систем ввода-вывода, когда все устройства компьютера, включая ОЗУ, общались с процессором через общую шину (см. рисунок 1.9а), которую называли системной. Все интерфейсы ПУ подключались к этой шине. Наиболее распространенной системной шиной в этот период стала сначала 8 разрядная, затем 16 разрядная шина ISA, работающая на частоте 8 МГц.

С ростом частоты работы ПК (см. таблицу 3) и изменения времени доступа к ОЗУ пропускная способность шины ISA в 8 Мбайт/сек стала тормозить работу процессора. Решение проблемы нашли в выделении канала передачи данных МП-ОЗУ в отдельную шину, построенную на базе внешнего интерфейса МП, и изолированную от медленной шины ISA посредством контроллера шины данных. Это повысило производительность работы центрального процессора. Все ПУ продолжали взаимодействовать с центральным процессором через системную шину (см. рисунок 1.9б).

С дальнейшим ростом частоты работы МП тормозом в работе стало ОЗУ. Тогда ввели дополнительную высокоскоростную КЭШ-память (см. рисунок 1.9в), что уменьшило простои МП. Все ПУ продолжали работать через системную шину, но кроме ISA появились более скоростные шины EISA и MCA (см. таблицу 3).

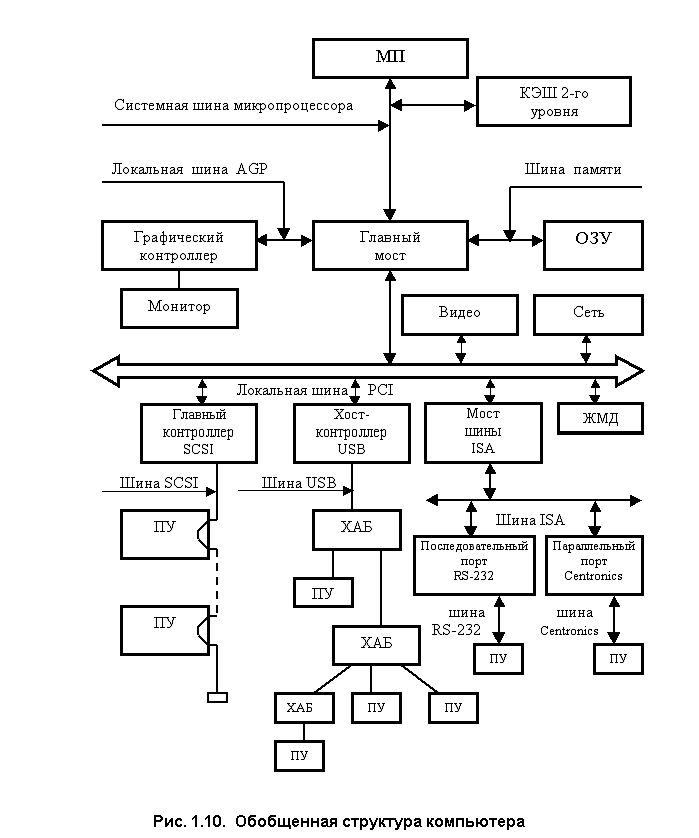
На определенном этапе развития компьютеров стали широко использовать мультимедиа. Сразу выявилось узкое место во взаимодействии центрального процессора и видеокарты. Потребовалась пропускная способность более 100 Мбайт/сек. Имеющиеся системные шины ISA, ЕISA, МСА не удовлетворяли этим условиям. Их пропускная способность составляла от 16 до 30 Мбайт/сек.

Выход был найден с разработкой и внедрением высокоскоростных локальных шин, посредством которых можно было связаться с памятью, на этой же шине работали жесткие диски, что также повышало качество вывода графической информации. Первой такой шиной была шина VL-bus, практически повторявшая интерфейс МП i486. Затем появилась локальная шина РСI. Она была процессорно-независимой и поэтому получила наибольшее распространение для последующих типов МП. Эта шина имела частоту работы 33 МГц и при 32-х разрядных данных обеспечивала пропускную способность в 132 Мбайт/сек (см. рисунок 1.9г). Системная шина ISA по-прежнему использовалась в компьютерах, что позволяло применять в новых компьютерах огромное количество ранее разработанных аппаратных и программных средств.

В такой системе ввода-вывода различные ПУ подключались к разным шинам. Медленные - к ISA, а высокоскоростные - к PCI. С появление шины PCI стало целесообразным использовать высокоскоростные параллельные и последовательные интерфейсы ПУ (SCSI, ATA, USB). На этом этапе системной стали называть шину МП, через которую он взаимодействовал с ОЗУ. Шина РСI и ISA и подобные другие назвали шинами ввода-вывода или шинами расширения. Действительно, эти шины как бы расширяли число устройств (см. рисунок 1.9г), работающих с ЦПр, и их основной функцией стало обеспечение процессов ввода и вывода информации.

Появление шины PCI не сняло всех проблем по качественному выводу визуальной информации для 3-х мерных изображений, "живого" видео. Здесь уже требовались скорости в сотни Мбайт/сек. В 1996г. фирма Intel разработала новую шину AGP, предназначенную только для связи ОЗУ и процессора с видеокартой монитора. Эта шина обеспечивала пропускную способность в сотни Мбайт/сек. Она непосредственно связывала видеокарту с ОЗУ минуя шину РСI.

На рисунке 1.10 изображена обобщенная структура системы ввода-вывода современного компьютера. Как видно из рисунков 1.9г и 1.10, через 20 лет снова пришли к многомагистральной структуре ввода-вывода с радиально-магистральными интерфейсами ПУ (см. рисунок 1.8).



Следует также отметить то, что шина ISA присутствует во всех структурах систем ввода-вывода. Это обусловлено стремлением пользователей сохранить преемственность новых компьютеров с ранее созданными и широко распространенными аппаратными и программными средствами. Хотя фирмам Intel и Microsoft предрекали, что с 1998г. шина ISA исчезнет, и все перейдут на соответствующее устройства для шины РСI, но шина ISA к 2000 году сохранила свои позиции.

Все шины систем ввода-вывода объединяются в единую транспортную среду передачи информации с помощью специальных устройств: мостов и контроллеров ввода-вывода.

Мост – устройство, применяемое для объединения шин, использующих разные или одинаковые протоколы обмена. Мост – это сложное устройство, которое осуществляет не толькокоммутацию каналов передачи данных, но и производит управление соответствующими шинами. В структуре компьютера, использующего шину РСI, применяются три типа мостов (см. рисунок 1.10).

Мост шины (РСI Bridge), производящий подключение шины РСI к другим шинам, например, ISA или ЕISA. Главный мост (Host Bridge), соединяющий шину РСI с системной шиной, кроме того, этот мост содержит контроллер ОЗУ, арбитр и схему автоконфигурации. Одноранговый мост (Peer-to-Peer) для соединения двух шин РСI между собой. Это делается для увеличения числа устройств, подключаемых к шине.

Контроллер ввода-вывода управляет обменом информацией с ПУ. Он практически представляет из себя мост между шиной ввода-вывода и интерфейсом ПУ. В контроллер ввода-вывода могут входить специальные схемы-адаптеры, предназначенные для преобразования представления и скорости передачи информации. Примерами таких адаптеров могут служить асинхронные приемопередатчики типа i8250/16450/1656A, используемые в интерфейсах типа RS-232C, адаптер параллельного интерфейса 8255 и др.

Для управления шинами и обеспечения выполнения функций интерфейсов, входящих в систему ввода-вывода, применяются специальные контроллеры и схемы. К ним можно отнести контроллеры прерываний 8259А и прямого доступа к памяти 8237А, таймер 8254А, часы реального времени, буферы шин данных, дешифраторы, мультиплексоры, регистры и другие логические устройства.

В первых компьютерах, построенных с использованием микропроцессоров, контроллер и другие устройства строились на базе набора интегральных схем малой, средней и большой степени интеграции. Адаптеры, таймер и др. выпускались в виде отдельных микросхем (8250, 8255, 8259, 8237 и т.д.)

С повышением производительности компьютеров и увеличением степени интеграции все вышеперечисленные устройства и схемы стали объединяться в микросхемы со сверхбольшой степенью интеграции, образуя специальные наборы интегральных схем, называемых «чипсет»

(ChipSet).

В настоящее время управление потоками передаваемых данных производится с помощью мостов и контроллеров, входящих в ChipSet. Именно ChipSet определяет основные особенности архитектуры компьютера и, соответственно, достигаемый уровень производительности в условиях, когда лимитирующим фактором становится не процессор, а его окружение – память и система ввода-вывода.

Первым серьезным шагом на пути повышения эффективности передачи информации в компьютерах на основе Pentium стал ChipSet третьего поколения 430FX (Triton) фирмы Intel, состоящий из 3-х микросхем. Чипсет первого поколения – 450GX/GX (Orion) был выполнен на 7 микросхемах. В 1999 году фирма Intel приступила к выпуску нового чипсета 440LXAGPset (см. рисунок 1.11) с технологией ускоренного графического порта (AGP), двойной независимой шиной (DIB) процессора Pentium II и поддержкой памяти типа SDRAM. Этот чипсет состоял из 2-х микросхем.

# 2. Програмований інтервальний таймер КР580ВІ53

# 2.1 Принцип роботи таймера

ВІС програмованого таймера КР580ВІ53 призначена для організації роботи мікропроцесорних систем у режимі реального часу і дозволяє формувати сигнали з різними тимчасовими і частотними характеристиками.

Мiкросхема являэ собою однокристальний трьохканальний програмуэмий пристрiй,призначений для отримання програмно-керованих часових затримок i виконання заданих часових функцiй в мiкропроцесорних системах. ВIС мiстить три канала. В кожному каналi э регiстр керуючого слова,16-розрядний програ-муэмий лiчильник працюючий у двійовому або двійково-десятковому кодi (реверсивний), а також двобайтний буферний регiстр, в який по спеціальній командi переписуються константи лiчильника.

Програмування кожного каналу таймера зводиться до слідуючих операцій:

-запис у регістр індивідуального керуючого слова керування каналом;

-запис у 16-розрядний програмуємий лічильник необхідної константи перерахунку;

Запис індивідуального слова керування каналу здійснюється по єдиній для всіх каналів адресі (А0=1, А1=1) регістру керуючого слова.

Вказівка конкретного каналу, до якого відноситься керуюче слово, містится у самому керуючому слові.

Адресація регістрів таймера

|  |  |  |
| --- | --- | --- |
| А0 | А1 | Найменування |
| 0  0  1  1 | 0  1  0  1 | Лічильник каналу 0  Лічильник каналу 1  Лічильник каналу 2  Регістр керуючого слова |

# 

# 2.2 Блок схема таймера

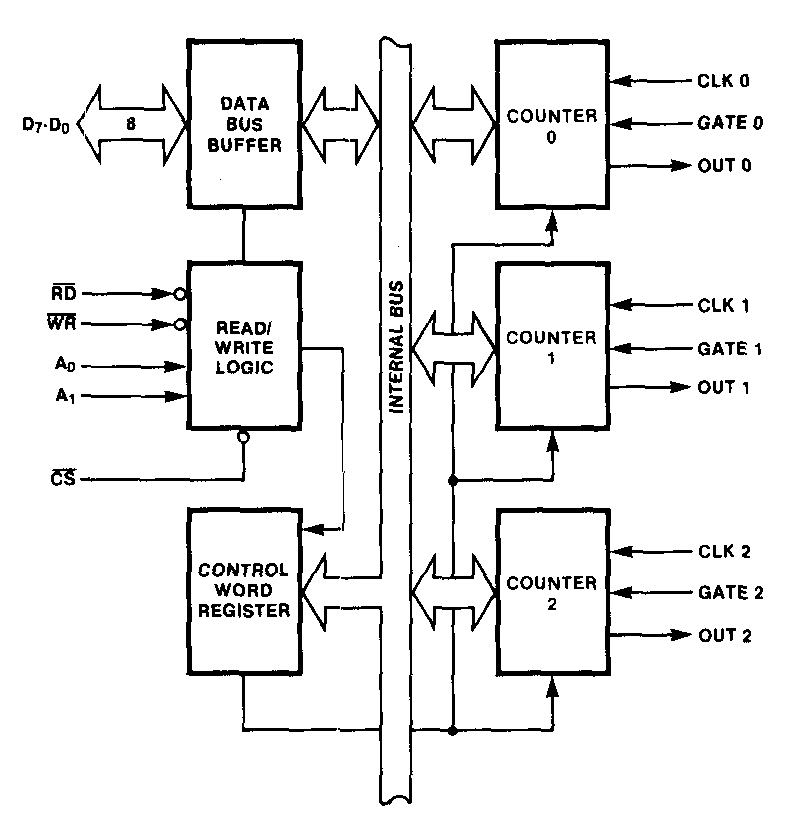
Спрощена структурна схема ПТ приведена на рис.3.1. До складу ВІС входять:

буфер даних *(BD),* призначений для обміну даними і керуючими словами між МП і ПТ;

схема керування читанням-записом *(RWCU),*щозабезпечує виконання операцій введення-виводу інформації в ПТ;

регістр керуючого слова *(RGR),* призначений для запису керуючих слів, що задають режими роботи лічильників;

лічильник каналів(СТ0-СТ2).



*DBB -* буфер даних

*RWL -* схема керування читанням-записом

*CWR -* регістр керуючого слова

*СТ0-СТ2 –* лічильник каналів

D7-D0- лінії шини даних

А0-А1- лінії адреси

/RD, /WR-сигнали читання-запису

/CS-сигнал вибору мікросхеми

CLK0, CLK1, CLK2-лічильні входи

OUT0, OUT1, OUT2-виходи лічильників

GATE0, GATE1, GATE2-входи дозволу-заборони ліку та апаратного запуску рахування

CS/ - вибiр корпусу;

Рисунок 2.1-Спрощена структурна схема таймеру

Підключення ПТ до шин мікропроцесора показане на рисунку 3.2.



ША

**ШД**

**ШК**

Рисунок 2.2-Підключення до шин системи

Умовне графічне зображення І8253 представлене на рис. 2.3.



Рисунок 2.3-Графічне зображення таймера

**2.3 Принцип роботи**

ВIС І8253/54 мiстить 3 16-розрядних двійково-десяткових регістрів з попереднiм заповненням .Функціонпльна конфiгурація його входiв i виходiв встановлюється за допомогою засобiв програмного забеспечення.

Лiчильники повнiстю незалежнi. Кожен може мати свiй режим роботи i тип лiчби (двійковий або двійково-десятковий). Завантаження лiчильникiв початковим значенням лiчби визначається програмно.

Зчитування змiсту кожного лiчильника здiйснюється за допомогою команди вводу.Таймер також мiстить додаткове обладнання i команду, якi дозволяють читати змiст лiчильника "на льоту", не заважаючи його роботi.

Програмування таймера вiдбувається за допомогою команди виводу. Кожен лiчильник iндивiдуально налагоджується вiдповiдно змiсту регiстра керуючого слова.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SC1**  00-канал 0  01-канал 1  10-канал 2  11-код команди RBC  00-код команди CLC  01-читання\запис мл.байту к.  10-читання\запис ст.байту к.  11-читання\запис мл. а потім ст. байту к.  000-режим 0  001-режим 1  010-режим 2  011-режим 3  100-режим 4  101-режим 5  0-двійковий  1-дв.\десятко-вий рахунок | **SC0** | **RW1** | **RW0** | **M2** | **M1** | **M0** | **BCD** |

CLK - тактовi входи лiчильника;

GATE - входи блокування роботи;

OUT - виходи лiчильника;

RD/ - читання лiчильника;

WR/ - запис команди або даних;

Рисунок 2.4 –Формат команди керування

**2.4 Режими роботи таймера**

Таймер має шість режимів роботи:

* + режим «нуль» - формування сигналу по закінченню інтервалу часу;
  + режим «1» формування імпульсу заданої тривалості;
  + режим «2» генератор імпульсі потрібної частоти;
  + режим «3» програмований генератор прямокутних імпульсів;
  + режим «4» формування імпульсу по закінченню заданого часу;
  + режим «5» лічильник подій.

Режими передбачають зміну виходу таймера OUT з низького рівня OUT=0 до високого OUT=1 після закінчення лічби ( досягнення значення 0 константи перерахунку).

**Режим 0. Формування сигналу по закінченню заданого інтервалу часу**

Після запису за допомогою сигналу /WR керуючого байта СWR, який переводить відповідний канал таймеру у режим 0 вихід таймера OUT встановлюється значення логічної «1»(рис .1.6 а).

Після запису константи перерахунку ( наприклад ,N=4), канал N починає роботу (рис 1.6,б) під час якої константа перерахунку зменшується по тактам вхідних імпульсів CLK, вихід OUT встановлюється в 0.

Коли константа перерахунку досягне нульового значення , вихід таймера встановлюється у високий рівень OUT=1 (рис.1.6,б).

Високий рівень входу GATE=1 дозволяє роботу лічильника, а GATE=0 – призупиняє його роботу. Таким чином, час видачі сигналу по закінченню лічби залежить від введеної константи перерахунку і може збільшуватись на тривалість ∆t за допомогою сигнала керування GATE (рис.1.6,в).

З рисунку видно, що в режимі 0 сигнал

GATE 1 –дозволяє лічбу.

Перехід GATE з 1 в 0 -призупиняє лічбу

Перехід GATE 0 в 1 -дозволяє продовження рахунку.

**Режим 1. Формування iмпульсу заданої тривалості (строб)**

Після запису за допомогою сигналу /WR керуючого байта СWR , який переводить відповідний канал таймера у режим 1 . Вихід таймера OUT встановлюється в 1 (рис.1.7а).

Після завантаження константи перерахунку (N=4) таймер починає роботу і формується інтервал часу ∆t строб в залежності від константи перерахунку, а вихід таймера встановлюється у низький рівень OUT=0 (рис.1.7,б).

Високий рівень входу GATE=1, а також перехід його в низький рівень і його утримання, дозволяє роботу лічильника, а перехід сигналу GATE з низького рівня у високий, призводить до перезавантаження константи перерахунку лічильника на початкове значення , що дозволяє змінювати інтервал стробу (рис.1.7,в).

З рисунку видно, що в режимі 1 сигнал

GATE 1 –дозволяє лічбу

Перехід GATE з 1 в 0 - дозволяє лічбу

Перехід GATE з 0 в 1 -перезавантаження константи перерахунку.

**Режим 2. Програмований генератор імпульсів потрібної частоти**

Після запису за допомогою сигналу /WR керуючого байта СWR , який переводить відповідний канал таймера у режим 2 . Вихід таймера OUT встановлюється в 1 (рис.1.8а).

Після завантаження константи перерахунку (N=4) таймер починає роботу.

Коли константа перерахунку досягне значення 1, вихід таймера встановлюється у низький рівень OUT=0 і тримається у цьому стані доки константа перерахунку досягне нульового значення, після чого вихід таймера приймає значення 1 (OUT=1) і відбувається апаратне перезавантаження константи перерахунку на початкове значення N=4 (рис.1.8,б).

Високий рівень GATE =1 , а також перехід сигналу GATE з високого рівня в низький і його утримання , дозволяє роботу таймера, а перехід сигналу GATE з низького рівня у високий, призводить до перезавантаження константи перерахунку лічильника на початкове значення, що дозволяє змінювати частоту імпульсів(рис.1.8,в).

З рисунку видно, що в режимі 2 сигнал

GATE 1 –дозволяє лічбу

GATE 0 –зупиняє лічбу

Перехід GATE з 0 в 1 -перезавантаження константи перерахунку.

**Режим 3. Програмований генератор прямокутних імпульсів**

Після запису керуючого байта СWR , відповідний канал таймера налаштовується на режим 3 . Вихід таймера OUT встановлюється в 1 (рис.1.9а).

Після завантаження константи перерахунку таймер починає роботу і рівень логічної одиниці на виході таймера OUT=1 буде триматися до половини значення константи перерахунку N/2 (якщо N парне), після цього встановлюється логічний 0 , який утримується до досягнення нульового значення константи перерахунку N(рис.1.9,б), після чого відбувається автоматичне перезавантаження константи перерахунку на початкове значення і таймер починає свою роботу спочатку. Таким чином на виході таймера формується сигнал прямокутної форми, в якому тривалість логічного нуля дорівнює тривалості логічної одиниці (меандр,якщо N парне). Якщо число N непарне то на виході таймера логічна одиниця буде протягом (n+1)/2 і логічний нуль протягом (n-1)/2.

Високий рівень GATE=1 дозволяє роботу таймера ,а низький рівень GATE=0 призупиняє, а перехід сигналу GATE з 0 в 1 перезавантажує константу перерахунку на початкове значення і дозволяє роботу таймера (рис.1.9,в).

З рисунку видно, що в режимі 3 сигнал

GATE 1 –дозволяє лічбу

GATE 0 –призупиняє лічбу

Перехід GATE з 0 в 1 -перезавантаження константи перерахунку.

**Режим 4. Формування iмпульсу по закiнченню заданого часу**

Після запису за допомогою сигналу /WR керуючого байта СWR , який переводить відповідний канал таймера у режим 4. Вихід таймера OUT встановлюється в 1 (рис.1.10а).

Після запису константи перерахунку (N=4) таймер починає роботу.

При досягненні константи N=0 стан виходу таймера OUT встановлюється в 0 на один період CLK , після чого OUT знову встановлюється в 1 ( тобто через період часу (n+1) Tсlk) OUT=0 на період одного такту (рис.1.10,б).

Високий рівень входу GATE=1 дозволяє роботу лічильника, GATE=0 – призупиняє його роботу, а перехід сигналу GATE з низького рівня у високий, призводить до продовження рахунку(рис.1.10,в).

З рисунку видно, що в режимі 4 сигнал

GATE 1 –дозволяє лічбу

Перехід GATE з 1 в 0 - призупиняє лічбу

Перехід GATE 0 в 1 -дозволяє продовження лічби.

**Режим 5. Лiчильник подiй**

Після запису за допомогою сигналу /WR керуючого байта СWR , який переводить відповідний канал таймера у режим 5. Вихід таймера OUT встановлюється в 1 (рис.1.11а).

Після запису константи перерахунку (N=4) таймер починає роботу.

При досягненні константи N=0 стан виходу таймера OUT встановлюється в 0 на один період CLK, після чого OUT знову встановлюється в 1 ( тобто через період часу (n+1) Tсlk) OUT=0 на період одного такту (рис.1.11,б).

Високий рівень входу GATE=1, а також перехід в 0 і його утримання, дозволяє роботу лічильника, а перехід сигналу GATE з низького рівня у високий, призводить до перезавантаження константи перерахунку лічильника на початкове значення , що дозволяє змінювати час видачі імпульсу (рис.1.11,в).

З рисунку видно, що в режимі 5 сигнал

GATE 1 –дозволяє лічбу

Перехід GATE з 1 в 0 - дозволяє лічбу

Перехід GATE з 0 в 1 -перезавантаження константи перерахунку.

Керуючі слова можуть бути записані в ПТ у довільному порядку. У будь-який наступний момент часу записуються початкові коди лічильників у відповідності зі значеннями розрядів *D5, D4* керуючих слів.

Виходячи з особливості впливу сигналу GATE на роботу лічильника каналу інтервального таймера, режими його роботи можна поділити на дві групи:

* + з призупинкою (паузою) по сигналу GATE (режим 0,3,4);
  + з перезавантаженням по сигналу GATE (режим 1,2,5).







# 2.6 Програмування таймера

Програмування таймера досить гнучке, не має значення послiдовнiсть завантаження керуючих слов режиму окремих лiчильникiв, тобто не обов'яз-ково повинно бути першим керуюче слово лiчильника 0, останнiм-2. Регiстр керуючого слова режиму кожного лiчильника має свою адресу i може завантажуватись незалежно . Однак завантаження змiсту лiчильника повинно вiдповiдати послiдовностi, запрограмованiй в керуючому словi режиму (-старший байт, молодший байт ).

При програмуванні таймера програміст вирішує наступні задачі:

1. Ініціалізація каналів інтервального таймера.

Здійснюється за допомогою

* + Запису керуючого слова CWR,
  + Запису константи перерахунку.

2.-Читання регістру стану інтервального таймера, або константи перерахунку RBC ( Read Back Command) здійснюється за допомогою

- запису керуючого байту RBC,

- читання регістру стану ініціалізації таймеру RST,

- читання константи перерахунку

а) з зупинкою таймера

б) без зупинки таймера.

Керуюче слово задає один з шести режимів роботи, тип ліку (двійковий чи двійково-десятковий), порядок завантаження і розмірність (один чи два байти) константи. Воно завантажується в регістри RSW каналів таймера, зберігається до слідуючого перепрограмування. Після цього завантажуються константи, які приймають значення:

* + двійковий – 0-65536
  + двійково-десятковий – 0-9999

Константи перерахунку завантажуються у таймер при А0, А1 ( 00, 01, 10 ).

Існує 2 варіанти завантаження: спочатку всі слова керування, а потім константи, чи слово управління і константу для кожного каналу послідовно.

**2.7 Читання змісту лічильника**

Якщо потрiбно знати значення лiчби у процесi роботи (особливо це використовується при визначеннi кiлькостi подiй), то можливо використати два засоб читання, якi не пошкоджують iнформацiю:

1. Використовуючи команду вводу iз обраного лiчильника (READ), керуючи входами А0-А1 п рограмiст може вибирати лiчильник значення якого потрiбно знати.

При читаннi значення лiчильника треба заборонити роботу лiчильника за допомогою команди GATE або шляхом заборони тактових iмпульсiв. При читаннi записiв лiчильника першим читається молодший байт LSB, другим- старший байт MSB. Слiд пам'ятати, що процедура читання повинна бути обов'язково доведена до кiнця. Якщо запрограмовано читати два байти, то вони повиннi бути прочитанi до слiдуючоi команди завантаження.

2)Другий засiб - читання пiд час лiчби по комані CLC (на льоту), без зупинки . Для реалiзацii цього засобу таймер має додатковий регiстр, звернення до якого здiйснюється за допомогою команди вивiд. При цьому для читання змiсту лiчильника "на льоту" в регiстр керуючого слова режиму завантажується спецiальний код, пiд дiєю якого змiст лiчильника без впливу на його роботу фiксується у додатковому регiстрi. Пiсля цього використовується звичайна операцiя ВВОД для читання змiсту регiстра.



Риунок 2.5 –Формат команди СLC

Таймеру відповідають чотири порти введення/виводу з наступними адресами наприклад:

40h - канал 0;

41h - канал 1;

42h - канал 2;

43h - керуючий регістр.

Наприклад:

MVI A, 00110000B ;формування керуючого слова лічильника 0

OUT PORT\_RUS ;запис 2-х байт конст., режим 0, рахунок двійковий-десятковий

MVI A, DATA\_LOW ; завантаження молодшого байта кон. У лічильник 0

OUT PORT\_0 ; при А0,А1=00

MVI A, DATA\_HIGH ; завантаження старшого байта кон. У лічильник 0

OUT PORT\_0 ; при А0, А1=00

MVI A, 00000000B ; формування керуючого слова для читання

OUT PORT\_RUS ; на льоту слова стану і завантаження в таймер

IN PORT\_0 ; читання молодшого байта лічильника 0

IN PORT\_0 ; читання старшого байта лічильника 0

Контрольні питання та завдання

1. Архітектура Програмованого Таймера (ПТ) І8253. Призначення, структурна схема, призначення основних блоків, вхідних/вихідних сигналів.

2. Система мікрокоманд, формати команд, формат слова-стану. Режими роботи ВІС І8253 (програмований таймер). Підключення до шин МП.

3.Поясніть адресацію регістрів таймера.

4.Намалюйте схему таймера та поясніть його роботу.

5. На лабораторній розробіть фрагменти програм ініціалізації таймера , в різних режимах.

**3 Програмований адаптер послідовного інтерфейсу**

**3.1 Послідовні інтерфейси**

Послідовний інтерфейс для передачі даних використовує одну сигнальну лінію, по якій інформаційні біти передаються один за одним послідовно. Звідси – назва інтерфейсу і порту. Англійські терміни - *Serial Interface* і *Serial Port* (іноді їх неправильно перекладають як "серійні"). Послідовна передача дозволяє скоротити кількість сигнальних ліній і збільшити дальність зв’язку. В ряді послідовних інтерфейсів використовується гальванічна розв’язка зовнішніх (звичайно вхідних) сигналів від схемної землі пристрою, що дозволяє з’єднання пристроїв, що знаходяться під різними потенціалами.

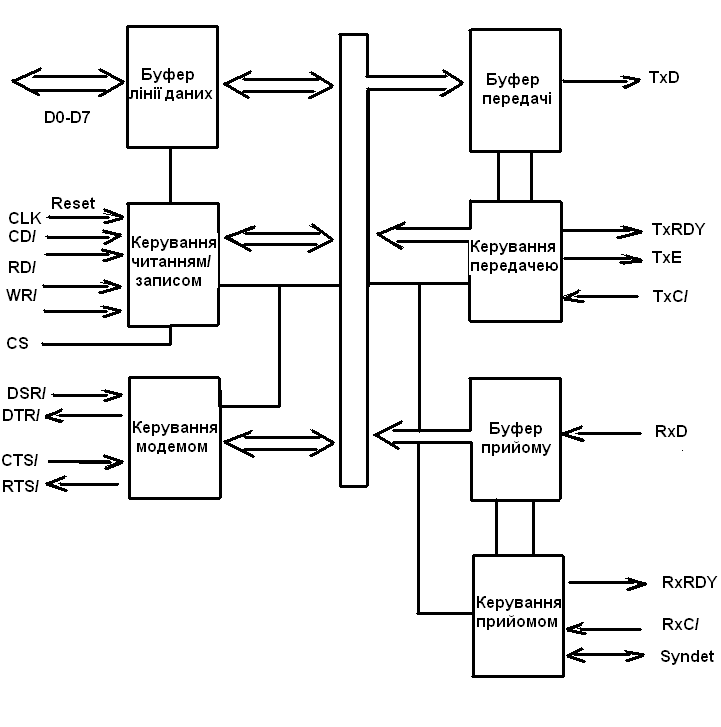
ВІС послідовного інтерфейсу І8251 являє собою універсальний синхронно-асинхронний прийомо - передавач (УСАПП) і призначена для організації обміну між MП і ЗП в послідовному форматі . УСАПП може приймати дані з 8-розрядної шини даних МП і передавати їх у послідовному форматі периферійним пристроям, а також одержувати послідовні дані від периферії й перетворювати їх у паралельну форму для передачі в МП.

Обмін даними виробляється в асинхронному режимі зі швидкістю передачі до 9,6 біт/с або в синхронному зі швидкістю до 56 біт/с. Довжина переданих символів становить від 5 до 8 біт. При передачі в МП символів, довжина яких менше 8 біт, ті біти, що не використовувалися заповнюються нулями. Формат символу включає також службові біти й необов'язковий біт контролю по парності (непарності) .

Спрощена структурна схема (блок-схема) УСАПП наведена на рисунку 4.1.

До складу ВІСа входять : буфер передавача зі схемою керування передавачем*,* призначені для прийому даних від МП і видачі їх у послідовному форматі на вихід ; буфер прuйомника зі схемою керування приймача*,* щовиконують прийом послідовних даних із входу Rx і передачу їх у МП у паралельному форматі; буфер лінії даних, що представляє собою паралельний 8-розрядний двунаправлений регістр із трьох стабільними кacкaдaми й службовецьдля обміну даними й керуючими словами між МП і УСАПП; блок керування записом/читанням*,* щоприймає керуючі сигнали від МП і генерує внутрішні сигнали керування; блок керування модемом (MCU), що обробляє керуючі сигнали, призначені для ЗП (зовнішнього пристрою). Призначення вхідних, вихідних і керуючих сигналів УСАПП розглянемо нижче.

**БЛОК-СХЕМА КР580ВВ51**



D0-D7 - лiнii даних.

RESET - гашення.

CLK - тактовий iмпульс.

СD/ - лог. 0 - на шинi даних данi; лог. 1 - команда.

RD/ - читання або вивiд.

WR/ - запис або введення.

CS/ - вибiр корпусу.

TXD - вихiд даних передавача.

RXD - вхiд даних приймача.

Рисунок 3.1

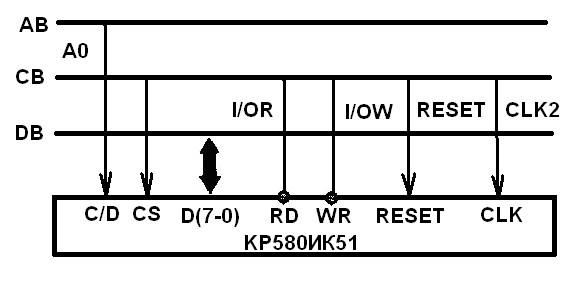


Рисунок 3.2- Підключення УСАПП до шин мікропроцесора

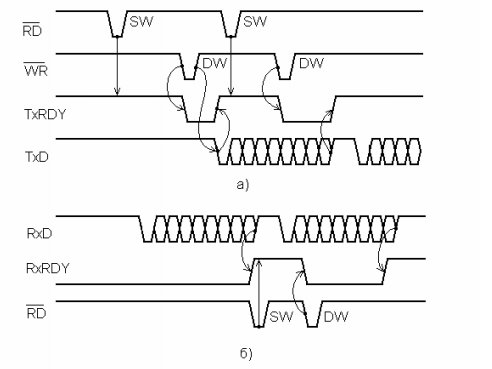


Рисунок 3.3-Тимчасові діаграми сигналів керування

а)передачі б)прийому а асинхроному режимі

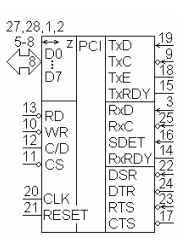


Рисунок 3.4- Мікросхема І8251

**3.2 Обмін інформацією між УСАПП і модемом**

Для управління модемом адаптер використовує наступні вхідні та вихідні сигнали:.

**DSR** – (DATA SET READY) – готовність модему прийняти дані від комп’ютера. Вхідний сигнал є сигналом загального призначення. Його стан можна перевірити за допомогою читання регістра стану.

**DTR** – (DATA ТERMINAL READY) – готовність терміналу до прийому/передачі даних від комп’ютера до модему. Основна функція – включити /виключити лінію. Програмно цей сигнал може бути встановлено за допомогою логічного нуля відповідного біта в команді керування. Вихідний сигнал DTR використовується для керування модемом, вибору швидкості і т.д.

**RTS** – (REQUEST TO SEND) – запит на посилку даних від комп’ютера до модему. Основна функція – переключення модему на прийом чи передачу. Програмно він може бути встановлений за допомогою відповідного біта розряду управління (D5).

**CTS**– (CLEAR TO SEND) – гашення в модемі при посилці даних шляхом заземлення. Стан логічного нуля дозволяє адаптеру проводити передачу даних, якщо розряд (біт) TxEN (D0=1) в команді управління знаходиться в стані логічної одиниці.

**TxRDY**  - готовнiсть передавача. Цей вихiд повiдомляе ЦП про готовнiсть передавача прийняти символ даних i може бути використаний для формування запиту в контролер переривань. ЦП може контролювати TxRDY за доп. читання регiстру стану. TxRDY автоматично встановлюється в лог. 0 при загрузцi символа iз ЦП. Сигнал TxRDY встановлюється в стан логiчноi 1 тiльки у тому випадку якщо CTS встановлено в 0. Цим сигнал TxRDY вiдрiзняетьcя вiд сигналу TxE.

**TxE** - вiдсутнiсть даних у передавача. Коли УСАПП не мае даних для передачi, на виходi TxE встановлюеться стан лог. 1. TxE автоматично встановлюється в стан лог. 0 при отриманнi символа iз ЦП. TxE може бути використаний для вказiвки закiнчення режиму передачi.

**TxC** - iмпульс тактування передавача. Керуе швидкiстю передачi cимвола. В режимi синхронноi передачi частота TxC дорiвнюе дiйснiй частотi передачi. У режимi асинхронноi передачi частота TxC кратна дiйснiй частотi.

**RxRDY** - готовнiсть приймача. Цей вихiд показуе що УСАПП має символ готовий до передачi в ЦП. RxRDY може бути використаний в структурi органiзацii перерви або при органiзацii опиту. RxRDY автоматично вствновлюеться в 0 пiсля передачi символа в ЦП.

**RXC** - iмпульс тактування приймача, керуе швидкiстю прийому символа.

**SYNC** - знаходження синхроiмпульса. Використовується тiльки в синхронному режимi.

**Блок управління передачею**

Управління передачею здійснюється за допомогою блока передавача, який має такі сигнали:

**TxRDY** - готовнiсть передавача. Цей вихiд повiдомляє ЦП про готовнiсть передавача прийняти символ даних i може бути використаний для формування запиту в контролер переривань. ЦП може контролювати TxRDY за доп. читання регiстру стану. TxRDY автоматично встановлюється в лог. 0 при завантаженні символа iз ЦП. Сигнал TxRDY встановлюється в стан логiчної 1 тiльки у тому випадку, якщо CTS встановлено в 0. Цим сигнал TxRDY вiдрiзняєтьcя вiд сигналу TxE.

**TxE** - вiдсутнiсть даних у передавача. Коли УСАПП не має даних для передачi, на виходi TxE встановлюється стан лог. 1. TxE автоматично встановлюється в стан лог. 0 при отриманнi символу iз ЦП. TxE може бути використаний для вказiвки закiнчення режиму передачi.

**TxC** - імпульс тактування частоти передавача. Керує швидкістю передачі символу. В режимі синхронної передачі частота TxC дорівнює дiйснiй частоті передачі. У режимі асинхронної передачі частота TxC кратна дiйснiй частоті. В синхронній передачі частота ТхС рівна передачі дійсній частоті. Частина команди режиму вказує на кратність частоти (1х, 16х, 64х).

**ПРИКЛАД**: якщо ТхС – 110Гц (1х) , то швидкість передачі 100 бод

ТхС – 1,76кГц (16х) , то це кратно 16 бод

ТхС – 7,01 кГц (64)

**DSR** - (Data Set Ready) - готовнiсть модему. Вхiдний сигнал DSR є сигналом загального призначення. Його стан може бути перевiрений за допомогою ЦП i регiстра стану. Вхiд звичайно використовується для перевiрки стану готовностi модему.

**DTR** - (Data Terminal Ready) - основна функцiя включить або виключити лiнiю. Програмно вiн може бути установлений в стан логічного 0 за допомогою команди керування. Вхiдний сигнал DTR використовується для керування i перевiрки готовностi термiналу, а також вибору швидкостi.

**RTS** - (Request To Send) - основна функцiя: переключить модем на прийом або передачу. Програмно вiн може бути установлений в стан логічного 0 за допомогою вiдповiдного розряду в командi керування.

**CTS** - (Clear To Send) - характеризує стан комутатора. Стан логічного 0 на цьому входi дозволяє УСАПП передачу даних, якщо розряд TXEN у команді керування знаходиться в 1.

**Блок управління прийомом**

**RxRDY** - готовнiсть приймача передати символи даних. Цей вихiд показує, що УСАПП має символ, готовий до передачi в ЦП. RxRDY може бути використаний в структурi органiзацii перерви або при органiзацii опитування. RxRDY автоматично встановлюється в 0 пiсля передачi символа в ЦП.

**RхC/** - iмпульс тактування приймача, керує швидкiстю прийому символа.

**SYNC** - знаходження синхроiмпульса. Використовується тiльки в синхронному режимi.

**D0-D7** – лiнiї даних.

**RESET** - гашення.

**CLK** - тактовий iмпульс.

**СD/** - лог. 0 - на шинi даних данi; лог. 1 - команда.

**RD/** - читання або виведення.

**WR/** - запис або введення.

**CS/** - вибiр корпусу.

**TхD** - вихiд даних передавача.

**RхD** - вхiд даних приймача.

**3.3 Програмування послідовного адаптера**

Функціональне призначення УСАПП визначається засобами програмного забезпечення. Для цього ЦП передає в УСАПП ряд керуючих слів після установки його в початковий стан.

Керуючi слова роздiляються на двi групи:

- команда режима;

- команда керування;

Команда режиму слiдує вiдразу за встановленням в початковий стан, тобто після команди RESET. Команди. керування повиннi йти за командою режиму або символами SYNC.

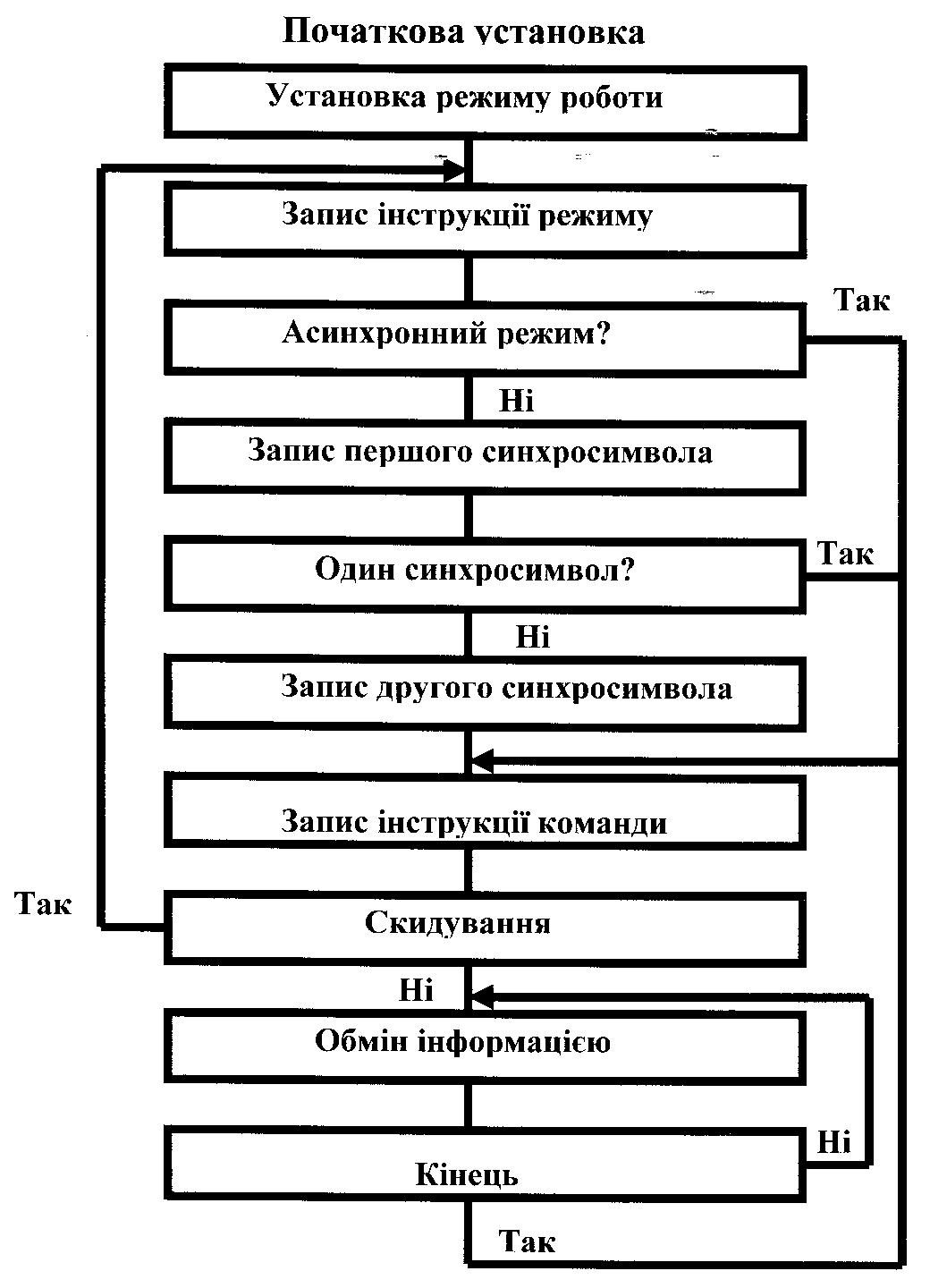


Рисунок 3.5- Послідовність програмування УСАПП

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регистр зсуву | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр приймача | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регистр зсуву | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр передавача | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр SYN1 | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр SYN2 | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр режиму | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Регістр наказу | | | | | | | |
|  |  |  |  |  |  |  |  |

7

0

|  |
| --- |
| Регістр приймача |

7

0

RxD

Керування

TxD

Керування

ШД ША ШУ Модем

Рисунок 3.6-Програмна модель адаптеру І8251

**Команди та сигнали керуванання**

Основні сигнали керування роботою УСАПП подаються на блок керування читанням/записом від МП і визначають вид оброблюваної інформації й напрямок передачі відповідно до табл.4.1.

Таблиця 4.1. Операції, обумовлені сигналами керування від МП

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Операція | Сигнали керування | | | |
| C/D | RD | WR | CS |
| Читання даних з УСАПП на D(7-0) | 0 | 0 | 1 | 0 |
| Запис даних з D>(7-0) в УСАПП | 0 | 1 | 0 | 0 |
| Читання слова стану з УСАПП на D(7~0) | 1 | 0 | 1 | 0 |
| Запис керуючого слова D(7-0)в УСАПП | 1 | 1 | 0 | 0 |
| Відключення УСАПП від D(7-0) | X | 1 | 1 | 0 |
| Те ж | X | X | X | 1 |
| Примітка X — будь-який стан сигналу. | | | | |

Команди керування можуть бути записанi в будь-який час роботи. Для повернення до команди встановлення нового режиму використовують регiстр команди керування. Якщо IR у командi керування встановити в 1, то вiдбудеться повернення до команди режиму.

Команда режиму - це команда, яка визначає основнi робочi характеристики адаптера.

Команда керування - виконує слiдуючi функцii:

а) встановлення операцii вводу або виводу;

б) скид триггерiв флагiв помилок;

в) керукання модемом.

Читання стану

Ця операцiя дозволяе ЦП в довiльний час роботи читати стан адаптера з метою виявлення помилок , а також сканування запитів зовнiшнiх пристроiв. Читання стану виконується, якщо сигнал CD встановлений в стан логiчноi 1.

Передача/прийом даних.

Пiсля того як слово режиму запрограмує потрiбний режим адаптера і при необхiдностi будуть завантаженi один або два синхроiмпульси – адаптер готовий до обмiну даними. Завантаження вiдповiдного керуючого слова визначає режим передачі або прийому iнформацii адаптером. Рiвень лог. 1 на TxRDY cигналiзує ЦП про те,що адаптер готовий до прийому сигналу. Пiсля запису символу в адаптер рiвень TxRDY встановлюеться в лог. 0. Адаптер може також приймати послiдовнi данi вiд модему або пристрою в/в . По завершенню прийому адаптер встановлює на RxRDY рiвень лог. 1, що служить сигналом для ЦП про готовнiсть адаптера передати йому цей символ. Адаптер не може почати передачу до тих пiр, поки розряд TXEN у команді керування не встановлений в лог. 1 i не отриманий сигнал гашення даних.

Асинхронний режим (передача).

При передачi даних адаптер до перетвореного послiдовного коду слова даних додае спочатку стартовий бiт, а в кiнцi стоповий. Крiм того якщо контроль парностi передбачений у ком. режиму, перед стоп-бiтом встановлюеться бiт перевiрки на парнiсть, або непарнiсть. Передача даних здiйснюеться через вихiд TXD. Послiдовнi данi видаються передавачем на вихiд по спаду TXC/ з частотою 1:1,1:16 або 1:64 вiд частоти TXC/. Вихiд TXD пiсля передачi слова даних i при вiдсутностi слiдуючого символа переходить в стан лог. 1, поки новi данi не надiйдуть вiв ЦП. У команді керування передбачена можливiсть переводу виходу TXD,при вiдсутностi даних, у стан лог. 0.

D7 D6 D5 D4 D3 D2 D1 D0

B1

B2

L1

L2

EP

S1

S2

PEN

**1**

**0**

**1**

**0**

**0**

**0**

**1**

**1**

**1:64**

**1:16**

**1:1**

**Синхрон. режим**

**Довжина символу**

0

0

0

5 біт

1

6 біт

7 біт

1

0

8 біт

1

1

**Дозвіл контролю**

**1 = дозвіл**

**0 = заборона**

**Вид контролю**

**1 = парність**

**0 = непарність**

**Число стопових біт**

**1**

**0**

**1**

**0**

**0**

**0**

**1**

**1**

**2**

**біти**

**1,5**

**біт**

**1 біт**

**Не визначено**

Рисунок 3.7-Формат команди асинхроного режиму

D7 D6 D5 D4 D3 D2 D1 D0

8

біт

7

біт

6

біт

5

біт

1

1

1

0

0

1

0

0

SCS

ESD

EP

PEN

L2

L1

O

O

**Довжина символу**

**Дозвіл контролю**

**0 = дозволено**

**1 = заборонено**

**Вид контролю**

**1 = парність**

**0 = непарність**

**Режим зовнішньої**

**синхронізації**

**1 = являється входом**

**0 = являється виходом**

**Кількість символів**

**синхронізації.**

**1 = один SYNC** **символ**

**2 = два SYNC символ**

Рисунок 3.8-Формат команди синхроного режиму

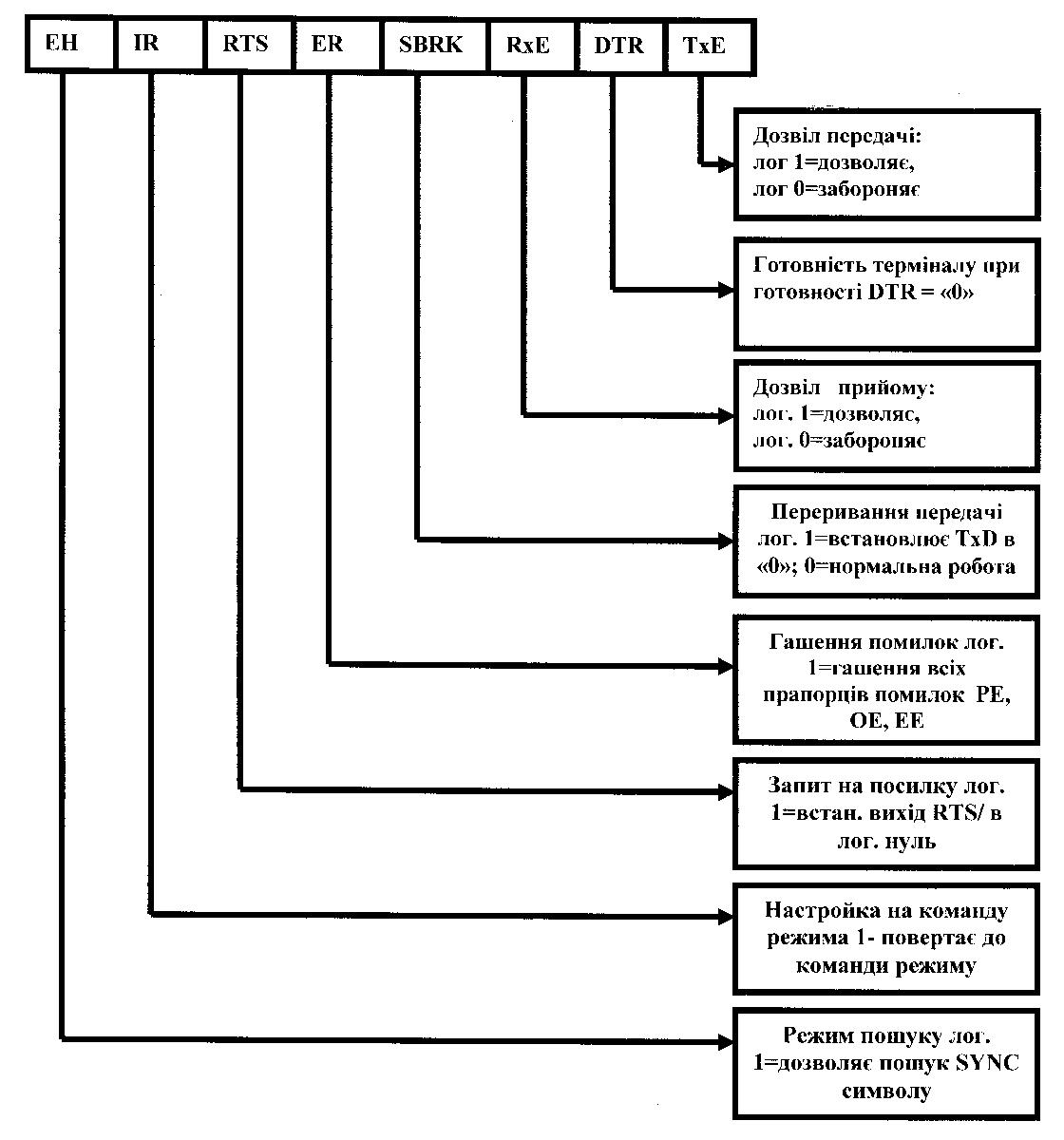


Рисунок 3.9-Формат команди керування

**DSR**

**SYNDET**

**FE**

**OE**

**PE**

**TxE**

**RxRDY**

**TxRDY**

**Помилка по парності. При виявленні помилки прапорець РЕ встановлюється в 0.**

**Гашення його проводиться розрядом ER команди управління РЕ не забороняє роботу К580ИК51**.

**Помилка по переповненню. Прапорець DE встановлюється в «1», коли ЦП зчитує наступний символ, не отримавши попереднього. Гашення його проводиться розрядом ER команди управління. DE не забороняє роботу К580ИК51, але, попередній символ губиться**.

**Помилка формату (асинхронний режим). Прапорець FE встановлюється в «1», коли в кінці символа не зчитується СТОП – БИТ. Гасится разрядом ER команди керування і не впливає на роботу К580ИК51**.

**Відповідає зовнішнім виводам**

Рисунок 3.10-Формат слова стану

Асинхронний режим (прийом).

Рiвень на входi RXD знаходиться в лог. 1. Спад сигналу на цьому входi свiдчить про появу старт-бiта.

Достовiрнiсть цього бiта контролюеться повторним тробуванням.Повторне знвходження рiвня лог. 0 свiдчить про достовiрнiсть стартового бiта. При цьому по спаду RXD/ запускаеться лiчильник, що вiдлiковує задану програмно довжину слова даних, бiт парностi i стоп-бiти. Данi приймаються в послiдовному кодi по фронту RXC/. При знаходженнi помилки по парностi або непарності встановлюється флаг помилки . Пiсля прийняття стоп-бiта адаптер здiйснюе передачу прийнятого символу у паралельному кодi у буфер даних для передачi у ЦП. При цьому сигнал RxRDY встановлюється в лог. 1 i поточна iнформацiя записується у буфер,стираючи попередню. Всi флаги помилок скидаються за допомогою команди керування.

Вихiд передавача

(TXD) ─────┐старт- ┌──── ─────┬──────────┬──── ─────┐

маркер │ бiт │ данi... │ контроль │стоп-бiти │

└───────┴───── ────┴──────────┘ └─────

Вхiд приймача

(RXD) ──────┐старт- ┌──── ────┬──────────┬──── ─────┐

маркер │ бiт │ данi... │ контроль │стоп-бiти │

└────────┴──── ────┴──────────┘ └──────

Вихiд послiдовних даних (TXD)

┌────────────┬───── ────┬───────────┬────── ──────┐

│cтарт -бiт │данi... │ контроль │ стоп-бiти │

└────────────┴───── ────┴───────────┴────── ──────┘

Вхiд послiдовних даних (RXD)

┌───────────┬───── ─────┬───────────┬────── ─────┐

│старт-бiт │ данi... │ контроль │ стоп-бiти │

└───────────┴───── ─────┴───────────┴────── ─────┘

Адресацiя портiв І8251

Адаптер І8251 збуджується при рiвнi лог. 0 на контактi вибору корпусу, який з'являється коли адреса порту в/в на адресних лiнiях знаходиться в дiапазонi EC..EF. Для адресацii використовуються розряди з 2-го по 7-й.

Молодший розряд адреси А0 керує входом CD.

┌─────────────────────┬────────────┬────────────┬──────────────────┐

│ адреса пристрою в/в │ команда │ функцiя │ направлення │

├─────────────────────┼────────────┼────────────┼──────────────────┤

│ ED або EF │ вивiд │ команда │ ЦП -> УСАПП │

├─────────────────────┼────────────┼────────────┼──────────────────┤

│ EC або EE │ вивiд │ данi │ ЦП -> УСАПП │

├─────────────────────┼────────────┼────────────┼──────────────────┤

│ ED або EF │ ввод │ стан │ УСАПП -> ЦП │

├─────────────────────┼────────────┼────────────┼──────────────────┤

│ EС або EE │ ввод │ данi │ УСАПП -> ЦП │

└─────────────────────┴────────────┴────────────┴──────────────────┘

Органiзацiя переривання за доп. І8251.

Для органiзацii перерви можна використати вихiд готовностi приймача RxRDY. Крiм цього можна використати вихiд готовностi передавача TxRDY, a також вихiд TXE.

TxRDY - лог. 1 коли адаптер готовий до прийому символiв вiд ЦП.

RxRDY - лог. 1 коли адаптер мае символ готовий до передачi в ЦП.

TXE - лог. 1 коли в буферi вiдсутнi данi.

Виходи TxRDY, TXE керуються бiтом дозволу прийому у ком. керування.

Комутацiя каналiв прийому/передачi.

Програмований послiдовний iнтерфейс забезпечуе передачу i прийом iнформацii . Вибiр пристроїв, з якими здійснюється прийом /передача даних визначається завдяки програмно керуємим рiвням DTR i RTS адаптера.

┌──────────────┐

──┤ ├── B0 ──┤ ────── канал 1

DTR ├───────┤ A0 B1 ──┤ ────── канал 2

──┤ ├── B2 ──┤ ────── канал 3

RTS ├───────┤ A1 B3 ──┤ ────── канал 4

──┤ ├── │

└──────────────┘

**Приклад програмування ASM-80**

DI заборона переривань

Mvi A,B6h; налаштування таймера

Out DFh; 3 режим, лічильник 2

Mvi A,80h; ввод молодшої частини константи

Out Deh; по адресі

Mvi A, 04h; старшої частини , швидкість 50 бод/сек

Out Deh; по адресі

XRА A; зкид буфера адаптера обнулити А, очистити буфер

Out EDh

Out EDh в буфер посилаємо ноль

Out Edh зкид буфера адаптера

Out Edh часова затримка

M1: IN Edh; перевірка очистки буфера

Ani 04h; ТхЕ=1

jz m1;

mvi A,40h; налаштування на режим IR=1

out edh;

mvi A,4eh; режим стоп-біт 1, без контролю

out Edh; 8 біт інформації, кратність 1\16

Mvi A,15h; команда СІ дозвіл передачі та прийому

Out edh

EI дозвіл переривань

М2: IN edh читати стан регістру

ani 01h D0 в слові стану, якщо вихідний буфер пустий і

готовий до передачі символів TxRDY=1

JZ m2

M3:Lxi h,mas завантаження масиву

Lda len довжина масиву

Mov A,C встановлення лічильника

Out EC

Mov a,m

Out edh

Inx h

Dcr c

Jz m3

Прийом даних

M4: IN edh

ani 04h

jz m4

IN ech

Mov M,A

OUT ECH

HLT

ADR : DB FFh

END

**Обмiн iнформацiею мiж двома МП комплектами у послiдовному** кодi

Прийомо-передавач, побудований на основi послiдовного iнтерфейсу І8251, який призначений для реалiзацii двунаправленого синхронного/асинхронного обмiну даними перетвореними у послiдовний формат, який е бiльш стiйкий до перешкод нiж паралельний формат. Унiверсальний с/а прийомо-передавач перетворює паралельний код, отриманий iз системи по ШД у послiдовний i посилае його у канал зв'язку (КЗ). УСАП виконуе i зворотне перетворення: приймае з лiнii зв'язку послiдовний код i формуе iз нього паралельний, який може бути переданий в систему по ШД. За допомогою мiкросхеми І8251 забезпечується функцiонування каналу зв'зку з одно- або двонаправленим засобами передачi iнформацii ( полудуплексний, дуплексний обмiн). Унiверсальнiсть мiкросхеми К580ВВ51 виявляеться в тiм, що завантаженням у неi керуючих слiв i слiв режиму програмуеться один iз можливих видiв обмiну ( синхронний або асинхронний ), довжина передаваемих символiв, контроль по парностi, швидкiсть передачi, кiлькiсть стоп. бiт, а також синхросимволiв i вид синхронiзацii.

Призначення i схемотехнiчний склад прийомо-передавача

Прийомо-передавач призначений для передачi даних мiж двома МП комплектами у послiдовному кодi, причому кожний МП-комплект може робити як на передачу даних так i на прийом. Виконанi на базi 2 УМК, кожний iз яких являе собою закiнчену мiкроЕОМ. Прийомо-передавач працюе у асинхронному режимi i його функцiонування здiйснюеться як за допомогою програмного опиту, так i за допомогою контролера переривань. Прийомо-передавач є розширенням мiкроЕОМ за допомогою плати ТЕЗ, на яку виведенi ША, ШД, ШУ. На платi ТЕЗ використалi слiдуючi мiкросхеми:І8251, І8253, І8255. Схема "або" використована для синхронiзацii передавача i приймача.

Передача iнформацii

Передача iнформацii роздiляеться на 2 етапи:

1) обмiн мiж ЦП i УСАП;

2) обмiн мiж УСАП i КЗ.

1. Якщо регiстр блоку передачi пустий тобто по закiнченi передачi у канал зв'язку на виходi TXE встановлена логiчна 1, то ця 1 поступае на вхiд IRO контролера переривань. Iз передавача за допомогою команди OUT завантажуеться у регiстр вводу УСАП по ШД число у паралельному кодi, яке перевантажуеться у здвиговий регiстр, де перетворюеться у послiдовний код для передачi у КЗ. При запису числа в УСАП iз ЦП сигнал на виходi TXE скидаеться в 0. При цьому в УСАП в команду керування завантажуеться 1 в розрядi D0 (дозвiл передачi). У разi програмного опиту ЦП постiйно опитуе регiстр команди читання стану УСАП i видiляе розряд D0 (TXRDY -готовнiсть передавача прийняти iз ЦП символ даних), який встановлюеться в 1, коли регiстр блоку передачi пустий.

2. Для видачi коду iз здвигового регiстру блока передачi у канал зв'язку на входi CTS/ (дозвiл передати iнформацiю) треба мати сигнал низького рiвня (0). Цей вхiд використовуеться для синхронiзацii роботи приймача i передавача. Для зв'язку УСАП i КЗ у команду керування УСАП завантажуеться одиниця в D5 (RTS/ в 0 - запит на посилку). В результатi на виходi RTS/ з'являеться 0, який поступае на вхiд DSR/ приймача. Якщо на вхiд CTS/ передавача поступае 0 з приймача, то здiйснюеться передача кода з УСАП в КЗ.

Прийом iнформацii

Подiляеться на 2 етапи:

1) обмiн мiж ЦП i УСАП;

2) обмiн мiж УСАП i КЗ.

1. По сигналу RXRDY = 1, який поступае на вхiд IR1 контролера переривань, процесор за допомогою пiдпрограми приймача читае iнформацiю з регiстра блоку прийому УСАП.

2. Обмiн мiж КЗ i УСАП - сигнал RTS/, поступаючий з виходу передавача на вхiд DSR/ приймача (готовнiсть модему). Встановлюе D7 в словi стану у логiчну 1 i на виходi DTR/ ( готовнiсть термiналу прийняти iнформацiю ) встановлюеться 0. Сигнал DTR/ разом з сигналом RXRDY приймача поступае на схему "I", а ii вихiд пiдключений на вхiд CTS передавача, який в разi DTR i RXRDY = 0 передає iнформацiю у канал зв'язку. По закiнченню прийому вихiд RXRDY приймача встановлюеться в логiчну 1 i сигнал CTS/ стае рiвним логiчнiй 1. В разi програмного опиту кiнець прийому фiксуеться по розряду D1 DTS/ (слово керування). Стан DTR скидуеться в 1 керуючим словом D1=0.

Програмне забезпечення прийому/передачi iнформацii.

Програма прийому/передачi будуеться iз 6 частин. Першi 3 частини являють собою основну програму загальну для приймача i передавача. Вона являе собою програмування таймера для того, щоб з виходiв OUT0 i OUT1 подавати синхросерii iмпульсiв на входи TXC i RXC/ УСАП.

Друга частина: програмування контролера переривань для того, щоб задати молодшi i старшi байти адреси пiдпрограми обробки переривань.

Третя частина - програмування режимiв роботи УСАП (кiлькiсть iнформацiйних символiв, швидкiсть, ...).

Четверта частина - пiдпрограма роботи передатчика.

П'ята i шоста - пiдпрограма роботи приймача.

Функцiональна схема обмiну мiж передавачем i приймачем

**УСАПП-1**

ТхD

RxD

DTR/

DSR/

RTS/

CTS/

TxRDY

RxRDY

TxE

**УСАПП-2**

ТхD

RxD

DTR/

DSR/

RTS/

CTS/

TxRDY

RxRDY

TxE

**Контролер**

**Переривань**

IRQ1

IRQ2

1

1

1

1

Контрольні питання та завдання

1. Архітектура ВІС послідовного інтерфейсу І8251. Структурна схема, призначення основних блоків і вхідних/вихідних сигналів.

2 Інструкції режиму, команди, слово стан. Режими роботи послідовного інтерфейсу І8251. Підключення до шин МП.

3.Розробіть програми вводу та виводу інформації

4.Поясніть синхронізацію прийому даних в синхроному та асинхроному режимах.

5.Чому з збільшенням швидкості передачі збільшується вірогідність похибок даних, які приймаються?

6.Намалюйте структурну схему УСАПП та поясніть режими роботи.

7.Намалюйте схему підключення УСАПП до мікропроцесорної шини.

8.Поясніть послідовність операцій ініціалізації УСАПП.

9. На лабораторній напишіть програми прийому та передачі інформації.

**4 Програмований контролер переривань І8259**

**4.1 Призначення і структура**

В мікропроцесорній системі необхідно щоб пристрої вводу/виводу такі як клавіатура, дисплей, датчики та інші компоненти мали ефективне обслуговування.

Найбільш поширений метод обслуговування таких пристроїв є опитування. ЦП послідовно перевіряє кожний пристрій , чи потребує він обслуговування. Таким чином більша частина основної програми буде складатися з безперервного циклу опитування, що негативно діє на пропускну спроможність системи.

Більш корисним буде метод, який дозволяє мікропроцесору виконувати головну програму і зупинятися тільки на обслуговування периферійних пристроїв по запиту. Цей метод дозволяє проінформувати процесор про необхідність завершити операції які виконуються в цей час і отримати запит на обслуговування периферійного пристрою.

Після процедури обслуговування пристрою процесор повернеться до виконання програми в тому місці де вона була зупинена. Це називається перервою. При цьому пропускна спроможність системи набагато збільшується.

Програмуємий контролер переривань отримує запити від периферійних пристроїв, визначає які з вхідних запитів мають пріоритет вищего значення, чим той що обслуговується в цей час .

**4.2 Основні функції контролера**

Оскiльки I8080 обслуговує тiльки один запит на переривання, що надходить на потенцiйний вхiд INT,то очевидно, що обслуговувати бiльшу кiлькiсть переривань без апаратноi пiдтримки неможливо. Функцii такої пiдтримки i покликаний виконувати контролер переривань. Контролер переривань обслуговує вiсiм запитiв на переривання i має вбудованi апаратнi засоби для збільшення кiлькостi запитiв до 64 (64=8\*8), шляхом каскадування, з iншими аналогiчними контролерами . Порядок обслуговування переривань задається програмно. Кожен запит на переривання може бути заборонено/дозволено програмним маскування.Програмований контролер переривань (ПКП, Programmable Interrupt Controller, PIC) реалізує векторну систему переривань.

Мікросхема І8259А фірми Intel , а так само її модифікації І8259А-2 і І8259А-8, підтримує 8 рівнів переривань від восьми різних пристроїв. Це каскадуємий до 64 векторних пріоритетних переривань, призначена для мінімізації програмного забезпечення і реальних додаткового часу для забезпечення багато-рівневих пріоритетних переривань. ВІС має декілька режимів, дозволяючи виконати оптимізацію системи. Блок схема контролера приведена на рис.4.1.

**INTA \ INT**

**Схема керування**

**Буфер лінії даних**

**Блок управління читан./записом**

**Блок управління компаратором**

**Регістр маски переривання**

**(IMR)**

**Регістр обслуго-**

**вуван.**

**переривань.**

**(ISR)**

**Регістр дільник**

**Пріоритетів**

**Регістр запитів переривань**

**(IRR)**

**D0-D7**

**RD\**

**IR0**

**WR\**

**A0**

**IR7**

**CS\**

**CS0**

**CS1**

**CS2**

**SP\**

Рисунок 4.1- Блок схема ПКП

**D0.D7** - лінія даних для взаємодіїй з ЦП . В великих системах можливо будуть необхідні шині драйвери, а в малих системах достатньо прямих з’єднань.

**RD/ -** системний сигнал читання , дозволяє розмістити на шину даних зміст регістрів IMR, ISR або IMR або приорітетний рівень, який видається на шину, і залежить від стану контролера .

**WR/** - системний сигнали запису. , який дозволяє прийняти дані з шини даних і використати їх для встановлення бітів в словах наказів.

**A0** - вхiд вибору адреси; цей вхідной сигнал використовується сумісно з WR та RD для запису команд в різні командні регістри, а також для зчитування різних регістрів стану мікросхеми. Ця лінія може підключатися безпосередньо до однієї з адресних ліній.

**CS/** - чіпселект; вибір корпусу ВІС

**CASO…CAS2** - лінії виходів каскадування, для ідентифікації конкретного підлеглого контролера .

**SP** - Виконує дві функції. Як вхід, визначає, яким є контролер: ведучим (SP#/EN# == 1) або підлеглим (SP#/EN# == 0). Як вихід, виконує заборону приймачів та передатчиків шини даних, коли дані передаються з контролера до ЦП.

**INT/** - вихід запиту переривання на процесор

**ІNTA/** - вхід підтвердження отримання переривання процесором

**IR0...IR7** - входи запиту

Основні функції контролера:

1) фіксація запитів на переривання від восьми зовнішніх джерел;

2) програмне маскування запитів, що поступають;

3) привласнення фіксованих або циклічно змінних пріоритетів входам контролера, на як поступають запити;

4) ініціація виклику процедури обробки апаратного переривання, що поступило.

Кількість обслуговуваних зовнішніх джерел переривань може бути збіль-шена шляхом каскадування декількох контролерів.

До складу контролера входять:

- схема управління читанням/записом;

- схема управління;

- схема каскадування;

- регістр запитів на переривання;

- схема обробки пріоритетів;

- регістр стану;

- регістр маскування запитів на переривання.

ПКП може знаходитися в двох основних станах: налаштування і обслуговування запитів на переривання. В стані налаштування контролер приймає керуючі слова ініціалізації (Initialization Command Words, ICW), в стані обслуговування операційні керуючі слова (Operation Control Words, OCW).

**4.3 Блок схема програмованого контролера переривань**

Контролер переривань обслуговуватє 8 запитів переривань і має вбудовані апаратні засоби для збільшення кількості обслуговуваних переривань до 64 шляхом підключення додаткових контролерів. Порядок обслуговуваних переривань задається програмно. Кожний запит переривання може бути заборонений (дозволений) програмним маскуванням. Порядок обслуговування поривань і маскування може динамічно змінюватися в довільний час.

**Регістри INTERRUPT REQUEST REGISTER (IRR) і IN-SERVICE REGISTER (ISR)**

Запити переривань на входах IR керуються двома послідовно сполученими регістами обслуговуваних переривань (IRR і ISR). IRR призначений для зберігання усіх запитів переривань ,які запитують обслуговування. ISR призначений для зберігання усіх запитів переривань , які обслуговуються. По фронту сигналу ( запит переривань на вході IR встановлюється розряд регістру IRR рівня лог. «1» на лінії INT. В регістрі IRR може бути встановлено і декілька розрядів, якщо вони не замасковані. Гашення IRR виконується послідовністю імпульсів /INTA від ЦП. Встатовлення розрядів ISR імпульсом /INTA в той час, коли гаситься розряд в IRR. Цей розряд утримується на протязі усього часу роботи програми, до того часу поки контролер на отримає команду ЕОІ ( кінець переривання).

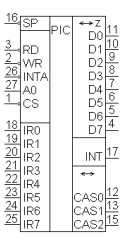


Рисунок 4.2-Корпус мікросхеми І8259

**Розподілювач пріоритетів PRIORITY RESOLVER**

Розподілювач пріоритетів визначає пріоритети розрядів, які встановлені в регістрі IRR. Вибирається розряд найвищого пріоритету і за допомогою імпульсу /INTA встановлюється відповідний розряд в регістрі ISR. Вивід INT (переривання) безпосередньо зв’язаний з відповідним входом ЦП. Вхід /INTA (підтвердження переривання) зв’язаний з регістром стану ЦП. Регістр стану виробляє три різних імпульса /INTA. За допомогою ціх імпульсів контролер видає на шину даних 3-х байтну команду CALL. Ця команда забезпечуэ перехід ЦП до програми обслуговування переривань в відповідності з алгоритмом пріорітетів і в відповідності з умовами запитів переривань.

**Регістр маски INTERRUPT MASK REGISTER (IMR**)

Регістр маски переривань (IMR) зберігає розряди запитів переривань, які повинні бути замасковані. Цей регістр працює на регістри IRR і ISR. Маскування розрядів з високим пріоритетом не впливаэ на запити переривань з низьким пріоритетом.

**Буфер даних DATA BUS BUFFER**

Буфер шини даних , двонаправленний , 8-розрядний , з тристабільним станом використовується для зв’язку з контролером і передачі команд керування або зчитування слова стану.

**Блок керування операціями зчитування запису READ / WRITE CONTROL LOGIC**

Цей блок забезпечує прийом команд з ЦП. Він має регістри команд ініціалізації (ICW) і робочіх команд (OCW), які зберігають різні керуючі слова , які забезпечують роботу контролера. Крім цього , цей блок дозволяє виводити на шину даних інформацію про стан контролера.

**Сигнал вибору кристалу CS (CHIP SELECT)**

Рівень лог. «0» на вході /CS дозволяє функціонування контролера .

**Сигнал –запис даних WR (WRITE**)

Рiвень лог.0 на входi WR/ дозволяє ЦП провести запис команд ICW i OCW в контролер.

**Сигнал- читання даних RD (READ)**

Рiвень лог.0 на входi RD/ дозволяє контролеру засилати iнформацiю стану регiстру запитiв переривань (IRR), регiстру обслуговуємих переривань (ISR), регiстру маски переривання (IMR), або двійково-десятковий код номеру старшого пріорітету переривання на шині даних.

**Сигнал на вході А0**

Вхiдний сигнал А0 використовується в сполученнi з сигналами на входах: WR/ i RD/ для запису команд в рiзнi регiстри команд, або читання iнформацii стану рiзних регiстрiв контролера.Вхiд А0 може бути безпосередньо пов'язаний з однiєю iз адресних лiнiй МП.

**Сигнал /SP**

Вхiд SP/-сигнал пiдлеглого контролера, забезпечує збiльшення кiлькостi запитiв переривання до 64, шляхом використовування декiлькох контролерів переривань (каскадування). В цьому разi один контролер є головним (SP1), iншi пiдлеглими (SP0).

**Блок буфера компаратора THE CASCADE BUFFER/COMPARATOR**

Блок буфера компаратора призначений для зберiгання i зрiвняння iдентифiкаторiв усiх контролерiв. Три контакта входу/виходу (CAS0.-.CAS2) блоку є виходами, коли контролер є головним, або входами, коли контролер є пiдлеглим. Головний контролер посилає iдентифiкатор CAS0-CAS2 для вибору пiдлеглого контролера. Вибраний контролер посилає адресу своєї пiдпрограми на шину даних протягом слiдуючих двох iмпульсiв /INTA.

**4.4** **Принцип роботи контролера**

При здiйсненнi передачi даних з перериванням програми виконується слiдуюча послiдовнiсть дiй:

- встановлюється рівень лог. «1» на одній або декількох ліній запиту переривання сигналізуючи контролеру , що зовнiшнiй пристрiй потребує обслуговування запиту переривання;

- контролер приймає ці запити, розподіляє пріоритети і посилає сигнал INT до ЦП;

- контролер підтверджує отримання сигналу INT посилкою сигналу /INTA;

- при отриманні сигналів /INTA з регістру стану ЦП контролер виставляє на шину даних код команди CALL (11001101);

- по цій команді виконується видача ще двох імпульсів сигналу /INTA;

- ці імпульси дозволяють контролеру виставити на шину даних адресу підпрограми. По першому імпульсу виконується видача восьми молодших розрядів адреси , по другому –старших розрядів;

- цим завершується 3-х байтна команда CALL, створена контролером.

Регістр обслуговуємих переривань не гаситься до завершення програми, коли в контролер поступає команда ЕОІ ( кінця переривань).

**4.4.1 Режими роботи**

**1.Режим фіксованих пріоритетів (Fixed Priority, Fully Nested Mode).**

В цьому режимі контролер знаходиться відразу після ініціалізації. Запити переривань мають жорсткі пріоритети від 0 до 7 (0 - вищий) і обробляються відповідно до пріоритетів. Переривання з меншим пріоритетом ніколи не буде оброблено, якщо в процесі обробки переривань з більш високими пріоритетами постійно виникають запити на ці переривання.

**2. Автоматичний зсув пріоритетів ( Automatic Rotation ).**

В цьому режимі дається можливість обробити переривання всіх рівнів без їх дискримінації. Наприклад, після обробки переривання рівня 4 йому автоматично привласнюється низький пріоритет, при цьому пріоритети для всієї решти рівнів циклічно зсовуються і переривання рівня 5 матимуть в даній ситуації вищий пріоритет і, отже, можливість бути обробленими.

**3. Програмно-управляючий зсув пріоритетів( Specific Rotation ).**

Програміст може сам передати команду циклічного зсуву пріоритетів ПКП, задавши відповідне управляюче слово. В команді задається номер рівня, якому вимагається привласнити максимальний пріоритет. Після виконання такої команди пристрій працює так само, як і в режимі фіксованих пріоритетів, з урахуванням їх зсуву. Пріоритети зсовуються циклічно, таким чином якщо максимальний пріоритет був призначений рівню 3, то рівень 2 отримає мінімальний і оброблятиметься останнім.

**4 Автоматичне завершення обробки переривання (Automatic End Interrupt, AEOI).**

В звичайному режимі роботи процедура обробки апаратного переривання повинна перед своїм завершенням очистити свій біт в ISR спеціальною командою, інакше нові переривання не оброблятимуться ПКП. В режимі AEOI потрібний біт в ISR автоматично скидається в той момент, коли починається обробка переривання потрібною процедурою обробки і від неї не потрібен видавати команду завершення обробки переривання (EOI). Складність роботи в даному режимі обумовлюється тим, що всі процедури обробки апаратних переривань повинні бути тими, що повторно входять, оскільки за час їх роботи можуть повторно виникнути переривання того ж рівня.

**5. Режим спеціальної маски (Special Mask Mode).**

Даний режим дозволяє відмінити пріоритетне впорядкування обробки запитів і обробляти їх у міру надходження. Після відміни режиму спеціальної маски попередній порядок пріоритетів рівнів зберігається.

**6. Режим опиту (Polling Mode).**

В цьому режимі апаратні переривання не відбуваються автоматично. Поява запитів на переривання повинна визначатися прочитуванням IRR. Даний режим дозволяє так само отримати від ПКП інформацію про наявність запитів на переривання і, якщо запити є, номер рівня з максимальним пріоритетом, по якому є запит

**4.5 Програмування контролера**

Програмування ВН59 здійснюється двома типами керуючих слів: ICW (Initialization Command Word) і OCW (Operation Command Word). На рис. 4.3 представлена послідовність операцій при ініціалізації.

Три керуючих слова ICW1-ICW3 завантажуються перед початком роботи і встановлюють ВІС у режим фіксованих пріоритетів.

Для оперативного управління роботою контролера в будь-який час у нього можуть бути завантажені команди керування OCW1-OCW3.

Можливі слідуючі режими:

* режим повного вкладення;
* режим циклічного пріоритету;
* режим спеціального маскування;
* режим опитування.

**4.5.1 Команди ініціалізації**

Контролер використовує три команди ініціалізації ICW1, ICW2 і ICW3.

Команда ініціалізації, в якій біт керуючого слова D4=1 при А0=0 інтерпретується як команда ICW1. Після її отримання в контролері запускається послідовність ініціалізації, тобто автоматично виконуються слідуючі операції:

* гасяться всі тригери. Тому перехід в стан лог. «1»якого небудь входу (IR) після ініціалізації генерує запит переривання;
* гаситься регістр маски переривання;
* входу IR7 присвоюється пріоритет 7;
* гасяться тригери читання стану і регістр спеціального маскування.

Дві перших обов'язкових команди ICW1 і ICW2 визначають базову адресу таблиці переходів (рис. 4.7). Адресний інтервал 4 або 8 задається бітом F в ICW1. Початкові адреси підпрограм обслуговування формуються за схемою: addr = base + k × N, де k = 4 або 8, N - номер запиту, прийнятого до обслуговування. У складі ICW1 біт S визначає відсутність або наявність каскадування ВН59. У каскадному варіанті завантажується ще команда ICW3 у ведучий і підлеглий КП. ICW3 для провідного КП містить 1 в розрядах, відповідних входів запитів, до яких підключені виходи INT підлеглих КП. ICW3 для підлеглого КП містить код номера входу ведучого, до якого він підключений.

Після ініціалізації контролер готовий до роботи в режимі фіксованих пріоритетів. Подальше керування роботою схеми здійснюється за допомогою команд OCW1 - OCW3, формати яких наведено на рис. 4.8.



S: S=1 –некаскадуємий контролер (один);S=0 –декілька контролерів

F: інтервал адреси, F=1 – 4 байти, F=0 – 8 байт

А7-А5: старші розряди молодшого байта адреси



А8-А15: старший байт початкової адреси обробки переривань

 для ведучого

Sі=1, якщо к відповідному номеру входу запита ведучого підключений ведений

для веденого

0,1,2-індефікатор веденого

Рисунок 4.7 –Формати команд ICW1-ICW3

**4.5.2 Робочі команди**

Робочі команди OCW

У довільний момент часу і незалежно від інших кожен запит може бути замаскованим командою OCW1. При встановленні 1 в розряді IMR забороняється прийом переривань з даного входу. Команда OCW2 служить для встановлення в 0 довільних розрядів ISR і циклічного зсуву пріоритетів з призначенням максимального значення будь-якому з восьми можливих рівнів. Для обслуговування запитів з рівними пріоритетами використовується циклічний зсув (команда EOI). При використанні команди зсуву одночасно зі скиданням ISR-біта, що має вищий пріоритет, реалізується циклічний зсув пріоритетів з призначенням нижщого щойно обслугованого рівня. Циклічний зсув не порушує послідовності вкладених один в одного переривань, що забезпечує правильне повернення з обслуговуючих їх програм. Пряма адресація рівня в слові OCW2 дозволяє скинути конкретний ISR-біт і таким чином завершити процедуру обслуговування цього запиту і циклічну зміну пріоритетів з явною вказівкою нижщого рівня. За допомогою OCW3 встановлюється режим спеціального маскування (див. рис. 4.6). У цьому режимі кожен біт в регістрі ISR забороняє тільки власний рівень, але дозволяє всі інші.

Командою OCW3 здійснюється керування режимом « поллінг» і вибір регістра IRR або ISR для читання його вмісту за допомогою програм. Режим ініціюється видачею в КП слова OCW3 з встановленням біта P (Polling). Контролер визначає наступний цикл читання при А0 = 0, як підтвердження переривання і видає на шину даних OCW3. За цим словом визначається запит з найвищим пріоритетом (рис. 4.9).



Маски переривань

Номер рівня

|  |  |
| --- | --- |
| 0 х 0  1 0 0  1 1 0  0 0 1  0 1 1  1 0 1  1 1 1 | Немає операції  Немає операції  Спеціальний зсув  ЕОІ  Спеціальний ЕОІ  ЕОІ з зсувом  Спеціальний ЕОІ  З зсувом |



|  |  |
| --- | --- |
| Немає  оперц.  Вибір IRR  Вибір ISR | 0 х  1 0  1 1 |

|  |  |
| --- | --- |
| 0 х  0 0  0 1 | Немає операції  Зкид спец.маскування  Спец.маскування |

Рисунок 4.8 – Формати OCW1-OCW3



Прапор запиту Рівень з найвищим приоритетом

Рисунок 4.9 – Формат даних при опитуванні

Вхід

Запис ICW1

Запис ICW2

Контролер один?

Запис ICW3

Запис ICW1

Запис ICW2

Запис ICW3

Необхідно

маскування?

Цикл. зсув приоритетів?

Необхідно маскування?

Режим полінгу?

Запис OCW3

Запис OCW1

Запис OCW2

Запис OCW3

Запис ІCW3

Ведучий

Ведучий

ні

Ведучий

так

Підлеглий

так

Підлеглий

ні

так

ні

так

Вхід в спецмаск.

Прогр.спецмаск.

ні

Для виходу

З спецмаскування

так

ні

Вихід

Рисунок 5.6 – Послідовність ініціалізації і керування режимами

Команда ICW1 мiстить 2 керуючих розряди i три розряда адреси команди CALL. Пiсля ICW1 подається команда ICW2, яка мiстить 8 старших розрядiв адреси команди CALL. Кожному входу КП встановлюеться вiдповiдна адреса пам'ятi, яка видається на лiнii у вiдповiдь на сигнал INTA/. Адреси можуть розмiщуватись з iнтервалом 4-8 байтiв.

Якщо кількість запитів більше 8, то застосовуються схеми каскадування контролерів . Один з контролерів отримує статус ведучого (вихід SP підключається через резистор до джерела 5В), а решта - підлеглих (виходи SP заземлюються). Вихідні лінії INT з підлеглих КП під'єднуються до входів запитів на переривання ведучого (рис. 4.10). Якщо ведучий КП, обробляючи запити, приймає на обслуговування запит, що надійшов з підлеглого, то після отримання першого імпульсу INTA встановлюється перший байт команди CALL. На шині каскадування CAS0 - CAS2 провідний КП зкидає код підлеглого, запит з якого прийнятий до обслуговування. Підлеглий КП, код якого відповідає коду на лініях каскадування, в другому і третьому циклах INTA встановлює молодший і старший байти початкової адреси підпрограми обслуговування переривання.

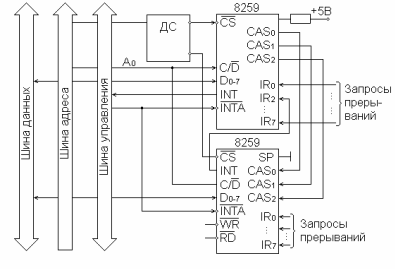


Рисунок 4.10 – Схема каскадування та підключення до шин

**Приклад ініціалізації на комплексі I8259, ASM-80**

Pic a0 20h

Pic a1 21h

Програмування КП

mvi a,00010110b ;ICW1 по 4

out 20h

mvi a,0h ;ICW2

out 21h

mvi a,……… ;OCW1 якщо потрібно

out 21h

mvi a,……… ;OCW2 якщо потрібно

out 21h

m1: головна програма

jmp m1

hlt

Таблиця переходів

org 0000h

call ir0 ; INT0

ret ; повернення з підпрограми

org 0004h

call ir1 ; INT1

ret ;повернення з підпрограми

org 0008h

call ir2

ret

org 000bh

call ir3

ret

org 000fh

call ir4

ret

org 0010h

call ir5

ret

org 0014h

call ir6

ret

org 0018h

call ir7

ret

Табдиця векторів

org 0100h

ir0: mvi b,0 ; номер вектора

ret

ir1: mvi b,1

;підпрограма

mvi a,20h ;OCW2 кінець переривання

out 20h ;

ret

ir2: mvi b,2

;підпрограма

mvi a,20h ;OCW2 кінець переривання

out 20h ;

ret

ir3: mvi b,3

ret

ir4: mvi b,4

ret

ir5: mvi b,5

ret

ir6: mvi b,6

ret

ir7: mvi b,7

ret

hlt

end

Контрольні питання та завдання

1 .. Поясніть, які можливості мають МП для організації обміну за запитами переривання (на прикладі МП ВМ80, ВМ85, ВМ86).

2. Яка організація радіальної системи переривань?

3. Поясніть переваги і недоліки прийому запитів на переривання у вигляді рівнів напруг (високого / низького) і фронтами сигналу.

4. Що розуміється під однорівневими і багаторівневими перериваннями?

5. Які способи формування початкового адреси підпрограм для обслуго-вування переривань вам відомі?

6. Наведіть приклади векторної 8-рівневої системи переривання.

7. Назвіть функції програмованого контролера І8259.

8. Назвіть керуючі слова ініціалізації І8259 та їх призначення.

9. Які функції виконують операційні управляючі слова OCW1-OCW3?

10. Зобразіть по пам'яті структурну схему контролера І8259 і поясніть призначення її компонентів.

11. Назвіть режими обробки запитів по пріоритетам контролера І8259.

12.На лабораторній написати програму ініціалізації контролера.

**5.ПРОГРАМOВАНИЙ ПАРАЛЕЛЬНИЙ IНТЕРФЕЙС ВВОДУ-ВИВОДУ І8255**

**Принцип роботи ВІС І8255**

BІС паралельного інтерфейсу І8255,блок-схема якої представлена на рис.6.1, має три 8 розрядних портів А,В, С. Порти А.В.С мають програмне керування і можуть бути включені в різних функціональних конфігураціях, при цьому кожний з портів має притаманні йому функціональні характеристики. Порти А,В,С включають у собі:

Порт А – один 8-ми розрядний буферний регістр виводу даних і один 8-розрядний регіст вводу даних

Порт В - один 8-ми розрядний буферний регістр вводу\виводу даних і один 8-ми розрядний буфер вводу даних

Порт С - один 8-ми розрядний буферний регітр виводу даних і один 8-ми розрядний буфер вводу даних. Цей порт може бути подiлений на 2 4-ох розрядних порти. Кожний 4-ох розрядний порт має 4-ох розрядний регістр i може бути використований при передачi сигналiв керування і прийому сигналів стану при роботі сумісно з портами А і В.

ЦП задає режими роботи портів за допомогою двох типів управляючих слів:

1) команди встановлення режимів роботи (розряд D7=1);

2) команди встановлення розряду порта С (розряд D7=0).

Формати слів керування і встановлення дані відповідно на рис.6.2,3. Тип слова керування визначається станом розряду D7.

**Вибір режиму роботи**

ВІС паралельного інтерфейсу може працювати в одному з трьох програмно задаваємих режимів:

Режим 0 –основний ввiд/вивiд.

Режим 1 -стробований ввiд/вивiд.

Режим 2 -двунаправлена передача даних.

При встановленi на входi R рiвня "лог 1" адаптер встановлюється у початковий стан, всi порти налаштовуються на ввод в режимі 0 (тобто усi 24 вихiднi лiнii встановлюются в стан з високим опiром). Пiсля зняття сигналу встановлення у початковий стан порти адаптеру залишаються в цьому режимі, додатковоi команди режиму не потрiбно. Для виконання програми вибору iнших режимів роботи можливо здійснити це за допомогою однiєю команди. Це дозволяє адаптеру обслуговивати вилику кiлькiсть зовнишнiх пристроiв засобамi програмування. Режими роботи портiв А i В можуть задаватися незалежно одне від одного, в той час порт С дiлиться на двi частини, режим

**Блок схема адаптера**

**Керування групою А**

**Група В**

**Порт С**

**(молодші 4біт)**

**Група А**

**Порт С**

**(старші 4 біта)**

**Група А**

**Порт А**

**(8 біт)**

**Група В**

**Порт В**

**(8 біт)**

**Буфер**

**шини**

**даних**

**Схема**

**Керуван-ня**

**Читан-ням/**

**записом**

**Керування**

**групою В**

**РА0-РА7**

**РС4-РС7**

**РС0-РС3**

**RD/**

**WR/**

**A1**

**A0**

**REZET**

**РВ0…РВ7**

**CS**

D0…D7 -лінії даних

REZET -апаратний зкид

CS/ -вибір корпуса ВІС

RD -вхід керування введенням

WR/ -вихід керування висновком

А0, А1 -адреса порту

РАО…РА7 -порт А

РВО-РВ7 - порт В

РСО-РС7 - порт С

Рисунок 5.1 – Блок схема адаптера

роботи яких задаються в залежностi вiд призначення режимів портiв А i В. При змiнi режимів роботи усi вихiднi регiстри адаптера а також тригери стану , встановлюється в початковий стан, за виключенням тригера готовності (вивiд готов) у режимах 1 i 2. Вибираючi різні комбінації режимів роботи портів можливо забепечити необхідну функціональну організацію практично для будь-якої структури вводу\виводу.

**Керування портом С**

Кожний з 8 розрядів порту С може бути встановлений в «1» або «0» за допомогою однiєї команди вивiд Ця можливість спрощує програмування ВІС паралельного інтерфесу при використанні керуючих сигналів.

В випадку, коли порт С використовується для керування портом А або В, ці розряди можуть бути встановленi в «1» або «0» словом встановлення порту С. При цьому група порту С, в яку входить встановлюємий розряд , вкючається на вивід.

**Керування сигралами переривання ВІС паралельного інтерфейсу**

При роботі в режимі 1 або 2 ВІС паралельного інтерфейсу формує сигнали, які можуть бути використані в якості запросів переривань до ЦП. Сигнали запитів переривань, які формуються в порту С, забороняються або дозволяються встановленням в «1» або гашенням тригеру переривань за допомоною команди встановлення порту С. Це дозволяє керувати запитами переривань одних зовнішних пристроїв на інші пристрої в структурі переривання. При встановленні відповідного розряду порту С в «1» тригер перериань встановлюється в «1» і дозволяє периванння , а при встановленні цього розряду в «0» тригер переривань встановлюється в «0» і забороняє переривання.

При виборі режиму роботи портів і встановленні по входу R в початковий стан усі тригери дозвілу гасяться.

Формат команди встановлення/гашення розряду порта С

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D7**  **0** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |

D7 – ознака слова

D6,D5, D4 – не використовуються

D3, D2, D1 – вибір біту порту С

D0 – 0 встановлення в 0; 1-в 1

Рисунок 5.2 – Формат команди встановлення гашення розряда порта С

**Группа В**

**Порт С (молодша)**

**1=ввод**

**0=вивід**

**Порт В**

1. **ввод**

**0- вивід**

**Вібор режима B**

**0-режим 0**

**1-режим 1**

**Група А**

**Порт С (старший)**

**1-ввод**

**0-вивід**

**Порт А**

**1- ввод**

**0- вивід**

**Вібор режима A**

**00- режим 0**

**01- режим 1**

**10- режим 2**

**Ознака виставлення режима 1-активний команда режима.**

Д7

Д6

Д5

Д3

Д2

Д1

Д0

Д4

**Керуюче слово**

Рисунок 5.3 Керуюче слово режиму

**РЕЖИМИ РОБОТИ І8255**

Режим 0 (головний ввод\вивід)

Функціональна конфігурація режиму 0 І8255 забезпечує простий ввод і вивід даних через кожний з трьох портів. В цьому режимі немає сигналів керування асинхронною передачею даних, дані просто записуються в порт або зчитуються з нтого. Часова діаграма роботи ВІС паралельного інтерфейсу в режимі 0 представлена на рис. 6.4.

Режим 0 має слідуючі функціональні характеристики:

- Два 8-розрядних порта i два 4-х розрядних порта;

- любий порт може бути включений на ввод або вивiд;

- вивід даних з промуміжковим збереженям;

- ввод даних без збереження.

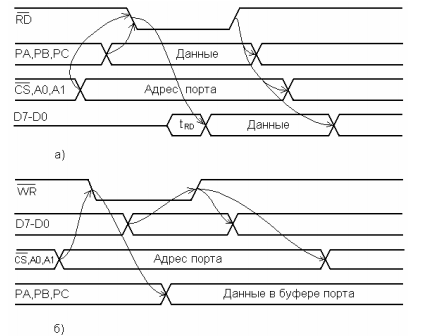


Рисунок 5.4 – Тимчасові діаграми роботи режима 0:

а) режим вводу; б) режим вивода

В цьому режимі можливі 16 конфігурацій вводу\виводу; 4 можливих (з 16) конфігурацій дані на рис.6.5



Рисунок 5.5-Комбінація схем вводу\виводу каналів І8255 в режимі 0

Режим 1(стробуємий ввод\вивід)

Функціональна конфігурація вводу\виводу в цьому режимі забезпечує передачу даних в заданий порт або з нього сумісно з асинхроними сигналами керування передачею. В режимі 1 порт А і порт В використовують порт С для формування або прийому сигналів керування.

Режим 1 має слідуючі функціональні характеристики:

- 2 порти передачi даних ( А i B);

- кожний порт передачi мiстить восьмирозрядний порт даних i чотирьохрозрядний порт керування\даних (порт С);

- восьмирозрядний порт даних може бути включений на ввод або вивід. Ввод і вивід даних здійснюється з проміжним збереженням.

Сигнали керування вводом в режимі 1:

- /STB (Strobe-вхід стробу).

При рівні лог. «0» сигналу дані записуються в вхідний регістр;

* IBF ( Input Buffer Full-ознака заповнення регістру вводу).

Високий рівень лог. «1» на цьому вході вказує на те, що дані записані у вхідний регістр; сигнал IBF встановлюється в «1» по зрізу сигнала /STB і гаситься по фронту сигналу RD ;

* INTR ( Interrupt-запит переривання).

Високий рівень на цьому вході може бути використаний для переривання ЦП при запиті обслуговування периферійним пристроєм. Сигнал INTR встановлюється в стан лог. «1» по фронту сигналу /STB при IBF=1 і при внутришньому сигналі дозволу переривання INTE=1. Гашення запиту переривання виконується зрізом сигналу RD. Такий засіб формування переривання дозволяє периферійному пристрою запитати обслуговування простим стропуванням даних, передаваємих в порт.

- INTRа.Внутришній сигнал дозволу переривання, який встановлює розряд PC3 порту С.

-INTRв.Внутришній сигнал дозволу переривання, який встановлює розряд PC0 порту С.

Сигнали керування вводу порту А і В

PC5-IBFa-(вихід)

PC4--(вхід)

PC3-INTRa-(вихід)

PC2-b-(вхід)

PC1-IBFb-(вихід)

PC0-INTRb-(вихід)

Конфігурація і часова діаграма вводу в режимі 1 ВІС показані на рис.6.6,7.

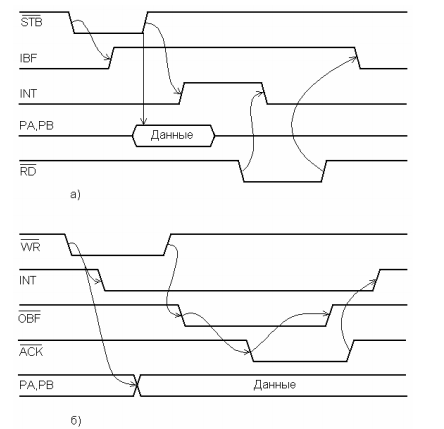


Рисунок 5.6 - Тимчасові діаграми роботи режима 1:

а) режим ввода; б) режим вивода



Рисунок 5.7- Режим 1(ввод)

Сигнали керування виводом в режимі 1:

- /OBF (Outpur Buffer Full- ознака заповнення регістру виводу). Переключення виводу до низького рiвня лог. «0» вказує на те,що ЦП закінчив запис даних в заданий порт. Сигнал /OBF встановлюється по фронту сигнала WR і гаситься по зрізу сигнала /ACK.

- /ACK ( ACKnowledge-підтвердження прийому).

Низький рівень лог. «0» сигналу повідомляє ВІС , що дані на порту А або В прийняті. Тобто це сигнал відповіді периферійного пристрою , який вказує про те що прийняті дані, які передав ЦП.

* INTR ( запит переривання).

Цей сигнал може бути використаний для перерви ЦП при прийомі даних периферійним пристроєм. Вихід INTR встановлюється в стан лог. «1» по фронту сигналу /ACK при /OBF =1 і INTE=1. Гашення сигналу INTR виконується по зрізу сигналу WR.

* INTRа.

Внутришній сигнал дозволу переривання, який встановлює розряд РС3 порту С.

* INTRв.

Внутришній сигнал дозволу переривання, який встановлює розряд РС0 порту С.

Сигнали керування виводу порту А і В

PC7--(вихід)

PC6- -(вхід)

PC3-INTRa-(вихід)

PC2- -(вхід)

PC1--(вихід)

PC0-INTRb-(вихід)

Конфігурація і часова діаграма виводу в режимі 1 ВІС показані на рис.5.6,5.9.



Рисунок 5.8- Режим 1(вивід)



Рисунок 5.9-Конфігурація режиму 1

**РЕЖИМ 2**

Функціональна конфігурація ВІС в режимі 2 забезпечує зв’язок з периферійними пристроями по одній 8 розрядній двонаправленій шині. Керування передачею даних здійснюється тими же асинхронними сигналами, що і в режимі 1.Здійснюється також формування сигналів переривання і дозволу або заборони переривання.

Режим 2 має слідуючі функціональні характеристики:

- в цьому режимі працює тільки порт А;

- один 8 розрядний порт двонаправленої передачі даних (порт А) і 8-розрядний порт керування (порт С);

- ввод і вивід даних з збереженням;

- 5-розрядний порт керування (порт С) використовується для прийому і передачі сигналів керування портом двонаправленої передачі (портом А).

Сигнали керування двонаправленою передачею в режимі 2:

-INTR ( запит переривання).

Високий рівень на цьому виході (лог. «1») може бути використаний для переривання ЦП при операції вводу або виводу.

Сигнали керування виводом в режимі 2:

* /OBF (ознака заповнення регістру виводу).

Вихід /OBF переключенням на низкий рівень лог. «0» вказує на те, що ЦП записав дані в порт А.

* /ACK (підтвердження).Низький рівень на цьому вході з тристабільним станом вихідного буферу порта дозволяє передачу даних. В іншому випадку , вихідний буфер буде в стані високого імпедансу.
* INTE (INTE Flip-Flop, зв’язаний з /OBF). Контролюэ встановлення або зкид біту PC6.

Сигнали керування вводом в режимі 2:

* / STB (Strobe Input)..Низький рівень на цьому вході дозволяє завантаження даних у вхідний буфер.

- IBF (Input Buffer Full F/F). Високий рівень на цьому виході вказує , що дані можуть бути зчитані з виходу зовнішнього пристрою.

* - INTE (INTE Flip-Flop, зв’язаний з IBF). Контролюэ встановлення або зкид біту PC4.

Сигнали IBF або OBF інформують зовнішній пристрій про готовність прийняти або передати дані. В відповідності зі станом IBF або OBF зовнішній пристрій або генерює чергові дані, супроводжуючи їх стробом STB, або формує сигнал підтверджения прийому АСК, готуючись до прийому даних. Низький рівень сигналу АСК відкриває вихідні буфери порту А, дозволяючи видачу даних на шину. В інших випадках шина порту А находиться в Z-стані.

* Синали керування вводу\виводу в режимі 2
* PC3 – INTRa -вивід
* PC4 - - вхід
* PC5 – IBFa - вихід
* PC6 - - вхід
* PC7 - - вихід



Рисунок 5.10- Часова діаграма роботи ВІС в режимі 2



Рисунок 5.11-Організація двонаправленого вводу/виводу

Порти А.В.С для роботи в вище згаданих режимах програмуються незалежно один від одного.

Cигнали керування порту А в режимі 2 при роботі з принтером

PC7--запит на передачу (від комп’ютера)

PC6--підтвердження прийому (від принтеру)

PC5-BUSY-сигнал зайнятості (рівнем 1)

PC4-PE-папір відсутній (від принтера рівнем 1)

**Приклади програмування**

**Прийом / передача даних в режимі 0**

Вивід даних в порт А,В;ввод з С в режимі 0

Port A 80h

Port B 81h

Port C 82h

RUS 83h

Org 2000h

Управляюче слово режиму 10001001-89h

Mvi a,89h ; RUS

Out 83h

Mvi a,xxh ; вивід даних

Out 80h` ; в А

Mvi a,xxh

Out 81h ; в В

In 82h ; з С

Mov m,a

Hlt

end

Контрольні питання та завдання

1. Архітектура ВІС паралельного інтерфейсу І8255. Призначення, структурна схема, призначення основних блоків, вхідних/вихідних сигналів.

2. Система мікрокоманд, формати команд, формат слова-стану. Режими роботи ВІС І8255 (паралельний інтерфейс). Підключення до шин МП.

3. Розробіть підпрограму вводу-виводу байта через порт В в режимі 0.

4. Розробіть схему підключення друкуючого пристрою і програму управління обміном, використовуючи режими 1 портів A і В.

5. Визначте гідності режиму 1 у порівнянні з режимом 0 на основі аналізу розроблених вами програм управління.

6. Складіть тимчасові діаграми обміну даними з квітуванням.

7. Зобразіть схему підключення І8255 до шини МПС.

8. Поясніть використання SW при організації введення-виведення даних.

# 6.1 Контролер прямого доступу до пам’яті

Контролер прямого доступу до пам'яті (ПДП, DMA - Direct Memory Access) забезпечує високошвидкісний обмін даними між пристроями вводу-виводу і ОЗУ без використання центрального процесора, що дозволяє звільнити процесор для виконання обчислень паралельно з обміном і незалежно від нього. Відчутні переваги дає використання ПДП у процесі обміну з пристроями, що приймають або передають дані досить великими порціями з високою швидкістю.

У ІВМ РС – подібних комп’ютерах функції КПДП виконує мікросхема 8237 фірмі INTEL. Контролер має 4 незалежних канали, кожний із яких може обслуговувати один периферійний пристрій.

# 6.1 Принципи роботи контролера ПДП

**Ввод-вивід з прямим доступом до пам’яті**

Обмін даними з повільно діючими периферійними пристроями, наприклад принтером створюється за допомогою переривань і якщо підпрограма обслуговування переривань триває 50 мксек а швидкість передачі 100 символів\с то на ввід-вивід витрачається 0,5% часу процесора і не виникає питань збільшення продуктивності процесора, але при передачі даних між основною і зовнішньою пам’яттю ЕОМ великих блоків даних (128-2048 байт) продуктивність процесора у режимі переривань недостатня.

Тому для передачі даних безпосередньо між пристроями зовнішньої пам’яті і оперативною пам’яттю , розроблений спеціальний метод передачі даних без участі ЦП , який отримав назву прямого доступу до пам’яті (ПДП). Апаратні засоби реалізації каналу ПДП звуться КПДП.

У ідеальному випадку ПДП не повинен впливати на дії процесора, але для цього потрібен складний тракт до основної пам’яті, тому у більшості ЕОМ використовується часове розділення (мультиплексування)загальної системної шини між процесором і КПДП. В звичайних умовах системною шиною розпоряджається процесор, який керує усіма передачами даних між оперативною і зовнішньою пам’яттю. Коли ініціалазується режим ПДП , шиною розпоряджається КПДП , а дії ЦП зупиняються і він відключається від системної шини.

Розроблено дві різноманітності ПДП. У режимі ідентифікації стану пам’яті передачі ПДП виконуються без інформування ЦП , для чого використовуються ті інтервали машиних циклів, коли ЦП не звертається до системної шини, а виконує внутрішні перетворення даних. Таким и інтервалами у мікропроцесора І8080 є такти Тu і Ts. ЦП ( або додаткова схема) ідентифікує ці інтервали для КПДП спеціальним сигналом, який дозволяє доступ до системної шини. Продуктивність ЦП у такому режимі не зменшується.

У режимі з пропуском тактів(вірніше «позикою» тактів) КПДП при необхідностя передачі даних сигналом запиту REQ або HOLD процесор відключається від системної шини на декілька тактів. Продуктивність ЦП у такому режимі зменшується.

# 6.2 Блок схема КПДП

Типовим прикладом сучасного КПДП є ВІС І8257/І8237. Мікросхема може керувати роботою 4 незалежних каналів ПДП , з урахуванням пріоритетів периферійних пристроїв. Для ЦП КПДП являє собою декілька паралельних 8 –бітних портів вводу-виводу. Після проми ініціалізації контролер керує передачею блоку даних до 16к без втручання ЦП. Допускається програмування довільного з трьох режимів роботи:

* Зчитування ( передача з основної пам’яті у зовнішнью)
* Запису(передача з зовнішньої пам’яті до основної) і перевірки ПДП.

Спрощена структурна схема КПДП приведена на рисунку 7..

До складу ВІС входять: двонаправлений двостабильний буфер даних(ВD), призначена для обміну інформацією між МП і КПДП; схема керування читанням / записом(RWCU), що адресує внутрішні регістри КПДП і керує обміном по шині D(7-0); блок керування (CU), що містить регістри режиму і стану КПДП і забеспечиваючий послідовність операцій , необхідну для організації режиму прямого доступу до пам'яті ; блок керування пріоритетами (PCU), що забезпечує визначений порядок обслуговування запитів зовнішніх пристроїв; чотири канали прямого доступу (CH0-CH3), кожний з який містить регістр адреси комірки пам'яті, з яким виробляється обмін , і лічильник циклів обміну , два старших розряди якого відведені для завдання операції обміну.

Призначення вхідних , вихідних і керуючих сигналів КПДП :

CLK - вхід для підключення тактового генератора FCLK = 3 МГц.

CS - вибір кристала. 0 CS = дозволяє роботу КПДП.

RESET - скидання. Сигнал високого рівня переводить КПДП в початковий стан, встановлюючи в нуль регістри команд, умов, тимчасового зберігання, а також встановлюючи в одиницю всі розряди маски.

READY - готовність. Вхідний сигнал, що використовується для синхронізації роботи КПДП з медленнодействующімі пристроями.

HLDA - підтвердження захоплення. Вхідний сигнал, що використовується ЦП для повідомлення КПДП про можливість виконання циклів ПДП.

DREQ3 - DREQ0 - входи запитів на ПДП від зовнішніх пристроїв. Полярність запитів задається програмно. Сигнали на цих входах повинні утримуватися до приходу сигналу DACK. У початковому стані пріоритет запитів природній, DREQ0 має найвищий пріоритет.

DB7 - DB0 - двунаправленная шина даних з буфером, що мають z-стан. У циклах ПДП на ці лінії видається вісім старших розрядів адресного коду, які необхідно «заклацнути» на зовнішньому регістрі сигналом ADSTB. У режимі роботи з ЦП по цих лініях здійснюється прийом / передача даних.

IOR - читання; як вхід використовується ЦП для читання вмісту внутрішніх регістрів КПДП; як вихід у режимі ПДП дозволяє видачу даних із зовнішніх пристроїв.

IOW - запис; як вхід використовується ЦП для завантаження даних в регістри КПДП; як вихід у режимі ПДП дозволяє запис даних в регістри зовнішніх пристроїв.

EOP - закінчення процесу. Вхід / вихід, який використовується для вказівки закінчення процесу передачі даних у режимі ПДП. Подаючи на цей вхід сигнал низького рівня, можна припинити передачу даних. Після завершення передачі даних по одному з каналів на виході встановлюється сигнал 0 EOP =. За цим сигналом (зовнішньому чи внутрішньому) знімається запит, і обслуговування припиняється. Якщо встановлено режим автоініціалізаціі, то відбувається завантаження робочих регістрів даного каналу вмістом базових регістрів, а розряди регістра маски не змінюються. У режимах без автоініціалізаціі розряди маски і розряд ТЗ у слові-стані встановлюються відповідно до стану обслуговуваного каналу. При передачі пам'ять - пам'ять висновок EOP орієнтований на вихід, і після закінчення рахунку на цьому виході формується сигнал. Якщо висновок EOP не використовується, то він повинен бути підключений через резистор до шини живлення (+5 В) для запобігання формування хибних сигналів закінчення процесу.

А3 - А0 - адресні входи / виходи. Використовуються як вхідні в режимі роботи з ЦП і для адресації до каналів і регістрам каналів КПДП. У режимі ПДП є виходами, по яких передаються чотири молодших розряду адреси ОЗП.

А7 - А4 - адресні виходи, на які в режимі ПДП передаються відповідні розряди адреси ОЗУ. У режимі роботи з ЦП переходять у z-стан.

HRQ - вихід запиту захоплення шин. Запит до ЦП для переходу в режим ПДП.

DACK3 - DACK0 - підтвердження ПДП. Вихідні лінії, на які видано-ются повідомлення для ВУ про можливість виконання циклів ПДП. Полярність сигналу задається програмно. Після сигналу RESET на виходах DACK встановлюється нуль.

AEN - дозвіл адреси. AEN = 1 встановлюється на час видачі восьми старших розрядів адреси ОЗУ на лінії DB7 - DB0.

ADSTB - строб адреси. Вихід, на якому формується імпульс (строб), який здійснює запис старших розрядів (А15 - А8) адреси ОЗП з шин DB7 - DB0 у зовнішній буферний регістр.

MEMR - читання з пам'яті. Вихід, який використовується в режимі ПДП для управління операцією читання з пам'яті.

MEMW - запис в пам'ять. Вихід, який використовується в режимі ПДП для управління операцією запису в пам'ять.

Ucc - шина живлення (+5 В).

GND - загальний.

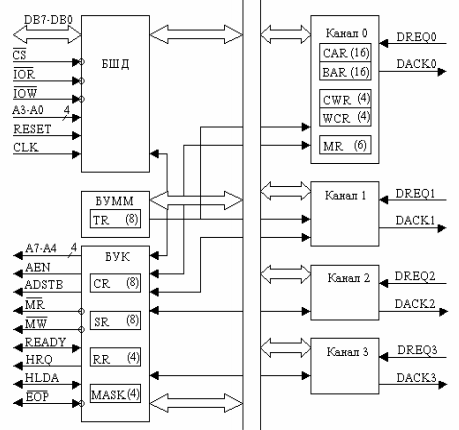


Рисунок 6.2 – Структурна схема КПДП

# 6.3 Опис внутрішніх регістрів КПДП

Контролер має 344 біта внутрішньої пам'яті, організованої у виді регістрів. Опис внутрішніх регістрів ПДП приведено в таблиці 7.1.

**Регістр початкової адреси (Base Address Register).**

У цьому регістрі задається стартова адреса ОЗУ, з якого починається передача. Регістр містить 16 розрядів і визначає адреса усередині заданої сторінки пам'яті розміром 64к. Завдання номера сторінки пам'яті здійснюється через спеціальні сторінкові регістри (Page Registers), підтримувані зовнішньою логікою. Кожен канал ПДП має свій регістр початкової адреси і сторінковий регістр. Такий розподіл пам'яті на сторінки не дозволяє здійснити обмін із блоком пам'яті, що знаходиться на перетинанні двох сторінок. Кожна сторінка починається із сегментної адреси, кратного 1000h (0, 1000h, 2000h, ..., 9000h).

**Регістр початкового лічильника циклів (Base Word Count Register).**

У цьому регістрі задається початкове число циклів передачі для програмувального каналу. Фактичне число переданих під час роботи ПДП елементів даних на одиницю перевищує задане число циклів, тобто якщо Ви задаєте 100 циклів передачі, а розмір елемента буде дорівнює 1 байтові, то за сеанс обміну буде переданий 101 байт інформації.

**Регістр поточної адреси (Current Address Register).**

Початкове значення заноситься в цей регістр одночасно з регістром початкової адреси. Надалі в ході передачі значення поточної адреси автоматично збільшується або зменшується (конкретний напрямок зміни задається при програмуванні в регістрі режиму). Якщо дозволено автоинициализация, то після закінчення передачі в регістр автоматично встановлюється значення з регістра початкової адреси.

Таблиця 6.1— Регістри контролера ПДП

|  |  |  |
| --- | --- | --- |
| Найменування регістра | Розрядність (біт) | Число регістрів |
| Регістр початкової адреси (Base Address Register) | 16 | 4 |
| Регістр початкового лічильника циклів (Base Word Count Register) | 16 | 4 |
| Регістр поточної адреси (Current Address Register) | 16 | 4 |
| Регістр поточного лічильника циклів (Current Word Count Register) | 16 | 4 |
| Робочий регістр адреси (Temporary Address Register) | 16 | 1 |
| Робочий регістр лічильника циклів (Temporary Word Count Register) | 16 | 1 |
| Регістр стану (Status Register) | 8 | 1 |
| Регістр команд (Command Register) | 8 | 1 |
| Регістр режиму (Mode Register) | 6 | 4 |
| Робочий регістр (Temporary Register) | 8 | 1 |
| Регістр масок (Mask Register) | 4 | 1 |
| Регістр запитів (Request Register) | 4 | 1 |

**Регістр поточного лічильника циклів (Current Word Count Register).**

Регістр містить поточне значення лічильника циклів (число циклів передачі, що залишилися,). Відображуване в ньому число циклів завжди на одиницю менше числа ще не переданих елементів даних, тому що зміна значення в цьому регістрі виробляється наприкінці циклу передачі, уже після фактичної передачі елемента даних, а кінець передачі фіксується в момент переповнення лічильника (зміна його значення з 0 на 0FFFFh).

**Регістр режиму (Mode Register).**

Даний регістр задає режими роботи свого каналу контролера.

Кожний з чотирьох каналів ПДП має свій набір регістрів, описаних вище. Крім того, мається наступний набір регістрів, загальних для всіх каналів.

Формат команди

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |

D1,D0-режим роботи

0-0-перевірка

0-1-запис до пам’яті

1-0-читання з пам’яті

1-1-неможлива комбінація

D2- Автоініціалізація: 1-дозволена, 0-заборонена

D3-Зміна поточної адреси при обміні: 0-збільшення, 1-зменьшення

D4, D5 – Тип передачі:

0-0-режим передачі по вимозі

0-1-режим звичайної передачі

1-0-режим блочної передачі

1-1-каскадний режим

**Регістр команд (Command Register).**

Цей 8-бітний регістр керує роботою контролера. Він програмується, коли контролер знаходиться в стані програмування й очищається командами скидання «Reset» і «Master Clear». Призначення битов регістра команд приведено на малюнку.

Формат команди

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |

D0:0-заборонити передачу пам’ять-пам’ять; 1 – дозволити

D1:0-заборонити фіксацію адреси в каналі 0; 1 – дозволити

D2:0-заблокувати; 1 - -разблокувати контролер

D3:0-нормальна часова діаграма; 1 – стиск в часі

D4:0-режим фіксованих приорітетів; 1 – режим циклічного зсуву

D5:0-затримка при запису,якщо біт 3 встановлений ігнорується режим розширеного запису

D6:0-активним є високий рівень сигналу запиту DREQ, 1 - низький

D7:0-активним є високий рівень сигналу підтвердження запиту на ПДП (DACK), 1 - низький

**Регістр стану (Status Register).**

Регістр відбиває поточний стан запитів і передач по всім чотирьох каналах. Біти 0 - 3 встановлюються в одиницю після завершення передачі по каналах 0 - 3 (біт 0 - канал 0, біт 1 - канал 1 і т.д.), якщо не заданий режим автоинициализации. Ці біти очищаються після команди скидання контролера і після кожної операції зчитування стану з регістра стану. Біти 4 – 7 указують по якому з каналів 0 - 3 активний у сучасний момент сигнал запиту на ПДП.

**Регістр масок (Mask Register).**

Кожен біт цього 4-бітового регістра маскує/демаскує свій канал ПДП, при цьому значення 1 маскує канал, значення 0 демаскує канал і дозволяє прийом сигналу запиту по цьому каналі.

**Регістр запитів (Request Register).**

Сигнал запиту на ПДП (DREQ) може бути виданий пристроєм, що як обслуговується, так і програмно. Для програмного видання сигналу запиту по одному з 4-х каналів ПДП необхідно установити відповідний біт у 4-розрядному регістрі запитів. Запит на ПДП може бути відмінний записом нульового значення у відповідний біт регістра. Біт запиту очищається автоматично при закінченні передачі по даному каналі. Усі біти запитів очищаються при скиданні контролера. Для того, щоб сприймати програмні запити на ПДП, канал повинний знаходиться у режимі блокової передачі.

**Робочий регістр (Temporary Register).**

Цей 8-розрядний регістр використовується для збереження елемента даних, переданого в режимі фіксованої адреси при передачі пам'ять-пам'ять або для тимчасового збереження переданого байта при всіх інших режимах передачі.

# 6.4 Програмне керування контролером ПДП

Програмне керування контролером ПДП здійснюється через порти виводу-введення. Доступ до кожного регістра контролера може бути здійснений через свої порти вводу-виводу.

Порти 0h - 7h призначені для запису вихідних значень у регістри початкову і поточну адреси, початкового і поточного лічильника циклів для всіх 4-х каналів. Тому що порти восьмирозрядні, а регістри, у які через них заносяться дані, 16-розрядні, то запис виробляється в два прийоми. Перед першою командою виводу в необхідний порт необхідно скинути -тригер-засувку, для чого виконується команда виводу довільного значення в порт 0Ch, після чого в необхідний порт виводиться молодший байт 16-розрядного значення і потім старший байт наступної команди виводу в той же порт.

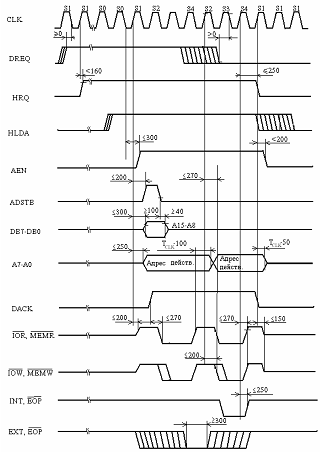


Рисунок 6.3 – Часова діаграма роботи КПДП

Вивід у порт 8h дозволяє занести значення в регістр команд ПДП. Читання з порту 8h зчитує регістр стану ПДП. Запис у порт 9h дозволяє установити або скинути біт запиту в регістр запитів для одного з каналів. Формат команди наступний:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |

D1,D0 – вибір каналу

0-0-канал 0

0-1-канал 1

1-0-канал 2

1-1-канал 3

D2-встановлення біту запиту на КПДП

1-встановити

0-зкинути

Рисунок 6.4 – Формат команди порту 9h

Запис у порт 0Ah дозволяє установити або скинути біт маски в регістрі масок для одного з каналів. Формат команди:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D7** | **D6** | **D5** | **D4** | **D3** | **D2** | **D1** | **D0** |

D1,D0 – вибір каналу

0-0-канал 0

0-1-канал 1

1-0-канал 2

1-1-канал 3

D2-встановлення біт маски

1-встановити

0-зкинути

Рисунок 6.5 – Формат команди порту 0Аh

Запис у порт 0Bh установлює значення в регістрі режимів одного з 4-х каналів ПДП. Біти 0 і 1 задають номер каналу (00 - 0, 01 - 1, 10 - 2, 11 - 3). У біти 2 - 7 заносяться значення, передані відповідно в біти 0 - 6 регістра режимів. Опис бітів регістра режимів приведене в п. 2.3.

Запис у порт 0Dh задає програмне скидання контролера (Master Clear). Висновок будь-якого байта в цей порт має той же ефект, що й апаратне скидання контролера. При програмному скиданні очищаються регістри команд, стану, запитів і робітник регістр. Так само скидається заскочка-клямка-тригер-засувка і установлюється всі біти масок у регістрі масок. Після програмного скидання контролер переходить у цикл чекання.

Вивід будь-якого байта в порт 0Eh очищає регістр масок - скидає біти масок усіх 4-х каналів ПДП і в такий спосіб дозволяє прийом запитів на ПДП по всіх каналах.

Через порт 0Fh можна задати довільне значення регістра масок ПДП. Для цього необхідно в бітах 0 - 3 регістри AL установити необхідне значення масок каналів 0 - 3 відповідно і вивести це значення в порт.

Сторінкові регістри ПДП призначені для завдання параметра сторінки пам'яті, з яким буде виробляється обмін. Під номером сторінки розуміються старші 4 біти повної 20-бітової адреси ОЗУ, тобто початкові сегментні адреси сторінок будуть кратні 1000h (0, 1000h, 2000h, ..., 9000h). Нагадаємо, що архітектура ПДП дозволяє працювати тільки зі сторінкою пам'яті розміром 64ДО, а логіка, що забезпечує переключення сторінок улаштована так, що сторінки мають тверді границі, описані вище. Через цю особливість неможливо за допомогою ПДП здійснити обмін із блоками пам'яті, що перетинають границю між двома такими сторінками.

# 6.5 Програмна модель КПДП. Програмування КПДП

Для початкової установки КПДП необхідно записати відповідну інформацію в 16-розрядний регістр адреси каналу (RGA), у 16-розрядний лічильник циклів каналу (З Т) і в 8-розрядний регістр режиму, загальний для всіх каналів. Запис цієї інформації виробляється за допомогою команди OUT, хоча можливий і інший спосіб звертання до КПДП як до комірок пам'яті. Запис інформації в 16-розрядні регістри здійснюється двома командами, починаючи з молодшого байта. Два старших розряди лічильника циклів визначають операцію обміну в такий спосіб: запис у пам'ять — 01, читання з пам'яті — 10, контроль — 00 (комбінація 11 заборонена). Стан КПДП можна контролювати читанням умісту RGA, СТ і 8-розрядного регістра стану, загального для всіх каналів, за допомогою команди IN. Для читання вмісту 16-розрядного регістра використовуються дві команди IN з однієї і тією же адресною частиною, причому спочатку відбувається зчитування молодшого байта висновків А (3 — 0) і значення сигналу CS для адресації внутрішніх регістрів КПДП. При програмуванні КПДП операції запису керуючих чи слів читання станів внутрішніх регістрів визначаються також значеннями сигналів I/OR і I/O W.

При програмуванні КПДП операції запису керуючих чи слів читання станів внутрішніх регістрів визначаються також значеннями сигналів I/OR і I/O W.

В таблиці 6.3 показані коди на А3-А0, відповідаючі камандам ЦП, а в табл.6.4-коди на А3-А0, які відповідають адресам регістрів КПДП.

Таблиця 6.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **А3** | **А2** | **А1** | **А0** | **Команда** | **Операція** |
| **1**  **1**  **1**  **1**  **1**  **1**  **1**  **1**  **1**  **1** | **0**  **0**  **0**  **0**  **0**  **1**  **1**  **1**  **1**  **1** | **0**  **0**  **0**  **1**  **1**  **0**  **0**  **0**  **1**  **1** | **0**  **0**  **1**  **0**  **1**  **0**  **1**  **1**  **0**  **1** | **Ввід**  **Вивід**  **Те саме**  **>>**  **>>**  **>>**  **Ввід**  **Вивід**  **Те саме**  **>>** | **Читання регістру стану**  **Запис до регістру команд керування**  **Запис до регістру запитів**  **Встановлення всіх розрядів маски**  **Запис до регістру режима**  **Встановлення режима ввода мол.байта**  **Читання регістру тимчасового зберігання**  **Загальний зкид**  **Зкид всіх розрядів маски**  **Встановлення розрада маски** |

Таблиця 6.4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **А3** | **А2** | **А1** | **А0** | **Команда** | **Операція** |
| **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0** | **0**  **0**  **0**  **0**  **0**  **0**  **0**  **0**  **1**  **1**  **1**  **1**  **1**  **1**  **1**  **1** | **0**  **0**  **0**  **0**  **1**  **1**  **1**  **1**  **0**  **0**  **0**  **0**  **1**  **1**  **1**  **1** | **0**  **0**  **1**  **1**  **0**  **0**  **1**  **1**  **0**  **0**  **1**  **1**  **0**  **0**  **1**  **1** | **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод**  **Вивід**  **Ввод** | **Заван. мл/ст байту в рег.BAR і CAR в к.0**  **Читання змісту CAR канала 0**  **Заван. мл/ст байту в рег.WCR і CWR в к.0**  **Читання змісту CWR канала 0**  **Заван. мл/ст байту в рег.BAR і CAR в к.1**  **Читання змісту CAR канала 1**  **Заван. мл/ст байту в рег.WCR і CWR в к.1**  **Читання змісту CWR канала 1**  **Заван. мл/ст байту в рег.BAR і CAR в к.2**  **Читання змісту CAR канала 2**  **Заван. мл/ст байту в рег.WCR і CWR в к.2**  **Читання змісту CWR канала 2**  **Заван. мл/ст байту в рег.BAR і CAR в к.3**  **Читання змісту CAR канала 3**  **Заван. мл/ст байту в рег.WCR і CWR в к.3**  **Читання змісту CWR канала 3** |

Формат керуючого слова, записується в регістр режиму( Mode Register ) -0Bh Даний регістр задає режими роботи каналу контролера.

Розряди команди записаної в регістр команд (Command Register)-08h D3-DO(EN3 - EN(J) задають дозвіл обміну по відповідному каналі, запис нуля в розряд забороняє обмін. Інші розряди визначають режими роботи каналу.

Розряд D4(RP) установлює порядок обслуговування запитів від каналів. При RP = 0 задається фіксований пріоритет каналів і канал 0 має вищий пріоритет. У режимі циклічного пріоритету (RP= 1) після обслуговування каналу йому привласнюється нижчий пріоритет, а наступному за ним по номері каналу — вищий. Причому циклічне зрушення пріоритетів виробляється після кожного циклу прямого доступу.

Режим розширеного запису (EW=і) збільшує за рахунок зсуву переднього фронту тривалість сигналів 1/OW і MEMW, генеруємих КПДП. Це дозволяє ВУ, що формує сигнал READY по фронті сигналу запису, зменшити час охолодження і збільшити швидкість обміну.

При TCS= 1 поява сигналу ТС в одному з каналів скидає відповідний розряд D3 — DO, у результаті чого канал відключається. Подальша робота цього каналу можлива після перезавантаження регістра режиму. Якщо TCS=0, то поява сигналу ТС не впливає на розряд дозволу роботи каналу і закінчувати передачу повинне ВУ за рахунок припинення вироблення сигналу DRQ.

У режимі автозавантаження (AL= 1) може працювати тільки другий канал, використовуючи уміст своїх внутрішніх регістрів і внутрішніх регістрів третього каналу. Після передачі даних відповідно до параметрів регістрів другого каналу і появи сигналу ТС параметри з регістрів третього каналу автоматично завантажуються в регістри другого каналу; прапор відновлення (UF) у регістрі стану каналів встановлюється в 1. Потім передача даних продовжується відповідно до нових параметрів регістрів другого каналу, а наприкінці пер-шого ж циклу прямого доступу з новими параметрами прапор UF скидається. Режим автозавантаження дозволяє організувати повторювані пересилання блоків даних з однаковими чи параметрами з'єднувати кілька блоків з різними параметрами.

У початковому стані SI запрограмований на визначений режим КПДП очікує запиту DRQ від ВУ. Переходячи в стан SO, він виробляє сигнал HRQ і очікує надходження від МП сигналу HLDA. Після надходження сигналу підтвердження HLDA починається цикл обміну. У стані S1 формується сигнал AEN для блоку-вання інших пристроїв системи від шин даних і керування, видається код молодших розрядів на виходи А (7 — 0), а код старших розрядів — на виходи D(7 — 0).

Видача старших розрядів адреси супроводжується стробуючим сигналом ADSTB для запису їх у зовнішній буферний регістр. У стані S2 виробляється сигнал DACK, що вказує на початок обміну, а також формуються пари сигналів MEMR, I/OR і MEMW, I/OW, що визначають напрямок обміну. У стані S3 відбувається передача даних у ЗУ чи ВУ.

Стан S4 завершує цикл прямого доступу. У цьому стані при передачі останнього байта видається сигнал ТС, а у випадку кінця

блоку — сигнал MARK. При необхідності узгодження швидкодії ЗП і ВП за допомогою сигналу READ Y між станами S3 і S4 уводиться необхідне число станів чекання SW. У режимі контролю перехід у стан SW не дозволяється.

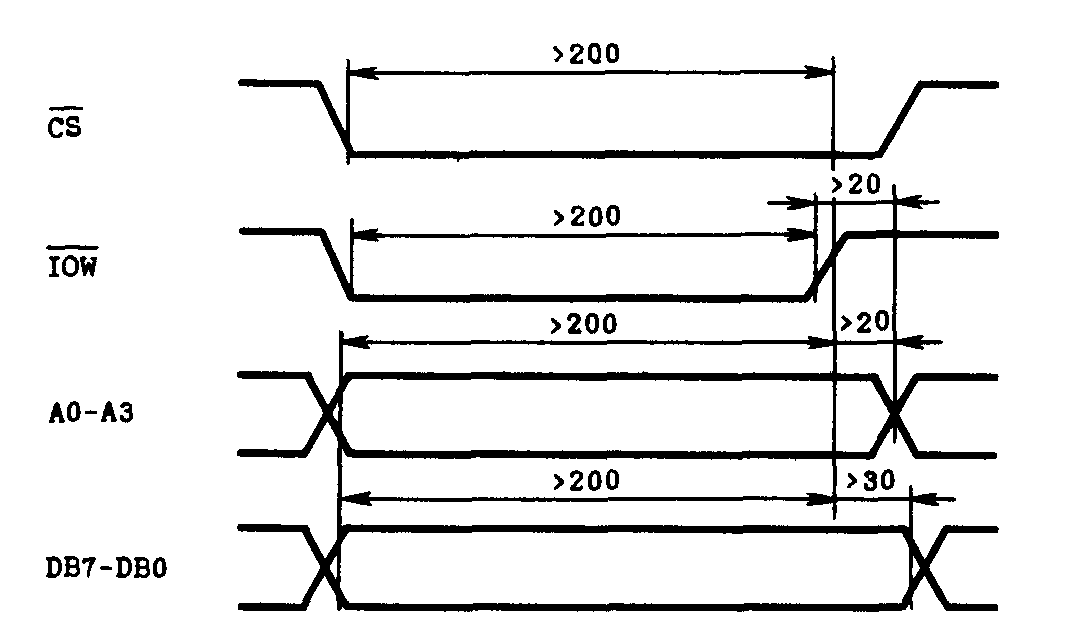


Рисунок 6.6 – Часова діаграма КПДП в циклі запису

В початковому стані *SI* запрограмований на визначений режим КПДП очікує запиту *DRQ* від ВП. Переходячи в стан SO, він виробляє сигнал *HRQ* і очікує надходження від МП сигналу *HLDA.* Після надходження сигналу підтвердження *HLDA* починається цикл обміну. У стані *S1* формується сигнал *AEN* для блокування інших пристроїв системи від шин даних і керування, видається код молодших розрядів на виходи *А (7 — 0),* а код старших розрядів — на виходи *D(7 — 0).*

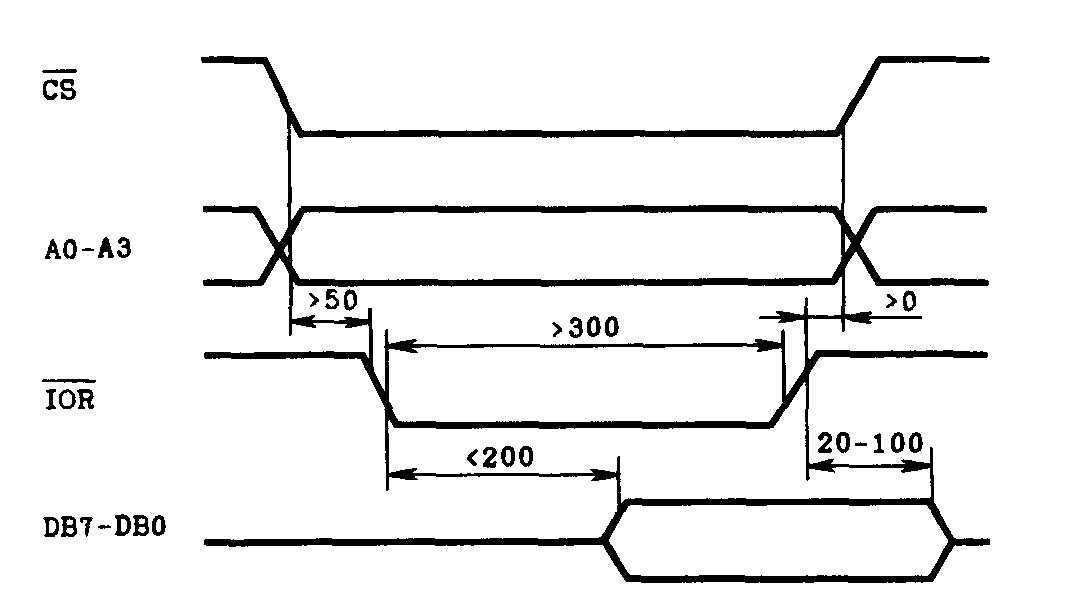


Рисунок 6.7 – Часова діаграма роботи КПДП в циклі читання

Видача старших розрядів адреси супроводжується стробуючим сигналом *ADSTB* для запису їх у зовнішній буферний регістр. У стані *S2* виробляється сигнал *DACK, що* вказує на початок обміну, а також формуються пари сигналів *MEMR, I/OR* і *MEMW, I/OW, що* визначають напрямок обміну. У стані S3 відбувається передача даних у ЗП чи ВП. Стан *S4* завершує цикл прямого доступу. У цьому стані при передачі останнього байта видається сигнал *ТС, а* у випадку кінця блоку — сигнал *MARK.* При необхідності узгодження швидкодії ЗП і ВП за допомогою сигналу *READ Y* між станами S3 і S4 уводиться необхідне число станів чекання *SW.* У режимі контролю перехід у стан *SW* не дозволяється.

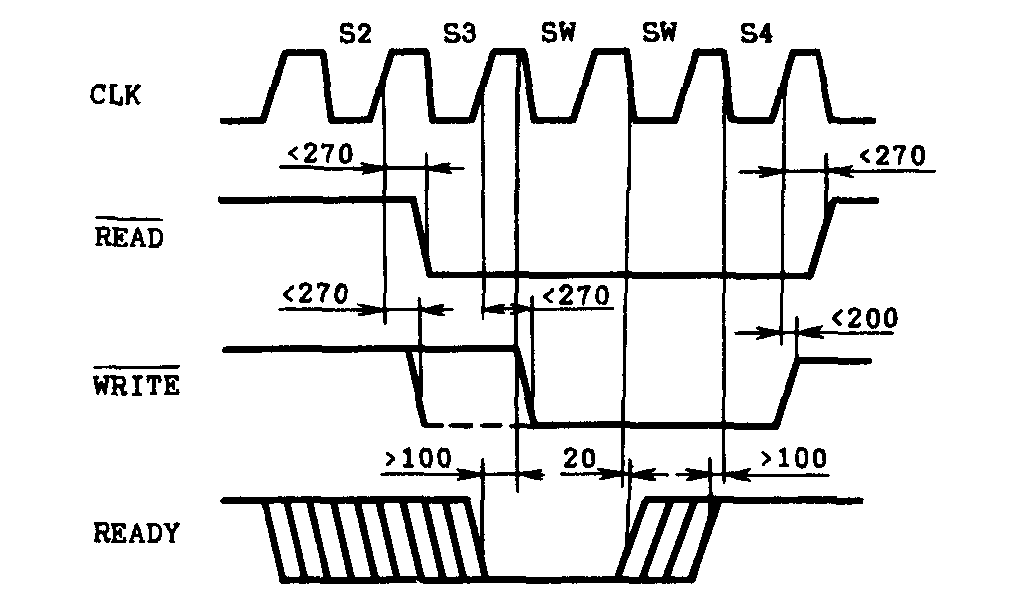


Рисунок 6.8– Часова діаграма роботи КПДП в циклах обміну

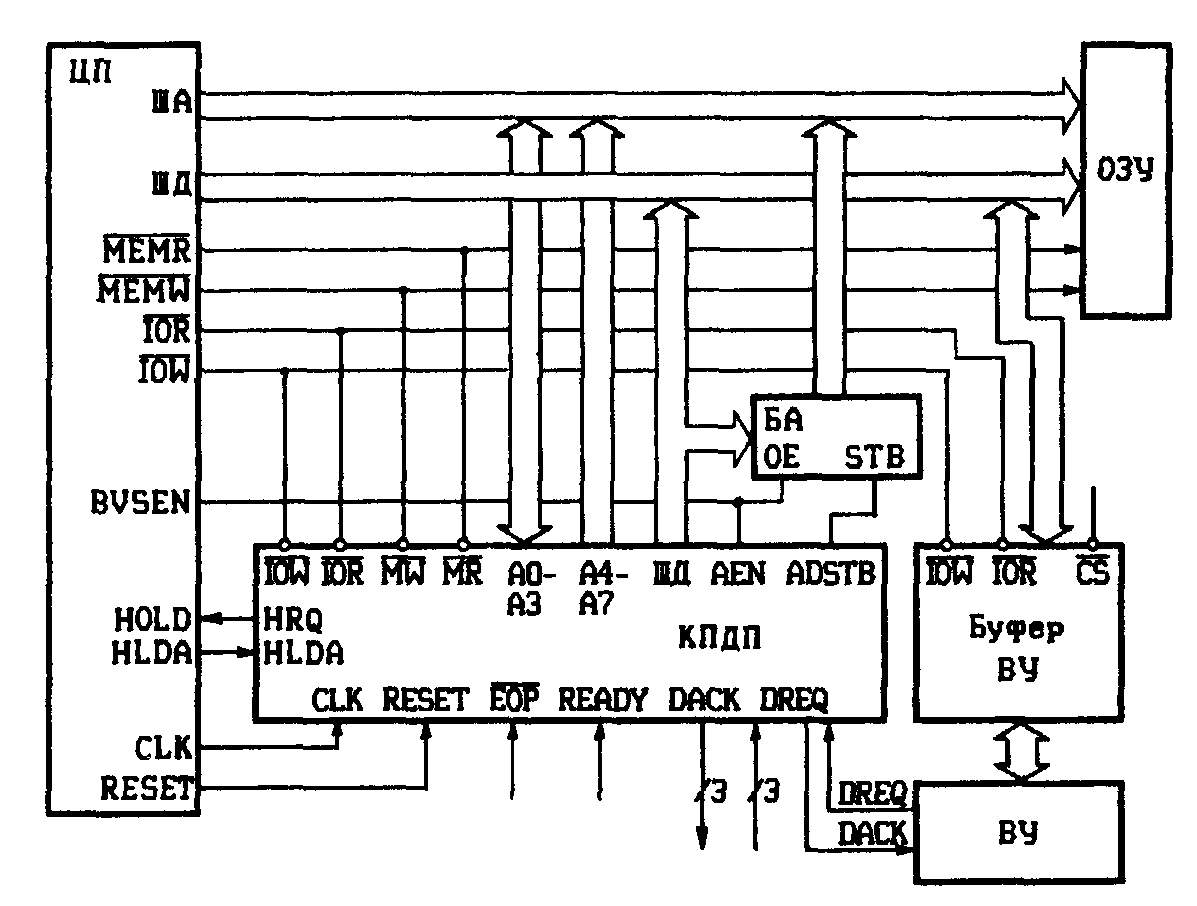


Рисунок 6.9 – Підключення КПДП до системної шини

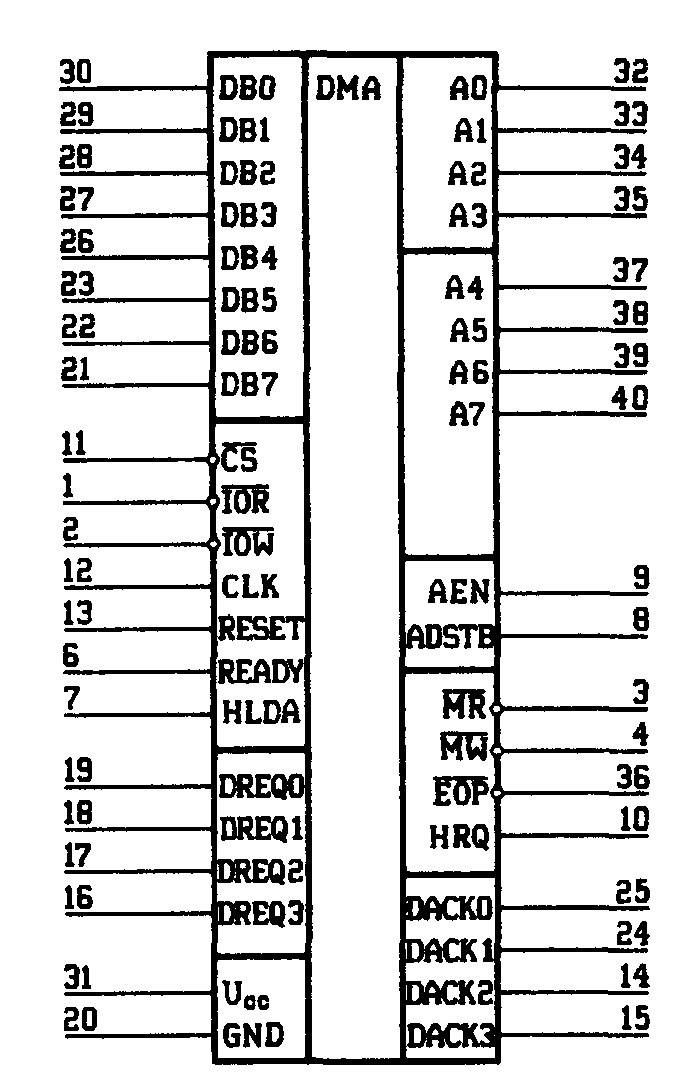


Рисунок 6.10- Графічне зображення корпусу КПДП

Фрагмент програми початкової установки КПДП при звертанні до нього як до зовнішнього пристрою може мати наступний вид:

**Приклад програмування КПДП**

DI ;заборона переривань

MVI A,28H ; завантаження керуючого слова у регістр режиму

; звичайна передача ,поточна адреса-зменьшення,

; авто ініціалізація-заборонена, режим-перевірка

OUT 0BH ; до регістру режиму

MVI A,1АH ; завантаження керуючого слова

; DACK- вис.рівень, DREQ-вис.рівень,затримка ;при запису, циклічний зсув приоритетів, ;зжимання часу передачі, блокування контролеру, ;дозвіл фіксації адреси, заборона передачі ;пам’ять-пам’ять

OUT 08H ; у регістр команд

MVI A,00H ; завантаження молодшого байта адреси

OUT 00H(0кан.) ; масиву в порт

02Н(1кан.)

04Н(2кан.)

06Н(3кан.)

MVI A,40H ;завантаження старшого байта адреси

OUT 00H(0кан.) ; масиву в порт

MVI A,01H ; завантаження номера сторінки пам'яті

OUT 87H(0кан.) ; у порт каналу

83H(1кан.)

81H(2кан.) ;

82H(3кан.)

MVI A,F4H ; завантаження молодшого байта лічильника циклів

OUT 01H(0кан.) ; у порт каналу

03Н(1кан.)

05Н(2кан.)

07Н(3кан.)

MVI A,00H ; завантаження старшого байта лічильника циклів

OUT 01H ; у порт каналу

03Н(1кан.)

05Н(2кан.)

07Н(3кан.)

EI ;дозвіл переривань

‘

У програмі виробляється настроювання каналу на запис масиву в область ОЗУ. Для передачі N байт у лічильник каналу завантажується число N- 1. Після програмного настроювання каналу СНО аналогічно набудовуються інші канали. У регістр режиму завантажується керуюче слово. Для попередження помилок перед програмуванням КПДП чи читанням умісту його регістрів необхідно забороняти переривання.

Підключення контролера до системної шини (рис. 7.9). Вісім старших розрядів адреси видаються на ШД і повинні бути записані сигналом ADSTB в зовнішній регістр. Лінія AEN використовується для того, щоб розряди адреси залишилися чинними на ША протягом трьох тактових періодів циклу ПДП. Лінії А7 - А0 підключаються безпосередньо до ША. Сигнали MEMR, MEMW, IOR, IOW керують у циклах ПДП відповідно ОЗП і буфером ВУ.

Контрольні питання та завдання

1. У яких режимах працюють КПДП і його функції в системі?

2. Визначте програмно-доступні регістри і їх адресацію.

3. Складіть схему підключення І8257 до шин адреси і даних МПС.

4. Складіть програму ініціалізації контролера для блочного обміну по одному каналу.

5. Поясніть реалізацію режиму ПДП на І8257.

6. Поясніть адресацію до регістрів КПДП при програмуванні.

7. Як формується 16-бітна адреса КПДП при управлінні обміном?

8. Які пріоритети запитів підтримує І8257 ?

9. В якій послідовності необхідно робити завантаження регистрів контролера при його програмуванні?

10.На лабораторній використовуючи емулятор КПДП написать програму передачі або прийому даних до пам’яті відповідно варіанту і відлогодити її.

# СПИСОК ЛІТЕРАТУРИ

1. 1970s Motorola History Highlights (<http://www.motorola.com/content/0,1037,121-286,00.html>)
2. 1980s Motorola History Highlights (<http://www.motorola.com/content/0,1037,122-287,00.html>)
3. 4 лучше чем 3, а 5? // КомпьютерПресс, 1992, декабрь.
4. Англо-русский словарь по вычислительной технике и программированию. © ABBYY, Е. К. Масловский. 1999.
5. Букчин Л. В., Безрукий Ю. Л. Дисковая подсистема IBM-совместимых персональных компьютеров. – М.: Бином, 1993.
6. Горбунов В. Л., Панфилов Д. И., Преснухин Д. Л. Справочное пособие по микропроцессорам и микроЭВМ. – М.: Высшая школа, 1988.
7. Григорьев В. Л. Архитектура и программирование арифметического сопроцессора. – М.: Энергоатомиздат, 1991.
8. Гук М. Аппаратные средства IBM PC. – СПб: Питер, 1996.
9. Гукин Д. IBM-совместимый персональный компьютер. – М.: Мир, 1993.
10. Джордейн Р. Справочник программиста персональных компьютеров типа IBM PC, XT и AT. – М.: Финансы и статистика, 1992.
11. Дьяконов В. П. Справочник по алгоритмам и программам на языке бейсик для персональных ЭВМ. – М.: Наука, 1987.
12. Знакомьтесь: компьютер. Под ред. В. М. Курочкина. – М.: Мир, 1989.
13. Знакомьтесь: Персональная ЭВМ Корвет. Ахманов С. А., Персианцев И. Г., Рахимов А. Т. и др. – М.: Наука, 1989.
14. Коффрон Дж. Технические средства микропроцессорных систем. – М.: Мир, 1983.
15. Кэмпбелл Дж. Л. Операционная система OS/2. – М.: Финансы и статистика, 1991.
16. Лю Ю-Чжен, Гибсон Г. Микропроцессоры семейства 8086/8088. – М.: Радио и связь, 1987.
17. Микропроцессорный комплект К1810. Казаринов Ю. М., Номоконов В. Н., Подклетнов Г. С., Филиппов Ф. В. – М.: Высшая школа, 1990.
18. Микропроцессоры: справочное пособие для разработчиков судовой РЭА. Гришин Г. Г., Мошков А. А., Ольшанский О. В., Овечкин Ю. А. – Л.: Судостроение, 1988.
19. Нортон П. Программно-аппаратная организация IBM PC. – М.: Радио и связь, 1992.
20. Нортон П., Гудман Дж. Персональный компьютер: аппаратно-программная организация. – СПб: BHV, 1999.
21. Персональный компьютер «Ириша». Барышников В. Н., Воронов М. А., Кулаков В. Б. и др. – М.: Патриот, 1990.
22. Персональные компьютеры. Информатика для всех. – М.: Наука, 1987.
23. Ратч Э. IBM AT. Руководство для начинающих. – М.: Радио и связь, 1993.
24. Рош У. Л. Последнее слово ещё не сказано: Процессор 486 в семействе PS/2. // PC Magazine USSR, 1991, июнь.
25. Рош У. Л. Библия по модернизации персонального компьютера. – Минск: ИПП «Тивали-Стиль», 1995.
26. Скэнлон Л. Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера. – М.: Радио и связь, 1989.
27. Смит Б. Э., Джонсон М. Т. Архитектура и программирование микропроцессора Intel 80386. – М.: Конкорд, 1992.
28. Токхайм Р. Микропроцессоры: курс и упражнения. М.: Энергоатомиздат, 1988.
29. Тули М. Справочное пособие по цифровой электронике. – М.: Энергоатомиздат, 1990.
30. Фафенбергер Б., Уолл Д. Толковый словарь по компьютерным технологиям и Internet. – Киев: Диалектика, 1996.
31. Холленд Р. Микропроцессоры и операционные системы. – М.: Энергоатомиздат, 1991.
32. Элфринг Г. Программирование на языке ассемблера для микроЭВМ. – М.: Радио и связь, 1987.
33. Intel думает о том, а не назвать ли новый процессор как-нибудь по-другому. // КомпьютерПресс, 1992, сентябрь.
34. Intel Microprocessor Hall of Fame (<http://intel.com/intel/intelis/museum/exhibit/hist_micro/hof/hof_main.htm>)
35. MASM32 help library: Intel Hex Opcodes And Mnemonics. © S. L. Hutchesson 1999 – 2000
36. Processor Madness (<http://skyscraper.fortunecity.com/dos/661/main.htm>)

37. Сидоренко В.В.Курс лекцій “Мікропроцесорні ВІС”