Кафедра электрофизических установок МИФИ

Микропроцессорные системы

Конспект лекций

Коршунов А.М.
16.02.2022

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ

Появление микропроцессоров, по их значимости, отмечено вехой на пути научно-технического прогресса, начавшегося, как известно, с промышленной революции XVIII - XIX веков, ознаменованной изобретением паровой машины, заменившей низко-производительный ручной труд на механический. Но окончательный процесс механизации производства был завершен только после создания электродвигателей, позволивших децентрализовать механические участки производства.

В дальнейшем, интенсификация производства была не возможна без его комплексной автоматизации как средства дальнейшего повышения производительности труда. Эта задача могла быть решена только с применением вычислительной техники. Проводя аналогию с паровой машиной, в процессе автоматизации ее роль играет ЭВМ, а роль электродвигателей, следовательно, призваны выполнить микропроцессоры.

Именно эти недорогие устройства, обладая превосходными характеристиками: малыми габаритами и массой, низким электропотреблением, высокой надежностью, устойчивостью характеристик в большом диапазоне температур, способностью выдерживать значительные динамические нагрузки обеспечивают инженерную разработку децентрализованных систем управления и обработки информации.

Применение микропроцессоров создает необходимые условия интенсификации производства, позволяет улучшить условия труда, открывает перспективы развития науки, образования, культуры, совершенствования систем связи и транспорта, сферы обслуживания, организации досуга и т. д. Трудно переоценить значение микропроцессоров при создании измерительно-вычислительных и управляющих комплексов, автоматизированных средств измерений и управления, предназначенных для исследований, контроля и испытаний сложных объектов.

1.1. Микропроцессоры, история их создания

Термин “микропроцессор” (МП) был впервые употреблен в 1972 г., хотя годом рождения этого прибора следует считать 1971 год, когда фирма Intel запустила в производство МП серии 4000, который тогда назывался “интегральное микропрограммируемое вычислительное устройство” и представлял собой однокристальный центральный процессор 4004 для калькуляторов. В своем составе он имел 4-разрядный параллельный сумматор, шестнадцать 4-разрядных регистров, аккумулятор и стек. МП был реа-

лизован на 2300 транзисторах и выполнял 45 различных команд. Быстродействие первого микропроцессора составляла 8000 оп/с.

История создания микропроцессора такова. Японская фирма Busicom обратилась в Intel с заказом разработать набор из 12 интегральных микросхем (ИМС) для нового семейства электронных калькуляторов. Такие ИМС характеризовались узкоспециализированными неизменяемыми функциями и предназначались для выполнения строго определенной работы. Фирма Intel не имела опыта работы с подобными микросхемами, но, как оказалось, это было только к лучшему. Взглянув на проблему "свежим взглядом", сотрудник Intel Тэд Хофф решил, что предлагаемый набор из многих ИМС слишком сложен, чтобы считать его производство экономически выгодным. Хофф, кстати, пришел на фирму, желая поработать над идеями, имеющей экономическую перспективу. Он нашел остроумное альтернативное решение, предложив сократить количество ИМС с 12 до 4-х, добавив к ним универсальную микросхему, которая бы выполняла функции сразу нескольких ИМС. Так была предложена идея микропроцессора, разработку которого выполнили сотрудники Intel Федерико Фэджин и Масатоси Сима (впоследствии они организовали фирму Zilog). Это и был МП 4004.

В сочетании с еще тремя микросхемами (4001 - микросхема масочной ROM емкостью 2048 бит; 4002 - микросхема RAM емкостью 320 бит; 4003 - микросхема интерфейса, представляющая собой 10-разрядный сдвиговый регистр) этот МП представлял, по сути, микро ЭВМ - машину, не уступавшую по мощности большим ЭВМ середины 50-х годов. Он заключал в себе столько же вычислительной мощи, сколько первый электронный компьютер ENIAC, созданный в 1946 г. и занимавший целую комнату. Первые области применения 4004 были весьма разнообразны – от контроллеров автоматического светофора и анализаторов крови до космического зонда Pioneer 10, который был запущен NASA в 1972 году. Последний сеанс связи с зондом был в 2003 году, когда Pioneer 10 находился на расстоянии 12,21 млрд. км от Земли.

Изначально Busicom приобрела права на микропроцессор, заплатив Intel $60 тыс. Осознав тот огромный потенциал, который скрывал в себе «интеллектуальный» чип, Intel решила возвратить Busicom эти деньги, чтобы вернуть себе права на микропроцессор. Busicom согласилась, и 15 ноября 1971 г. фирма Intel представила миру МП 4004. Он продавался по цене $200 за штуку.

Последующие поколения МП фирмы Intel представляли собой 8-, 16-, 32- и 64-разрядные устройства (см. табл. 1.1).

Таблица 1.1. Микропроцессоры фирмы Intel до 2000 г

Тип	Год разработки	Тактовая частота	Разрядность	Число транзисторов	Адресуемая память	Виртуальная память	Примечание
4004	1971	108 кГц	4	2300 (10 мкм)	640 байт		Первый микропроцессор, арифметические действия
8008	1972	108 кГц	8	3500 (10 мкм)	16 Кб		Действия с данными и символами
8080	1974	2 МГц	8	6000 (6 мкм)	64 Кб		10-кратная производительность 8008
8086	1978	10 МГц	16	29000 (3 мкм)	1 Мб		10-кратная производительность 8080
8088	1979	8 МГц	8	29000 (3 мкм)	1 Мб		Идентичен 8086 кроме 8-битной шины данных
80286	1982	12 МГц	16	134000 (1.5 мкм)	16 Мб	1 Гб	3-6-кратная производительность 8086
Intel386 DX	1985	33 МГц	32	275000 (1 мкм)	4 Гб	64 Тб	Первый X86 32-битный процессор
Intel386 SX	1988	20 МГц	32	275000 (1 мкм)	4 Гб	64 Тб	16-битная адресная шина дешевого 32-битного процессора
Intel486 DX	1989	50 МГц	32	1,2 млн. (0,8 мкм)	4 Гб	64 Тб	Кэш 1-го уровня на кристалле
Intel486 SX	1991	33 МГц	32	1185000 (0,8 мкм)	4 Гб	64 Тб	Идентичен Intel486 DX, но без мат. сопроцессора
Pentium®	1993	166 МГц	32	3,1 млн. (0,8 мкм)	4 Гб	64 Тб	Суперскалярная архитектура с 5-кратной производительностью 33-МГц Intel486 DX
Pentium® Pro	1995	200 МГц	32	5,5 млн. (0,32 мкм)	4 Гб	64 Тб	Высокопроизводительный процессор с архитектурой динамического исполнения
Pentium® II	1997	266 МГц	64	7 млн. (180 нм)	64 Гб	64 ТБ	
Pentium® Ш	1999	500 МГц	64	8,2 млн. (130 нм)	64 Гб	64 ТБ	

За прошедшие годы в 20 веке количество транзисторов на кристалле возросло в 5000 раз, тактовая частота - в 5000 раз, а эквивалентная производительность МП увеличилась на 3 - 4 порядка величин. По своим показа-

телям темпы микропроцессорной эволюции не имеют себе равных - они даже превосходят интенсивность развития обычных средств вычислительной техники. Время, прошедшее между изобретением микропроцессора и внедрением 32-разрядной архитектуры, составляет 10 лет, т.е. почти вдвое меньше, чем для ЭВМ.

В настоящее время интенсивно продолжается разработка новых архитектур микропроцессоров, после CISC и RISC-процессоров выпускаются VLIW-процессоры со статическим параллелизмом с большой длиной команд, которые характеризуются тем, что одна инструкция процессора содержит несколько операций, которые могут выполняться параллельно. Благодаря параллельной обработке увеличивается вычислительная мощность. Кроме того, за счет интеграции памяти и периферийных компонентов на кристалле (SoC) уменьшается число необходимых для всей системы внешних микросхем с большим временем доступа. В 2006 году Intel начал выпуск процессоров серии Core, базирующихся на многоядерной архитектуре. Последний модельный ряд Core i9 имеет процессоры с 18 ядрами, работающие с тактовой частотой до 5,3 ГГц, выпускаемые по 10 нм техпроцессу (определение техпроцесса в настоящее время не соответствует тому, что было в 20 веке), количество транзисторов достигло 6 млрд.

Представленные на рынке современные микропроцессоры имеют на кристалле от 1 млн. до 9 млрд. транзисторов. Степень интеграции определяется размерами структурного элемента, который составляет в настоящее время 0,2 - 0,5 мкм (ячейка статической памяти из 6 транзисторов FinFET). Дальнейшая миниатюризация до 0,1 мкм будет реализована без особых усложнений технологии. Другой способ увеличения количества транзисторов на кристалле - увеличение его площади. Современные микропроцессоры имеют размеры от 10х10 мм^2 до 17х17 мм^2 и их увеличение проблематично, так как процент брака из-за дефектов в кристалле растет в квадратичной зависимости от его площади. Тем не менее площади в 25х25 мм^2 стали реальностью.

Для размещения большого количества элементов на кристалле используется многослойная металлизация. В МП типа Motorola 68040 или Intel 486 применялись только 2 слоя, а в PowerPC имел уже 5 слоев, из которых для соединений используются четыре, а пятый - обеспечивает связь с выводами корпуса. В современных МП количество слоев многократно выросло. Возросла проблема с выводом сигналов “во внешний мир” - некоторые МП насчитывают на корпусе сейчас более 1000 выводов. Решением ее, как это уже делается в дешевых МП, является многократное мультиплексирование отдельных выводов.

Помимо числа транзисторов, важной характеристикой МП является его тактовая частота. Однако с возрастанием частоты появляются трудности: требуется соответствующая высокочастотная технология, а также подключение запоминающих устройств становится все более дорогим. Кроме того, растет расход мощности и, следовательно, как результат - тепловыделение в кристалле, что нежелательно, так как их срок службы находится в обратной зависимости от температуры, при которой они эксплуатируются. До 1990 года микропроцессоры работали с напряжением питания 5 вольт, однако при ширине структурных элементов меньше 0,5 мкм напряжение питания снизилось до 3 вольт и далее до 1 вольта. Со снижением напряжения падает и потребляемая мощность.

В начале развития микропроцессоры изготовлялись по p-МОП технологии, затем стали отдавать предпочтение комплементарной МОП технологии (КМОП). Теперь применяется множество разнообразных видов технологий: n-МОП с обогащением и обеднением, биполярная, интегрально-инжекционная $И^2Л$ и другие. Основным материалом для изготовления микропроцессоров является кремний, однако инвестируются большие средства в разработку процессоров на основе арсенида галлия (фирма Cray Computer). Использование арсенида галлия дает 3-х кратный выигрыш в быстродействии, кроме того он более радиационно-стойкий, чем кремний.

1.2. Базовая структура микропроцессорной системы

Обобщенная структурная схема микропроцессорного устройства (МПУ) изображена на рис. 1.1. В общем случае, работа МПУ сводится к следующей последовательности действий микропроцессора: получение данных от различных внешних устройств (ВУ) (с клавиатуры, из каналов связи, внешних запоминающих устройств и др.), обработка данных и выдача результатов обработки на ВУ. При этом данные от ВУ, подлежащие обработки, могут поступать и в процессе их обработки.

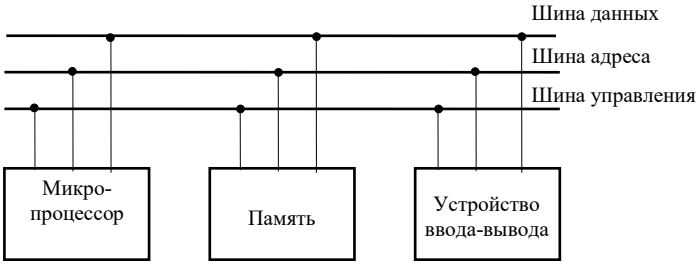


Рис.1.1. Обобщенная структурная схема микропроцессорного устройства

Для выполнения этих действий в МПУ кроме микропроцессора предусмотрены следующие устройства: память, предназначенная для хранения программ, определяющих работу МП, и различных данных (исходных, промежуточных и конечных результатов обработки данных), устройства ввода-вывода, обеспечивающие обмен данными различных ВУ с процессором и памятью. Кроме того, имеются линии связи, объединенные по функциональному назначению в шины: шина данных, адресная шина и шина управления.

Следует отметить, что обмен данными с памятью зависит от архитектуры процессора. Неймановская архитектура размещает программы и данные в одном адресном пространстве, а гарвардская – в разных, т.е. имеется отдельная память программ и память данных, и обращение к ним происходит с участием разных сигналов.

В ходе работы МП выдает на шину адреса номер ячейки памяти, в которой хранится очередная команда (инструкция), а по шине управления в память поступают сигналы, обеспечивающие чтение содержимого ячейки памяти. Запрошенная команда выдается на шину данных, откуда она принимается микропроцессором. Далее команда расшифровывается, и если данные, действие над которыми предусматривает команда, находятся в регистрах процессора, то он приступает к выполнению указанной в команде операции. Если при расшифровке команды выясняется, что участвующие в операции данные находятся в памяти, то МП опять выдает на адресную шину адрес ячейки, хранящей эти данные, в сопровождении с управляющим сигналом чтения данных, и после выдачи их из памяти на шину данных считывает и выполняет указанную в команде операцию. После завершения выполнения текущей команды на шину адреса выдается адрес следующей команды и описанный процесс повторяется.

В процессе работы микропроцессорного устройства может потребоваться выдача результата обработки данных на внешние устройства, либо прием данных от них. Для этого группа ВУ подключается к шинам МПУ через устройство ввода-вывода (устройство сопряжения), обеспечивающее необходимые уровни сигналов и управляющее процессом обмена данными в соответствии с, так называемым, протоколом обмена данными с ВУ. Если устройства сопряжения программируемые, то до начала обмена МП должен выдать туда информацию о режимах, используемых при передаче, направлениях передачи данных и другие установки. Затем, в момент, когда потребуется, например, передать поступающие из ВУ данные в оперативную память, МП, выполняя команду ввода данных, подает на устройство ввода-вывода соответствующие управляющие сигналы. Данные из ВУ при-

нимаются в регистр устройства ввода-вывода (порт), откуда они поступают на шину данных. Далее, эти данные с шины принимаются микропроцессором в регистр, после чего в процессе выполнения другой команды они передаются в оперативную память. Аналогично происходит обмен данными в обратном направлении.

Описанные способы обмена данными предполагают, что моменты обмена известны заранее уже на этапе программирования, и в программе предусматриваются в определенных местах соответствующие команды. Однако существуют ситуации, например, в случае аварии на ВУ, когда обмен данными должен определяться самим внешним устройством. В этих случаях ВУ, подавая в микропроцессор определенные сигналы, переводит его в режим так называемого прерывания. В этом режиме МП прекращает выполнение основной (фоновой) программы и переходит к исполнению команд другой, находящейся в памяти программе обработки прерывания, обеспечивающей обмен данными по запросу ВУ. После окончания выполнения такой программы микропроцессор возвращается к продолжению основной программы.

Эти способы обеспечивают низкую скорость обмена данными, и применять их целесообразно лишь с низкоскоростными ВУ. При работе с высокоскоростными ВУ используется, так называемый, режим прямого доступа к памяти (ПДП). В этом режиме микропроцессор отключается от шин, предоставляя их в распоряжение ВУ для непосредственного обмена данными с памятью (без участия МП). Обмен при этом организуется специальным контроллером ПДП. В режиме ПДП ВУ для большей эффективности обменивается не одиночными данными, а блоками данных. Перед обменом в таком режиме, МП предварительно передает в контроллер ПДП информацию, необходимую для управления обменом: адреса источника и назначения данных и размер блока данных. В процессе обмена контроллер ПДП выдает на шины адрес ячейки памяти, данные и управляющие сигналы, т.е. заменяет собой, таким образом процессор. После завершения передачи заданного размера блока данных контроллер ПДП извещает об этом микропроцессор и тот, в свою очередь, восстанавливает связь с шинами и продолжает выполнение основной программы.

2. ПРИНЦИПЫ ПОСТРОЕНИЯ ПРОЦЕССОРА

2.1. Общая структура процессора

Процессор предназначен для обработки данных, он состоит из двух устройств - операционного и управляющего (см. рис. 2.1). Операционное устройство (ОУ) - устройство, в котором выполняются определенные операции. Оно включает в себя в качестве узлов регистры, сумматор, каналы передачи данных, мультиплексоры для коммутации каналов, шифраторы, дешифраторы и другие узлы, в зависимости от заданных требований к процессору.

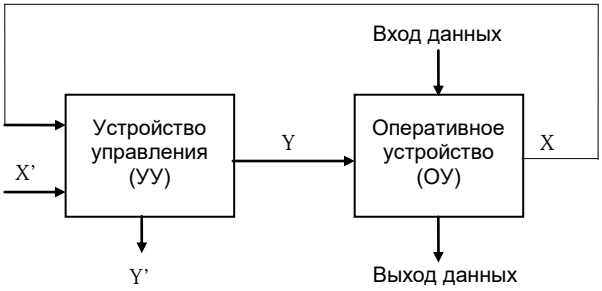


Рис.2.1. Общая структура процессора

Управляющее устройство (УУ) координирует действия узлов ОУ. Оно вырабатывает в определенной временной последовательности управляющие сигналы Y(X,X'), под действием которых в узлах ОУ выполняются требуемые действия. Процессор имеет внешние выводы, по которым поступают данные и в обработанном виде выдаются для дальнейшего использования, а также входные и выходные линии управления X' и Y', соответственно, связанные с внешними устройствами. На рис. 2.1 все изображенные линии обозначают шины, которые могут иметь различную ширину (различное количество линий).

Последовательность функционирования ОУ распадается на последовательность элементарных действий в его узлах. Такими элементарными действиями могут быть:

1) установка регистра в некоторое состояние:
   $R1 \Leftarrow 0$;

2) инвертирование разрядов содержимого регистра:
R2 ⇐ (R2’); (символ ‘ в дальнейшем определяет инверсию)
3) пересылка содержимого одного регистра в другой:
R1 ⇐ R2;
4) сдвиг содержимого регистра влево:
R1 ⇐ СдвЛ(R1);
5) счет, при котором содержимое число в регистре возрастает на 1:
R1 ⇐ (R1) + 1;
6) сложение:
R2 ⇐ (R2) + (R1);
7) сравнение содержимого регистра на равенство с некоторым числом: результат - либо 0, либо 1;
8) некоторое логическое действие: поразрядное выполнение операции или конъюнкции, или дизъюнкции, или др.

Каждое такое элементарное действие, выполняемое в одном из узлов операционного устройства в течение одного тактового периода, называется микрооперацией. В определенные тактовые периоды одновременно могут выполняться несколько микроопераций. Такая совокупность одновременно выполняемых микроопераций называется микрокомандой. А весь набор микрокоманд, предназначенный для выполнения определенной операции - микропрограммой.

Если в ОУ предусматривается возможность выполнения n различных микроопераций, то из УУ выходят n управляющих цепей, каждая из которых соответствует определенной микрооперации. И для выполнения некоторой микрооперации достаточно по управляющей цепи, соответствующей данной микрооперации, подать сигнал, например, напряжение уровня логической 1.

2.2. Проектирование устройства управления

Существует два подхода к проектированию управляющего устройства: использование схемной логики (аппаратный) и использование программируемой логики (микропрограммный).

В первом случае, из набора интегральных микросхем или, используя микроэлектронную технологию, создаются такие схемы, которые обеспечивают требуемое функционирование УУ. В качестве их узлов используются комбинационные и последовательностные схемы для выработки требуемых управляющих сигналов.

Комбинационные схемы строятся на основе логических элементов. Эти схемы характеризуются отсутствием памяти, и поэтому их выходные сигналы определяются только входными сигналами (рис. 2.2). Алгоритм работы устройства на основе комбинационной схемы может быть представлен в виде таблицы соответствия, содержащей значения выходных сигналов для всех возможных комбинаций значений входных сигналов.

Последовательностные схемы отличаются от комбинационных схем, прежде всего, наличием памяти. Их выходные сигналы являются функцией не только входных сигналов, но и функцией внутреннего состояния, в котором устройство находилось до поступления входных сигналов. На рис. 2.3 показана логическая организация этих схем. Память в них может быть любого типа - здесь она представляет собой совокупность триггеров T_1, T_2,...,T_p. Такая архитектура является универсальной и может рассматриваться в качестве модели любой цифровой системы независимо от ее сложности.

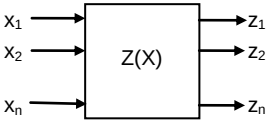


Рис. 2.2. Комбинационная схема

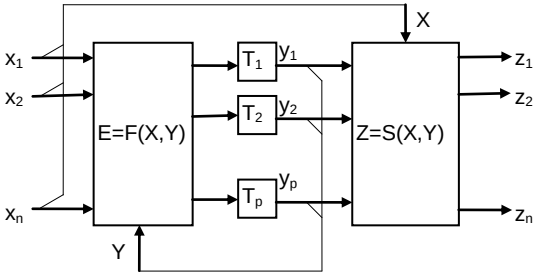


Рис. 2.3. Последовательностная схема

Связь между входными и выходными сигналами описывается следующим образом. Имеется n двоичных входов x_1... x_n, т.е. существует 2^n комбинаций, каждая из которых может рассматриваться как вектор:

$X = [x_1,x_2,...,x_n]$, где x_i - либо 0, либо 1.

Имеется m двоичных выходов z_1....z_m. Выходной вектор имеет вид:

$Z = [z_1,z_2,...,z_m]$, где z_i - либо 0, либо 1.

Выходной вектор памяти, определенный на выходе триггеров задается в следующем виде:

$Y = [y_1,y_2,...,y_p]$, где y_i - либо 0, либо 1.

Связь между векторами может быть представлена следующими функциями:

выходов $Z = S(X,Y)$;

возбуждения $E = F(X,Y)$

переходов $Y' = G(X,Y)$

Функция переходов G, определяющая выходной вектор памяти - это не булевская функция, описывающая комбинационную схему, а уравнение следующего состояния. Y' - это состояние памяти, в которое она перейдет из текущего состояния Y при поступлении на вход схемы сигналов.

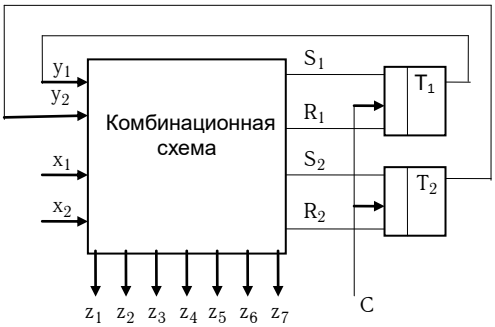


Рис. 2.4. Структурная схема управляющего устройства на основе схемной логики для процессора умножения целых чисел

Устройства, построенные на основе таких схем, обладают наивысшим быстродействием, однако узкая специализация и, следовательно, малосерийность делает их дорогостоящими. На рис. 2.4 изображено управляющее устройство простого процессора, выполняющего умножение n-разрядных целых чисел без знака.

Другой подход предполагает построение устройства управления с использованием одной или нескольких больших интегральных схем (БИС) универсального устройства, в котором требуемое функционирование обеспечивается занесением в управляющую память устройства определенной микропрограммы. В этом случае число БИС, необходимых для построения УУ, небольшое, и это обеспечивает целесообразность их выпуска промышленностью массовыми тиражами.

В этом случае управляющие сигналы записываются в управляющую память в виде микрокоманд. При этом, выполнение операции сводится к последовательной выборке из управляющей памяти микрокоманд и выдачи их в операционное устройство в виде сигналов $z_1...z_m$. В управляющей памяти можно хранить много микропрограмм, предназначенных для выполнения различных операций. Таким образом устроены современные микропроцессоры, выполняющие сотни операций. По выбранной из оперативной памяти инструкции (команды) в управляющей памяти находится соответствующая микропрограмма, микрокоманды которой последовательно считываются в ОУ для выполнения заданной операции.

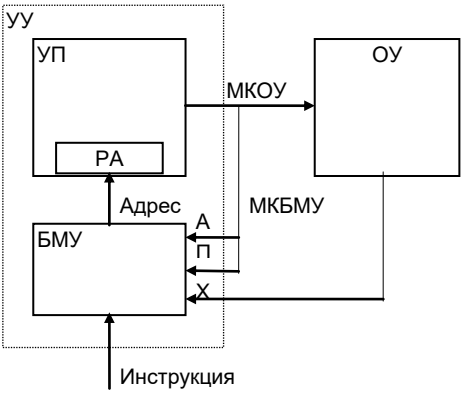


Рис. 2.5. Структурная схема процессора с микропрограммным способом управления

На рис. 2.5 изображена структурная схема процессора с управляющим устройством, построенным на основе программируемой логики. Блок микропрограммного управления (БМУ) определяет адрес очередной микрокоманды (МК) в управляющей памяти (УП). Поступающая из оперативной памяти инструкция содержит код операции (КОП), определяющий адрес первой микрокоманды реализуемой операции. Адреса всех последующих микрокоманд определяется в БМУ.

Поле адреса				Поле условных переходов			Поле управляющих сигналов						
				П	x_1	x_2	z_1	z_2	z_3	z_4	z_5	z_6	z_7
МК БМУ							МК ОУ						

Рис.2.6. Формат микрокоманды для операции умножения целых чисел без знака

В формате микрокоманды, изображенной на рис. 2.6, имеется поле адреса, которое содержит адрес очередной МК. Причем, при разветвлении программы этот адрес может модифицироваться в зависимости от значения условия. Для этого предназначено поле условных переходов, в котором в разряде вида перехода П указывается, имеет ли место условный или безусловный переход (0 - безусловный, 1 - условный), а в разрядах условий x_1x_2 признак использования их значений. Формирование адреса осуществляется замещением младшего разряда содержимого поля адреса текущей микрокоманды значением соответствующего условия, если разряд перехода П указывает на необходимость его изменения.

Так как на выполнение микрокоманд, модифицирующих адреса переходов, затрачиваются тактовые периоды, быстродействие процессора ухудшается по сравнению с аппаратным управлением.

2.3. Конвейерный способ организации управления

Быстродействие процессора сильно зависит от длительности тактового периода, за время которого должен быть сформирован адрес очередной МК, далее микрокоманда должна быть прочитана из управляющей памяти, а затем выполнена в операционном устройстве. На рис. 2.7 приведена временная диаграмма работы процессора с микропрограммным управлением.

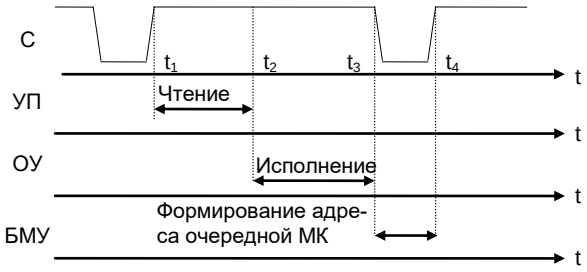


Рис. 2.7. Временная диаграмма работы процессора с микропрограммным управлением

В момент времени t_1 происходит прием в регистр адреса РА УП (см. рис.2.5) сформированного адреса следующей микрокоманды. Далее адрес из РА выдается на внутреннюю шину адреса и поступает непосредственно в память, где происходит чтение очередной микрокоманды. Процесс чтения завершается к моменту времени t_2, и с этого момента в ОУ поступает мик-

рокоманда и начинается процесс ее выполнения, который завершается к моменту t_3. В момент t_4 полученный результат фиксируется в соответствующем регистре ОУ. В БМУ в интервале $t_3...t_4$ под действием МК БМУ происходит формирование адреса очередной микрокоманды и фиксация его в момент t_4 в регистре адреса УП.

Последовательный способ чтения и исполнения микрокоманды задает длительность тактового периода, определяющего быстродействие процессора. Этот недостаток может быть устранен использованием конвейерного способа чтения и исполнения микрокоманды. При этом способе осуществляется параллельное чтение и исполнение микрокоманды. В процессе исполнения в ОУ n-ой МК из УП производится чтение (n+1)-ой МК, и в том же тактовом периоде в БМУ формируется адрес (n+2)-ой МК. Реализация этого способа требует использования конвейерного регистра, изображенного на рис. 2.8. Временная диаграмма работы процессора с конвейерным регистром показана на рис. 2.9.

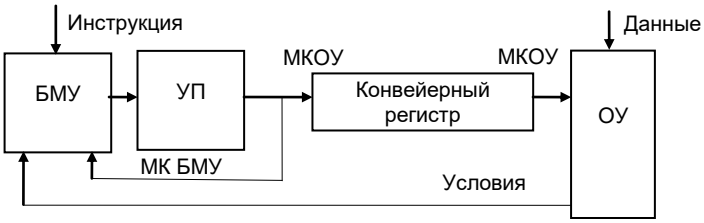


Рис. 2.8. Структура процессора с конвейерным регистром

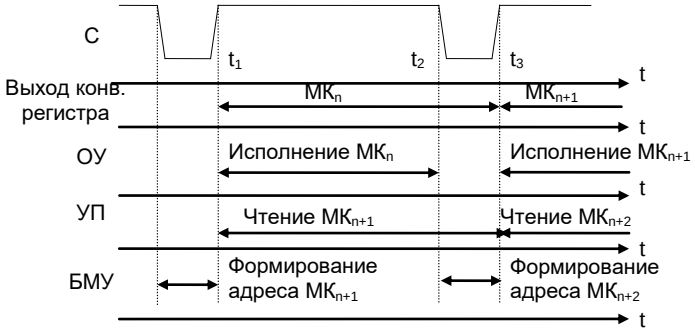


Рис. 2.9. Временная диаграмма работы процессора с конвейерным регистром

В интервале времени $t_1...t_3$ из конвейерного регистра выдается n-я МК ОУ, прочитанная из УП на предыдущем такте. В интервале времени $t_1...t_2$ она исполняется в ОУ. В этом же такте производится чтение из УП (n+1)-

ой МК по адресу, сформированному в БМУ в предыдущем такте. В интервале времени $t_2...t_3$ в БМУ формируется адрес (n+2)-й МК.

Введение конвейерного регистра усложняет микропрограмму, так как в период времени выработки условия в ОУ для формирования следующего адреса микрокоманды, выполняемой по условию, в микропрограмму необходимо вводить холостые микрокоманды, не изменяющие состояния ОУ по мере их поступления туда из конвейерного регистра.

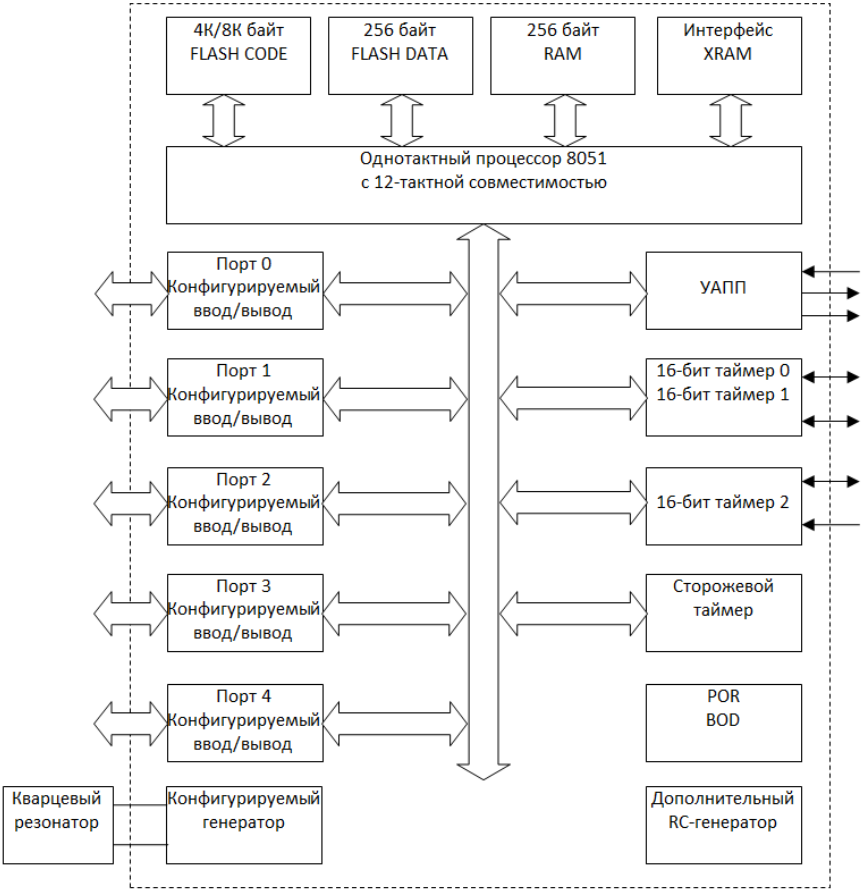
При использовании управляющего устройства с конвейерным регистром длительность тактового периода может оказаться той же, что и в устройстве, построенным на схемной логике. Однако и в этом случае на выполнение того же алгоритма остается все же больше тактовых периодов.

3. АРХИТЕКТУРА МИКРОПРОЦЕССОРА

Архитектура микропроцессора - это его логическая организация, определяемая возможностями МП по аппаратной и программируемой реализации функций, необходимых для построения микропроцессорной системы. Понятие архитектуры включает в себя:

- структуру МП;
- способы обращения ко всем доступным пользователю элементам структуры;
- способы представления и форматы данных;
- набор операций, выполняемых МП;
- способы адресации данных, участвующих в операциях;
- форматы управляющих слов, поступающих от периферийных устройств;
- характеристики и назначения управляющих сигналов, вырабатываемых МП;
- реакцию МП на внешние сигналы.

Архитектуру микропроцессора рассмотрим на примере центрального процессорного элемента микроконтроллера AT89LP51/52. Упрощённая структура микроконтроллера показана на рис. 3.1.



POR –Power-On Reset; BOD – Brown-Out Detection

Рис.3.1. Упрощенная структурная схема микроконтроллера AT89LP51/52

Особенности архитектуры микроконтроллера AT89LP51/52

• 8-разрядный микроконтроллер, совместимый с 8051 (1980 г.).
• Улучшенная архитектура 8051:
- один такт на выборку байта;
- 12 тактов в режиме совместимости машинного цикла;
- до 20 MIPS (Мега инструкций/сек) при тактовой частоте 20 МГц;
- полностью статические операции: от 0 Гц до 20 МГц;
- внутренний 2-х тактный аппаратный умножитель цикла;
- внутреннее ОЗУ 256 х 8;
- интерфейс внешней памяти данных/программы;
- два указателя данных;
- 4 уровня приоритета прерываний.
• Энергонезависимая память программ и данных:
- 4К/8К байтовая программируемая в системе (ISP) флэш-память программ;
- 256 байтовая флэш-память данных;
- 256 байтовый сигнатурный массив пользователя;
- 10 000 циклов записи/стирания;
- последовательный интерфейс для загрузки программы;
- быстрый режим программирования 64 байтовыми страницами;
- 3-х уровневая программа для блокировки памяти (Software Security);
- программирование памяти программ из приложения.
• Периферийные особенности:
- три 16-разрядных таймера/счетчика;
- улучшенный приемопередатчик:
- автоматическое распознавание адреса;
- обнаружение ошибки кадра;
- режимы эмуляции интерфейсов SPI (Serial Peripheral Interface) и TWI (Two-wire Serial Interface) для программирования микроконтроллера;
- программируемый сторожевой таймер с программным сбросом и предварительным делителем.
• Специальные функции микроконтроллера:
- обнаружение неустойчивого питания (BOD) и сброс по включению питания (POR) с флагом Power-off;
- выбор полярности на выводе внешнего сброса;
- энергосберегающие режимы Low Power Idle и Power-down;
- восстановление по прерыванию из режима Power-down;
- дополнительный внутренний генератор 1,8432 МГц.
• Ввод-вывод и корпуса:

• Конфигурируемые режимы портов (по 8 бит на порт):
  - до 36 линий программируемого ввода-вывода;
  - квази-двунаправленный (80C51);
  - двунаправленный (Tristate);
  - подтянутый к питанию CMOS выход;
  - с открытым стоком.
• Экологические корпуса (без свинца/галогенида):
  - 40-выводный PDIP;
  - 44-выводый корпус TQFP/PLCC;
  - 44-выводный VQFN/MLF.
• Условия эксплуатации:
  - диапазон напряжений VCC 2.4V - 5.5V;
  - диапазон температур от -40 ° C до 85 ° C;
  - от 0 до 20 МГц @ 2.4V–5.5V;
  - от 0 до 25 МГц @ 4.5V–5.5V.

3.2. Структура процессора

Структурная схема процессора 8051 приведена на рис. 3.1.

Процессор 8051 является функционально законченным параллельным 8-разрядным устройством с фиксированной системой команд. Его структурная схема изображена на рис. 3.2.

В состав процессора входят:

- 8-разрядное арифметико-логическое устройство (ALU);
- регистр состояний (PSW), фиксирующий вырабатываемые АЛУ признаки в процессе выполнения операций, а также статусную информацию;
- аккумулятор (A) и расширитель аккумулятора (B)
- десятичный корректор (DAA), выполняющий преобразование двоичного кода в двоично-десятичный код;
- регистр команд (IR), предназначенный для хранения первого байта инструкции, содержащего код операции;
- блок регистров для приема, хранения и выдачи информации в процессе выполнения программ, содержащий программный счетчик команд (PC), регистр адреса данных (DPTR), указатель стека (SP), шесть регистров общего назначения (R0, R1, R2, R3, R4, R5, R6, R7) и вспомогательный регистр (TP1);

- схемы управления и синхронизации на основе программируемой логики, формирующие последовательность управляющих сигналов для работы АЛУ и блока регистров;
- управляющую память на основе программируемой логики;
- буферные регистры адреса и данных;
- устройства, обеспечивающие работу процессора в режиме прерывания;
- другие схемы для обеспечения обмена данными между узлами процессора и внутренней шиной данных.

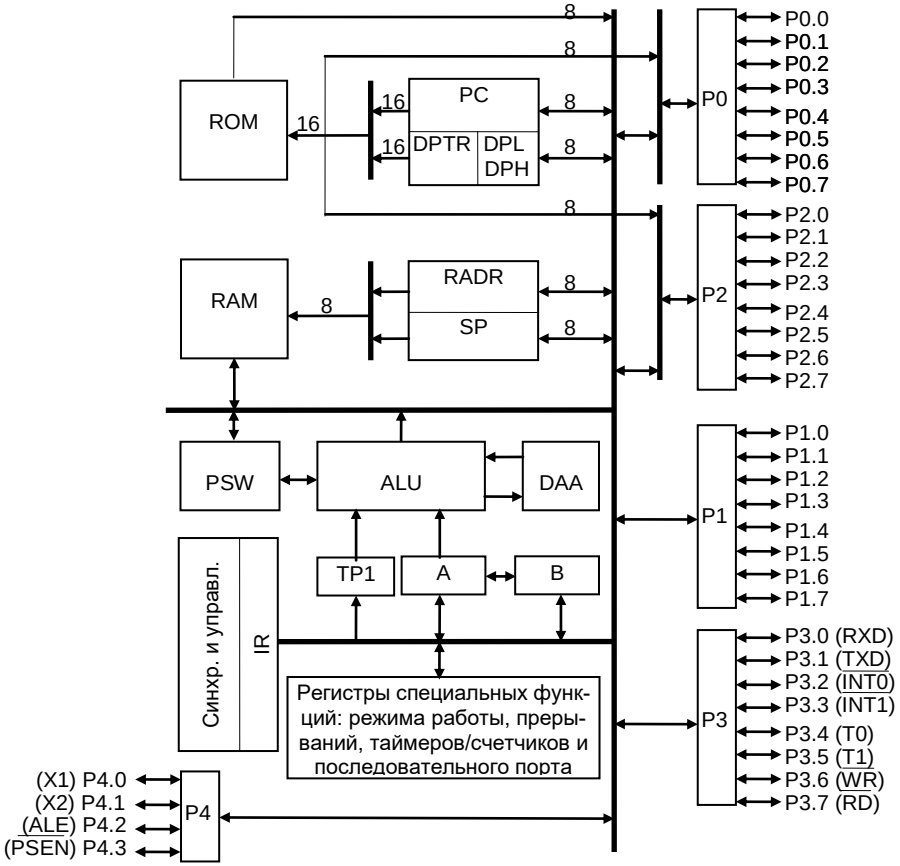


Рис. 3.2. Структура процессора

Порты ввода/вывода

Порты ввода/вывода P0, P1, P2 и P3 работают в четырех режимах. Значение по умолчанию зависит от fuse-бита Tristate-Port. Когда fuse-бит установлен, все порты ввода/вывода при включении питания или сбросе возвращаются в режим только для ввода (в третьем состоянии). Если fuse-бит не активен, порты P1, P2 и P3 стартуют в квази-двунаправленном режиме, а P0 в двунаправленном режиме с открытым стоком. Порт P4 всегда работает в квази-двунаправленный режиме. P0 может быть настроен на внутреннее подтягивание, задав его в квази- или двунаправленном режимах вывода.

Альтернативные функции порта P3

Символ	Позиция	Функция
RXD	P3.0	Вход приемника последовательного порта
TXD	P3.1	Выход передатчика последовательного порта
INT0’	P3.2	Вход внешнего прерывания 0
INT1’	P3.3	Вход внешнего прерывания 1
T0	P3.4	Внешний вход или выход таймера/счетчика 0
T1	P3.5	Внешний вход или выход таймера/счетчика 1
WR’	P3.6	Строб Write интерфейса внешней памяти (активный-низкий уровень).
RD’	P3.7	Строб Read интерфейса внешней памяти (активный-низкий уровень).

Альтернативные функции порта P4

Символ	Позиция	Функция
X1	P4.0	Вход в инвертирующий усилитель генератора и внутренние схемы генерации тактовых импульсов. Его можно использовать как вывод порта, если в качестве источника тактового сигнала выбран внутренний RC-генератор.
X2	P4.1	Выход инвертирующего усилителя генератора. Его можно использовать в качестве вывода порта, если в качестве источника тактового сигнала выбран внутренний RC-генератор или внешний тактовый генератор.
ALE	P4.2	Разрешение защелки адреса в интерфейсе внешней памяти.
PSEN’	P4.3	Разрешение строба программы в интерфейсе внешней памяти (активный-низкий уровень).

Регистры данных. Регистр A (аккумулятор) предназначен для обмена информацией с внешними устройствами. При выполнении арифметических, логических и сдвиговых операций аккумулятор служит источником операнда, в него же помещается результат выполненной операции. Расши-

ритель аккумулятора (B) используется для увеличения разрядности до 16 бит. 16-разряднай аккумулятор нужен в операциях умножения и деления.

Блок регистров общего назначения R0, R1, R2, R3, R4, R5, R6, R7 используется для размещения 8-разрядных данных. Регистры размещены в оперативной памяти RAM, занимая 32 адреса с 00H по 1FH, и сгруппированы в 4 банка, подключаемых из регистра состояний PSW (см. табл.3.1).

Указатель стека. 8-разрядный регистр служит для адресации особого вида памяти, называемый стеком. Стек - память с упрощенной формой адресации байтов, размещение которых осуществляется по правилу: последний вошел, первый вышел (LIFO). В качестве стека может использоваться любая область оперативной памяти. Указатель стека содержит адрес входа в стек (вершины стека). При записи в стек байта (см. рис. 3.3) содержимое указателя стека увеличивается на 1: SP+1, и по этому адресу происходит запись. После чтения из стека содержимое регистра указателя стека уменьшается на 1: SP-1. Такой способ адресации элементов стека называется прединкрементным/постдекрементным.

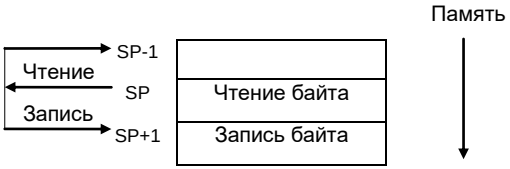


Рис. 3.3. Содержимое указателя стека при записи и чтении данных

Счетчик команд. 16-разрядный регистр PC предназначен для хранения текущего адреса инструкции. Он автоматически увеличивается на 1, 2 или 3 и т.д., в зависимости от формата выполненной команды, или загружается новым адресом при переходе или вызове подпрограммы.

Указатель данных. 16-разрядный регистр DPTR предназначен для фиксации адреса в операциях с обращением к внешней памяти данных. Он используется и как 16-разрядный регистр, и как два независимых 8-разрядных регистра DPH и DPL. Также в качестве указателей могут быть применены регистры общего назначения R0 и R1.

Арифметико-логическое устройство (АЛУ). В 8-разрядном АЛУ предусмотрена возможность выполнения арифметических операций: сложение с учетом переноса и без него, вычитание с учетом заёма, умножение и деление; четырех логических операций: конъюнкции, дизъюнкции, неравнозначности и сравнения и четырех видов циклического сдвига. При выполнении арифметических и логических операций одним из операндов

служит содержимое аккумулятора, результат операции помещается туда же. Циклический сдвиг выполняется только над содержимым аккумулятора.

В АЛУ предусмотрена возможность выполнения арифметических операций над десятичными числами. При хранении десятичного числа разряды регистра делятся на два ниббла размером 4 разряда, в каждом из которых находится одна десятичная цифра, представленная в коде 8421. При выполнении операций суммирования десятичных чисел, если результат окажется больше 9, может потребоваться его коррекция путем прибавления к нему двоичного числа 0110. Такая коррекция выполняется схемой десятичной коррекции (DAA).

Регистр состояний. Для управления процессом выполнения программы используется слово состояния программы PSW, которое содержит флаги признаков, определяемых результатом выполнения арифметических и логических операций, и статусные биты. Формат регистра слова состояния программы представлен в табл. 3.1.

Табл.3.1. Формат регистра PSW

Символ	Позиция	Название и назначение
C	PSW.7	Флаг переноса. Устанавливается и сбрасывается аппаратными средствами или программой при выполнении арифметических и логических операций
AC	PSW.6	Флаг вспомогательного переноса. Устанавливается и сбрасывается только аппаратными средствами при выполнении команд сложения и вычитания и сигнализирует о переносе или заеме в бите 3
F0	PSW.5	Флаг 0. Может быть установлен, сброшен или проверен программой как флаг, специфицируемый пользователем
RS1	PSW.4	Выбор банка регистров. Устанавливается и сбрасывается программой для выбора рабочего банка регистров. (См. примечание)
RS0	PSW.3	
OV	PSW.2	Флаг переполнения. Устанавливается и сбрасывается аппаратно при выполнении арифметических операций
-	PSW.1	Не используется
P	PSW.0	Флаг паритета. Устанавливается и сбрасывается аппаратно и фиксирует четное/нечетное количество единичных бит, т.е. выполняет контроль по четности/нечетности

Примечание. Выбор рабочего банка регистров

RS1	RS0	Банк	Границы адресов
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

Наиболее активным является флаг переноса, который принимает участие и модифицируется в процессе выполнения операций, включая сложение, вычитание и сдвиги. Кроме того флаг переноса выполняет функцию битового аккумулятора в булевых операциях. Флаг переполнения фиксирует арифметическое переполнение при операциях с целыми числами со знаком и делает возможным использование арифметики с числами в дополнительном коде.

Блок управления. Он состоит из регистра инструкций (команд), куда принимается первый байт инструкции, и управляющего устройства, формирующего сигналы, под действием которых в отдельных его узлах выполняются микрооперации. Управляющее устройство содержит выполненную на программируемой логике управляющую память для размещения там микропрограмм всех операций. Изменить содержимого в управляющей памяти пользователь не может, и поэтому МП имеет фиксированный набор команд.

Буферные регистры. Буферные регистры данных и адреса обеспечивают связь центрального процессора с внешним миром. Особенность их в том, что они имеют три состояния - кроме низкого и высокого уровней напряжения имеется высокоимпедансное состояние, при котором микропроцессор отключается от внешних шин, позволяя использовать их другим внешним устройствам.

3.3. Формат данных и инструкций

Данные в микропроцессоре представлены в виде 8-разрядных (однобайтовых) кодов. Для идентификации отдельных разрядов (бит) в байте они нумеруются справа налево D7...D0. Однобайтовый код может рассматриваться либо как положительное целое число от 0 до 255, либо как целое число со знаком в дополнительном коде в диапазоне от -128 до +127. Для целых чисел можно использовать двухбайтовое и многобайтовое представление.

Форматы инструкций могут быть одно-, двух-, трехбайтовыми и более. Многобайтовая инструкция размещается в последовательно расположенных ячейках памяти, и в первом байте всегда находится код операции.

3.4. Режимы адресации

Для обращения к обрабатываемым процессором данным используются различные режимы адресации.

Прямая адресация. При прямой адресации операнд в инструкции специфицирован 8-битным адресом. Только данные внутреннего ОЗУ (RAM) или регистров специальных функций (SFR) могут адресоваться прямо.

Косвенная адресация. При косвенной адресации инструкция специфицирует регистр, который содержит адрес операнда. Косвенно адресовано может быть как внутреннее, так и внешнее ОЗУ. Регистром для 8-битных адресов может быть или указатель стека, или R0 или R1 выбранного банка регистров. Регистром для 16-битных адресов данных может быть только 16-битный регистр указателя данных DPTR.

Регистровая адресация. Банки регистров, содержащие регистры R0 - R7, могут быть доступны инструкциям, чьи коды операций имеют 3 бита для спецификации регистра. Инструкции, имеющие доступ к регистрам делают код более эффективным, так как этот режим заменяет адресный байт. Когда выполняется инструкция, доступен один из восьми регистров в выбранном банке, который выбирается из четырех возможных двумя битами в регистре PSW.

Инструкции со специфицированными регистрами. Некоторые инструкции специфицированы определенными регистрами. Например, когда оперируют с аккумулятором, не нужен какой-либо байт для адресации данных. В этих случаях код операции указывает на нужный регистр. Инструкции, которые ссылаются на аккумулятор как A или ACC, определяются как код операции, специфицированный аккумулятором.

Непосредственная адресация. Код операции может содержать в качестве операнда константу.

Индексная адресация. Память программ может быть доступна только через индексную адресацию. Этот режим адресации предназначен для табличного обращения к памяти программ. 16-битный базовый регистр (либо DPTR, либо PC) указывает на базу таблицы, а аккумулятор задает точку входа в нее. Входной адрес в память программ формируется добавлением данных аккумулятора к базовому адресу.

Другой тип индексной адресации используется в инструкциях передачи управления. В этом случае адрес назначения передачи управления вычисляется как сумма базового указателя и содержимого аккумулятора.

3.5. Набор инструкций

Все типы микроконтроллеров семейства MCS51 выполняют один и тот же минимальный набор инструкций. Этот набор оптимизирован для 8-битных приложений, он обеспечивает разнообразные быстрые режимы адресации к резидентной (внутренней) памяти данных и упрощенные байтовые операции с небольшими структурами данных. Набор инструкций поддерживает однобитовые переменные как отдельный тип данных, позволяя разнообразные битовые манипуляции в управляющих и логических системах, требующих булеву обработку.

Следующий обзор набора инструкций дает краткое описание использования некоторых из них. Количество машинных циклов выполнения инструкций в таблицах указано для совместимого режима.

3.5.1. Арифметические инструкции

Арифметические инструкции приведены в табл. 3.2. В таблице указаны режимы адресации, которые могут использоваться с любой инструкцией, заменяя операнд <byte>. Например, инструкция ADD A, <byte> может быть записана в следующем виде:

```
ADD A, 7FH      (прямая адресация)
ADD A,@R0       (косвенная адресация)
ADD A,R7        (регистровая адресация)
ADD A,#127      (непосредственная адресация)
```

Следует отметить, что любой байт резидентной памяти данных может быть инкрементирован или декрементирован без использования аккумулятора. Инструкция INC DPTR оперирует с 16-битным указателем данных. Указатель данных генерирует 16-битные адреса для внешней памяти, таким образом, возможность инкрементирования в 16-битной операции является полезной особенностью.

Инструкция MUL AB умножает данные в аккумуляторе на данные в регистре B и помещает результат в регистре B и аккумуляторе.

Инструкция DIV AB делит данные в аккумуляторе на данные в регистре B и размещает 8-битную целую часть результата в аккумуляторе, а остаток – в регистре B

Инструкция DA A предназначена для двоично-десятичных операций. В двоично-десятичной арифметике после инструкций ADD и ADDC всегда необходимо выполнять операцию DA A, чтобы результат был так же в двоично-десятичном коде. Заметим, что DA A не конвертирует двоичное число

в двоично-десятичное число. Операция DA A дает значащий результат только как следующий шаг в дополнение операции с двумя двоично-десятичными байтами.

Табл. 3.2. Арифметические инструкции

Мнемокод	Операция	Режим адресации				Число
		Прям.	Косв.	Рег.	Непоср.	циклов
ADD A,<byte>	A = A +< byte>	X	X	X	X	1
ADDC A,<byte>	A = A+< byte>+C	X	X	X	X	1
SUBB A,<byte>	A = A–<byte>–C	X	X	X	X	1
INC A	A = A+1	Только аккумулятор				1
INC <byte>	<byte> = <byte>+1	X	X	X		1
INC DPTR	DPTR = DPTR+1	Только указатель данных				2
DEC A	A = A–1	Только аккумулятор				1
DEC <byte>	<byte> = <byte>–1	X	X	X		1
MUL AB	B:A = B x A	Только ACC и B				4
DIV AB	A = Int [A/B] B = Mod [A/B]	Только ACC и B				4
DA A	Десятичная коррекция	Только аккумулятор				1

3.5.2. Логические инструкции

Логические инструкции приведены в табл.3.3. Инструкции булевых операций (AND, OR, исключающее OR, NOT), выполняют их побитно. Таким образом, если аккумулятор содержит 00110101B, а <byte> содержит 01010011B, то после выполнения операции ANL A, <byte> в аккумуляторе будет находиться результат 00010001B.

Табл. 3.3. Логические инструкции

Мнемокод	Операция	Режим адресации				Число
		Прям.	Косв.	Рег.	Непоср.	циклов
ANL A,<byte>	A = A . AND. <byte>	X	X	X	X	1
ANL <byte>,A	<byte> = <byte>.AND.A	X				1
ANL <byte>,#data	<byte> = <byte>.AND.#data	X				2
ORL A,<byte>	A = A .OR.< byte>	X	X	X	X	1
ORL <byte ,A	<byte> = <byte>.OR. A	X				1
ORL <byte>,#data	<byte> = <byte>.OR.#data	X				2
XRL A,<byte>	A = A .XOR.< byte>	X	X	X	X	1
XRL <byte>,A	<byte> = <byte>.XOR.A	X				1

XRL <byte>,#data	<byte> = <byte>.XOR.#data	X				2
CLR A	A = 00H	Только аккумулятор				1
CPL A	A = .NOT. A	Только аккумулятор				1
RL A	Циклический сдвиг ACC влево на 1 бит	Только аккумулятор				1
RLC A	Циклический сдвиг ACC влево через перенос	Только аккумулятор				1
RR A	Циклический сдвиг ACC вправо на 1 бит	Только аккумулятор				1
RRC A	Циклический сдвиг ACC вправо через перенос	Только аккумулятор				1
SWAP A	Обмен нибблов в ACC	Только аккумулятор				1

В табл.3.3 также даны режимы адресаций, которые дают доступ к операнду <byte>. Таким образом, инструкция ANL A, <byte> может иметь следующие формы:

```
ANL A,7FH     (прямая адресация)
ANL A,@R1     (косвенная адресация)
ANL A,R6      (регистровая адресация)
ANL A,#53H    (непосредственная адресация.)
```

Заметим, что булевы операции могут выполняться с любым байтом в резидентной памяти данных или в пространстве SFR без аккумулятора, используя прямую адресацию.

Инструкция XRL <byte>, #data, например, дает возможность быстро и просто инвертировать биты порта, как в следующем примере:

```
XRL P1,#FFH
```

Циклические инструкции сдвигают содержимое аккумулятора на один бит. Для левой ротации старший бит сдвигается в младший бит. Для правой ротации младший бит – в старший.

Инструкция SWAP A обменивает старший и младший нибблы в аккумуляторе.

3.5.3. Инструкции перемещения данных

Для резидентной памяти

В табл. 3.4 приведены инструкции и соответствующие режимы адресаций, используемые для перемещения данных в резидентной памяти. Инструкция MOV <dest>,<src> позволяет передачу между любыми двумя ячейками, размещенными в резидентной RAM или SFR, без использования аккумулятора.

Табл. 3.4. Инструкции перемещения данных в резидентной памяти

Мнемокод	Операция	Режим адресации				Число циклов
		Прям.	Косв.	Рег.	Непоср.	
MOV A,<src>	A = <src>	X	X	X	X	1
MOV <dest>,A	<dest> = A	X	X	X		1
MOV <dest>,<src>	<dest> = <src>	X	X	X	X	2
MOV DPTR,#data16	DPTR = 16-битная константа				X	2
PUSH <src>	INC SP: MOV @SP,<src>	X				2
POP <dest>	MOV <dest>,@SP; DEC SP	X				2
XCH A,<byte>	Обмен данными между ACC и <byte>	X	X	X		1
XCHD A,@Ri	Обмен младшими ниблами между ACC и @Ri		X			1

Заметим, что в микроконтроллере стек размещен в резидентной памяти и «растет» вверх. Инструкция PUSH прежде инкрементирует указатель стека SP, а затем копирует байт в стек. Инструкции PUSH и POP используют только прямую адресацию для идентификации байта, который должен быть сохранен или восстановлен, но сам стек доступен посредством косвенной адресацией с использованием регистра SP. Это означает, что стек может превысить 128 байт, но не в пространство SFR, так как оно адресуется только прямо.

Инструкции передачи данных включают 16-битную инструкцию MOV, которая может инициализировать указатель данных DPTR для табличного обращения к памяти программ или для доступа к внешней памяти данных. Инструкция XCH A, <byte> обменивает данные между аккумулятором и адресованным байтом. Инструкция XCHD A, @Ri подобна предыдущей, но только обмениваются младшие ниблы.

Для внешней памяти данных

Табл.3.5 содержит инструкции перемещения данных, которые осуществляют доступ к внешней памяти данных. В этом случае может исполь-

зоваться только косвенная адресация. Могут использоваться либо для однобайтной адресации @Ri, где Ri может быть R0 или R1 из выбранного банка регистров, либо для двухбайтной адресации @DPTR. Неудобство использования 16-битной адресации, когда подключено несколько Кбайт внешней памяти, в том, что 16-битные адреса используют все 8 бит порта 2, как у адресной шины.

Заметим, что во всех обращениях к внешней памяти данных аккумулятор используется либо как цель, либо как источник. Стробы чтения и записи для внешней памяти активируются только в течение выполнения инструкции MOVX. Нормально, эти сигналы не активны и, если инструкция не выполняется, то не используются вовсе, и их выводы доступны как внешние линии ввода-вывода.

Табл. 3.5. Инструкции перемещения данных, осуществляющие доступ к внешней памяти данных

Ширина адреса	Мнемоника	Выполняемая операция	Число циклов
8 бит	MOVX A,@Ri	Чтение из внешней RAM @ Ri	2
8 бит	MOVX @Ri,A	Запись во внешнюю RAM @ Ri	2
16 бит	MOVX A,@DPTR	Чтение из внешней RAM @ DPTR	2
16 бит	MOVX @DPTR,A	Запись во внешнюю RAM @ DPTR	2

3.5.4. Таблицы обращения к памяти

Табл. 3.6 показывает две инструкции, которые пригодны для чтения таблиц доступа к памяти программ. Так как эти инструкции осуществляют доступ к памяти программ, таблицы могут только читаться и не могут обновляться. Мнемоникой их является MOVC (move constant). Если обращаются к таблице во внешней памяти программ, то генерируется строб чтения PSEN.

Первая инструкция MOVC в табл. 3.6 может работать с таблицей, имеющей до 256 входов с номерами от 0 до 255. Номер используемого входа загружается в аккумулятор, а указатель данных задает точку, указывающую на начало таблицы. Поэтому следующая инструкция копирует содержимое из нужной ячейки таблицы в аккумулятор.

```
MOVC A,@A+DPTR
```

Другая инструкция MOVC работает подобным образом, используя в этом случае в качестве базы таблицы программный счетчик PC, а через таблицу обращаются к подпрограммам. Вначале в аккумулятор загружается номер нужного входа, а затем вызывается подпрограмма.

```
MOV A,ENTRY_NUMBER
CALL TABLE
```

Табл. 3.6. Инструкции чтения таблиц обращения к памяти

Мнемоника	Выполняемая операция	Число циклов
MOVC A, @A + DPTR	Чтение памяти программ из (A + DPTR)	2
MOVC A, @A + PC	Чтение памяти программ из (A + PC)	2

Подпрограмма TABLE должна выглядеть, как в следующем примере:

```
TABLE: MOVC A,@A+PC
RET
```

Сама таблица непосредственно следует после инструкции RET (return) в памяти программ. Этот тип таблицы может иметь до 255 входов, пронумерованных от 1 до 255. Номер 0 может не использоваться, так как в это время выполняется инструкция MOVC и PC содержит адрес инструкции RET. Входом, имеющий номер 0 будет сам код операции RET.

3.5.5. Булевы инструкции

Микроконтроллеры семейства MCS51 имеют булевый (побитный) процессор. Резидентная память данных RAM содержит 128 адресуемых бит в пространстве адресов 20H – 2FH, и пространство регистров специальных функций SFR также поддерживает битовую адресацию. Все линии порта тоже побитно адресуемые, и каждая из них может быть использована как отдельный однобитный порт. Инструкции, обращающиеся к битам - не только условные переходы, но и инструкции перемещения, установки, очистки, инвертирования, дизъюнкции и конъюнкции. Эти битовые операции не так просто получаются в других архитектурах с байт ориентированным программным обеспечением.

Набор инструкций булевого процессора показан в табл. 3.7. Все битовые обращения имеют прямую адресацию. Адреса бит от 00H до 7FH находятся в младшем 128 битном пространстве, а с 80H до FFH – в пространстве SFR.

Табл. 3.7. Булевы инструкции

Мнемоника	Выполняемая операция	Число циклов
ANL C,bit	C = C .AND. bit	2
ANL C,/bit	C = C .AND..NOT. bit	2

ORL C,bit	C = C .OR. bit	2
ORL C,/bit	C = C .OR..NOT. bit	2
MOV C,bit	C = bit	1
MOV bit,C	bit = C	2
CLR C	C = 0	1
CLR bit	bit = 0	1
SETB C	C = 1	1
SETB bit	bit = 1	1
CPL C	C = .NOT. C	1
CPL bit	bit = .NOT. bit	1
JC rel	Переход, если C = 1	2
JNC rel	Переход, если C = 0	2
JB bit, rel	Переход, если bit = 1	2
JNB bit, rel	Переход, если bit = 0	2
JBC bit, rel	Переход, если bit = 1; CLR bit	2

Следующий пример показывает, как легко внутренний флаг может быть передан в вывод порта.

```
MOV C,FLAG
MOV P1.0,C
```

В этом примере FLAG является именем любого адресуемого бита в 128 битном пространстве или SFR. Линия ввода-вывода (младший бит порта 1 в данном случае) устанавливается или очищается в зависимости от значения бита флага, который может быть или 1, или 0.

Бит переноса в регистре PSW используется в качестве однобитного аккумулятора булевого процессора.

Имеются инструкции из серии бит тестируемых, которые выполняют переход, если адресуемый бит установлен (JC, JB,), или если адресуемый бит не установлен (JNC, JNB). Другая тестируемая инструкция JBC выполняет переход, если бит установлен, после чего бит очищается. Таким образом, флаг может быть протестирован и очищен в одной операции.

3.5.6. Относительное смещение

Адрес назначения для этих переходов специфицируется для ассемблера либо меткой, либо конкретным адресом в памяти программ. Однако при ассемблировании он заменяется байтом относительного смещения. Если выполняется переход, то этот байт со знаком (в арифметическом дополнении) прибавляется к содержимому программного счетчика PC. Диапазон перехода находится в пределах от -128 до +127 байт памяти программ относительно первого байта, следующего после инструкции.

3.5.7. Инструкции перехода

В табл. 3.8. показаны инструкции безусловных переходов. В таблице имеется одна инструкция JMP addr, но фактически их три - SJMP, LJMP и AJMP, которые различаются форматом адреса назначения. JMP является основной мнемоникой, которой может воспользоваться программист, когда он не заботится о том, каким образом адрес перехода будет закодирован. Инструкция SJMP кодирует адрес назначения с относительным смещением, как сказано выше. Инструкция имеет размер в 2 байта, включая код операции и байт относительного смещения. Инструкция LJMP кодирует адрес назначения как 16-битную константу. Инструкция имеет размер в 3 байта, включая код операции и двухбайтный адрес. Адрес назначения может быть в любом месте 64К пространства памяти программ. Инструкция AJMP кодирует адрес назначения как 11-битную константу. Инструкция имеет размер в 2 байта, включая код операции, который содержит 3 бита 11-битного адреса, и следующий байт, содержащий младшие 8 бит адреса назначения. Когда выполняется инструкция, эти 11 бит просто заменяют младшие 11 бит регистра PC. Старшие 5 бит остаются теми же самыми. Таким образом, адрес назначения должен быть в некотором блоке размером 2К, следующем после инструкции AJMP.

Во всех случаях программист специфицирует для ассемблера адрес назначения в виде метки или 16-битной константы. Ассемблер переводит адрес назначения в нужный формат для данной инструкции.

Инструкция JMP @A+DPTR поддерживает вычисляемые переходы. Адрес назначения вычисляется во время выполнения инструкции как сумма 16-битного регистра DPTR и аккумулятора. Обычно DPTR загружается адресом таблицы переходов, а аккумулятор является индексом в таблице. Например, если требуется 5 вариантов перехода, то в аккумулятор загружаются целые числа от 0 до 4. Программа может иметь следующий вид:

```
MOV DPTR,#JUMP_TABLE
MOV A,INDEX_NUMBER
RL A
JMP @A+DPTR
```

Инструкция RL A конвертирует номер индекса (0 – 4) в четное число в диапазоне от 0 до 8, так как каждый вход в таблицу переходов разделен 2 байтами, как показано в следующем примере:

```
JUMP_TABLE:
AJMP CASE_0
AJMP CASE_1
```

```
AJMP CASE_2
AJMP CASE_3
AJMP CASE_4
```

Табл. 3.8. Безусловные переходы

Мнемоника	Выполняемая операция	Число циклов
JMP addr	Переход к addr	2
JMP @A+DPTR	Переход к A + DPTR	2
CALL addr	Вызов подпрограммы addr	2
RET	Возврат из подпрограммы	2
RETI	Возврат из прерывания	2

Табл. 3.8. показывает только одну инструкцию CALL addr, но имеются еще 2 инструкции LCALL и ACALL, которые отличаются способом определения адреса подпрограммы. Основная мнемоника CALL используется программистом, если его не заботит, каким образом кодируется адрес. Инструкция LCALL использует 16-битный формат адреса, и подпрограмма может находиться в любом месте 64К пространстве памяти программ. Инструкция ACALL использует 11-битный формат и подпрограмма должна быть в 2К блоке, следующем после инструкции ACALL. В любом случае программист специфицирует адрес подпрограммы для ассемблера или как метку, или как 16-битную константу. Ассемблер сформирует адрес в правильном формате для данной инструкции.

Подпрограммы должны заканчиваться инструкцией RET, которая возвращает выполнение программы к инструкции, следующей после CALL. Инструкция RETI используется для возврата из подпрограммы обслуживания прерывания. Отличие между RET и RETI лишь в том, что RETI сообщает системе управления прерыванием об окончании обслуживания прерывания. Если прерывания нет во время выполнения RETI, то тогда RETI функционально идентично RET.

В табл. 3.9 приведен список условных переходов. Все они осуществляют переход на определенный адрес назначения через относительное смещение, и таким образом, ограничены областью от –128 до +127

байт, следующей после инструкции условного перехода. Однако, пользователь специфицирует ассемблеру нужный адрес назначения тем же образом, что и для других переходов: меткой или 16-битной константой.

В PSW бит Zero отсутствует. Для определения этого условия инструкции JZ и JNZ тестируют аккумулятор. Инструкция DJNZ (Decrement and Jump if Not Zero – декремент и переход, если не ноль) является управляющей для циклов.

Табл. 3.9. Условные переходы.

Мнемоника	Операция	Режим адресации				Число циклов
		Прям.	Косв.	Рег.	Непоср.	
JZ rel	Переход, если A = 0	Только аккумулятор				2
JNZ rel	переход, если A ≠ 0	Только аккумулятор				2
DJNZ <byte>,rel	Декремент и переход, если не 0	X		X		2
CJNE A,<byte>,rel	Переход, если A ≠ <byte>	X			X	2
CJNE <byte>,#data,rel	Переход, если <byte> ≠ #data		X	X		2

Чтобы выполнить цикл N раз, надо загрузить счетчик числом N и закончить цикл DJNZ, передающим управление в начало цикла, как показано ниже для N = 10.

```
MOV COUNTER,#10
LOOP: (начало цикла)
*
*
*(конец цикла)
DJNZ COUNTER,LOOP
(продолжение)
```

Инструкция CJNE (Compare and Jump if Not Equal – сравнение и переход, если не ноль) может так же использоваться для управления циклом, как показано на рис.13. Два байта специфицируются в поле операндов инструкции. Переход выполняется только, если два байта не равны. В примере ниже два байта были данными в регистре R1 и константа 2AH. Начальное значение содержимого R1 было 2EH. Каждый раз при выполнении цикла R1 декрементируется, и цикл продолжается до тех пор, пока оно не станет равным 2AH.

```
MOV R1,#2EH
LOOP:
DEC R1
CJNE R1,#2AH,LOOP
(продолжение)
```

Другое применение этой инструкции для сравнений «больше, чем», «меньше, чем». Два байта в поле операндов берутся как беззнаковые целые числа. Если первое меньше второго, тогда бит переноса Carry устанавливается в 1. Если первый больше второго, или равен, то бит переноса очищается.

3.6. Выполнение инструкций

Микроконтроллеры AT89LP51/52 используют расширенный процессор 8051, который работает со скоростью больше стандартного 8051 в 6 – 12 раз. Увеличение производительности связано с двумя факторами. Во-первых, процессор выбирает однобайтовую инструкцию из памяти программ каждый тактовый период. Во-вторых, процессор использует простой двухступенчатый конвейер для загрузки и выполнения инструкции параллельно. Эта основная конвейерная концепция позволяет процессору получить до 1 MIPS (миллион инструкций в секунду) на 1 МГц. AT89LP51/52 также имеют режим совместимости, который сохраняет 12-тактный машинный цикл стандартных МК 8051 таких, как AT89S51/52.

В быстром режиме однобайтовая инструкция извлекается за каждый системный такт. Набор инструкций 8051 включает в себя инструкции переменной длины от 1 до 5 байт. В системе с выборкой байта за один такт означает, что на выполнение каждой инструкции, по меньшей мере, требуется столько же тактов. Большинство команд в AT89LP51/52 следует этому правилу: с несколькими исключениями время выполнения инструкции в системных тактовых циклах равно числу байтов в инструкциях. Переходам и вызовам подпрограмм требуется дополнительный цикл для вычисления целевого адреса, а также ряд других сложных инструкций требуют нескольких циклов.

Примеры выполнения инструкций в быстром режиме показаны на рис. 4.1. Обратите внимание, что инструкции в быстром режиме выполняются в три раза дольше, если они выбираются из внешней памяти программ.

В режиме совместимости байтовые инструкции выбираются за каждые три системных такта, а процессор работает с машинным циклом с 6 состояниями, полученными делением на 2 машинного цикла, состоящего из

12 тактовых периодов системного генератора. Стандартные инструкции выполняются за 1, 2 или 4 машинных цикла.

Режим совместимости можно использовать для сохранения профилей выполнения унаследованных приложений. Примеры инструкций в режиме совместимости показаны на рис. 3.5.

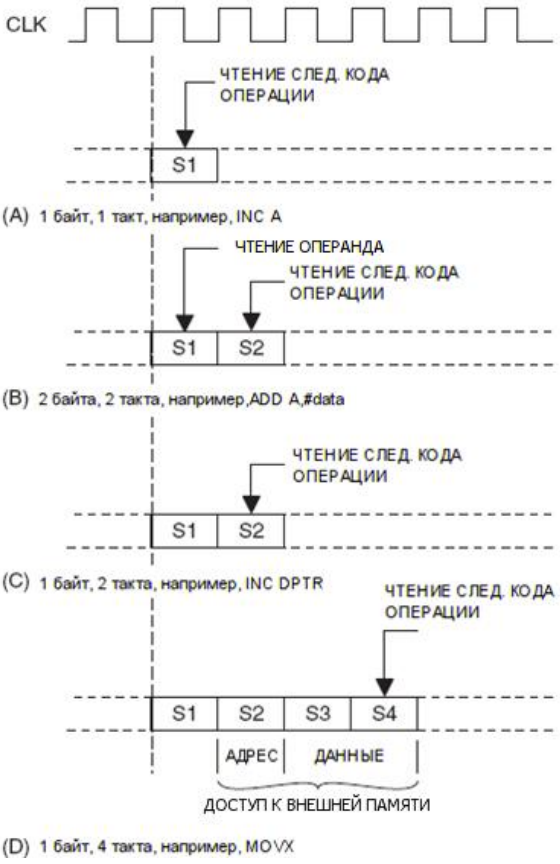


Рис. 3.4. Выполнение инструкций в быстром режиме

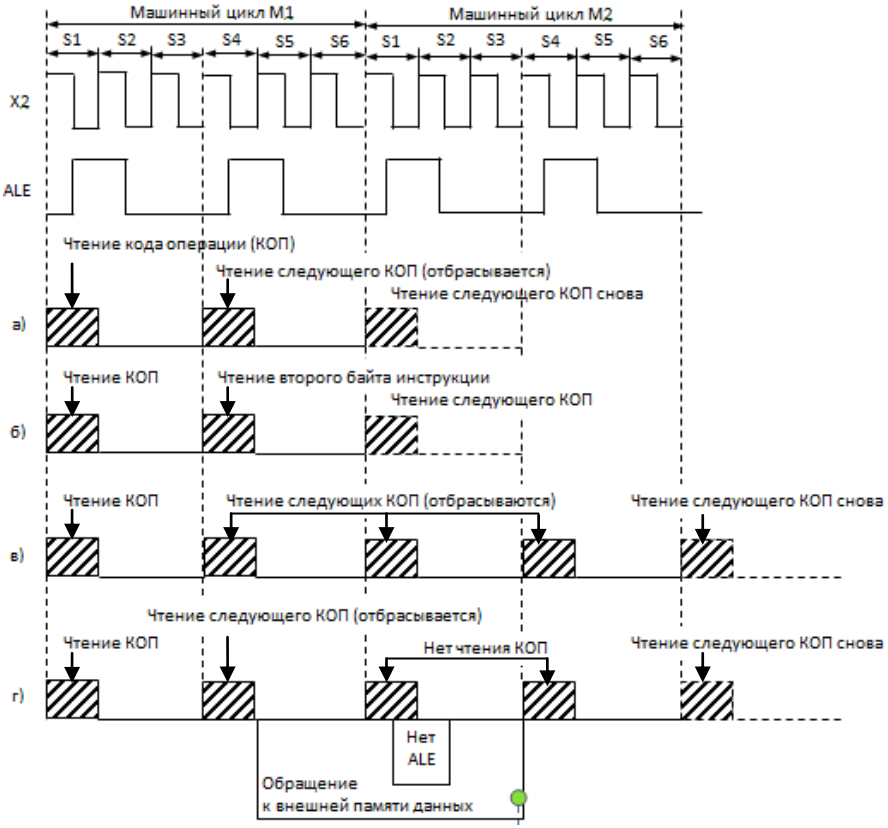


Рис.3.5. Последовательности выборки и выполнения инструкций в совместимом режиме

а) - Инструкция 1 байт/1 цикл, например, INC A; б) - Инструкция 2 байта/1 цикл, например, ADD A,#d; в) - Инструкция 1 байт/2 цикла, например, INC DPTR; г) - Инструкция 1 байт/2 цикла, например, MOVX A,@DPTR

3.7. Режимы работы микропроцессора

AT89LP51/52 поддерживает два разных режима снижения энергопотребления: Idle и Power-down. Доступ к этим режимам осуществляется через регистр управления мощности PCON.

Режим холостого хода устанавливается соответствующим битом IDL в регистре PCON. При этом продолжает работать внутренний генератор синхросигналов, все регистры сохраняют свое значение, на выводах портов удерживается уровень напряжения, установленный до холостого режима. Обращений к внешним устройствам памяти нет. Выход из режима холостого хода происходит по сигналу сброса RST или по прерыванию. После завершения программы обслуживания будет исполнена команда, которая следует за командой установления режима.

Способ установки режима пониженного энергопотребления (спящего) тот же: устанавливается бит PD в регистре PCON, в результате которого останавливается генератор синхросигналов, содержимое регистров сохраняется, а на выводах портов удерживаются значения, соответствующие содержимому их буферных регистров. При этом основное питание может быть выключено, переходя на резервное. Выход из режима пониженного энергопотребления возможен только по сигналу внешнего сброса RST, сбросу при включении питания или определенными разрешенными прерываниями.

4. ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

4.1. Классификация и структура запоминающих устройств

Наибольший удельный вес по числу микросхем и суммарной их стоимости в современных МПС имеет система памяти. Наряду с резидентной памятью, к которой процессор обращается непосредственно по системной шине, используется внешняя, обладающая большим объемом память на магнитных, магнитно-оптических и твердотельных носителях информации. В настоящем разделе рассматривается только резидентная память.

Основной элементной базой резидентной системы памяти являются БИС запоминающих устройств (ЗУ). По способу занесения информации они делятся на оперативные/Random Access Memore (ОЗУ/RAM) и постоянные/Read Only Memory (ПЗУ/ ROM), по режиму работы - на статические и динамические, по технологии изготовления - на биполярные и униполярные (МОП). Классификация БИС ЗУ приведена на рис. 4.1

.

ОЗУ статического типа получили наибольшее применение в специализированных системах, так как они обладают большим быстродействием и высокой надежностью. Системы памяти универсальных устройств строятся, как правило, с использованием БИС ОЗУ динамического типа, обладающих большим объемом памяти.

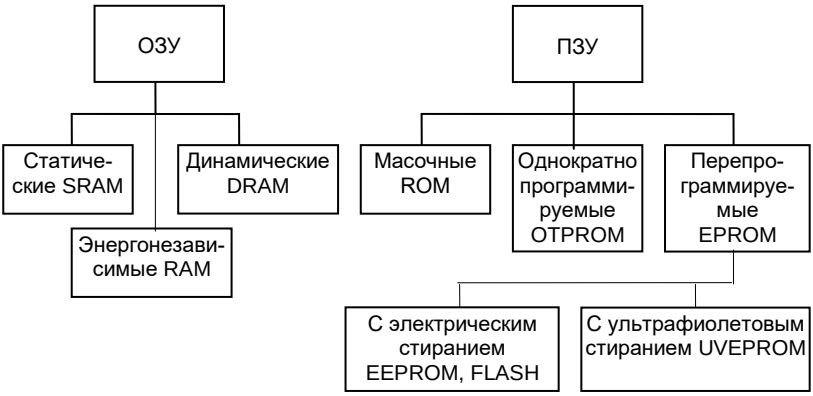


Рис. 4.1. Классификация БИС запоминающих устройств

Основными характеристиками ЗУ являются емкость и быстродействие. ЗУ содержит некоторое число ячеек N, в которых может храниться код с определенным числом разрядов n. Ячейки памяти последовательно нумеруются двоичными числами, номер ячейки называется адресом. Количество информации, которое может храниться в ЗУ, определяет его емкость. Емкость может выражаться либо числом ячеек N с указанием разрядности n хранимой в них информации: N x n, либо произведением в битах, например, 512К x 8 или 4М.

Быстродействие ЗУ характеризуется двумя параметрами: временем выборки t_f, представляющим собой интервал времени между моментом подачи сигнала выборки и появлением считанных данных на выходе; и циклом записи t_w, определяемым минимально допустимым временем между подачей сигнала выборки при записи и моментом, когда возможно следующее обращение к памяти.

БИС РПЗУ дополнительно характеризуются временем хранения записанной в них информации и допустимым количеством циклов перезаписи.

На рис. 4.2 изображена типичная структура запоминающего устройства. Информация хранится в накопителе, который представляет собой матрицу, составленную из элементов памяти (ЭП), расположенных по

строкам и столбцам. Элемент памяти может хранить бит информации (0 либо 1). Кроме того, ЗУ имеет устройство управления, которое обеспечивает его работу в режимах чтения, записи и хранения данных.

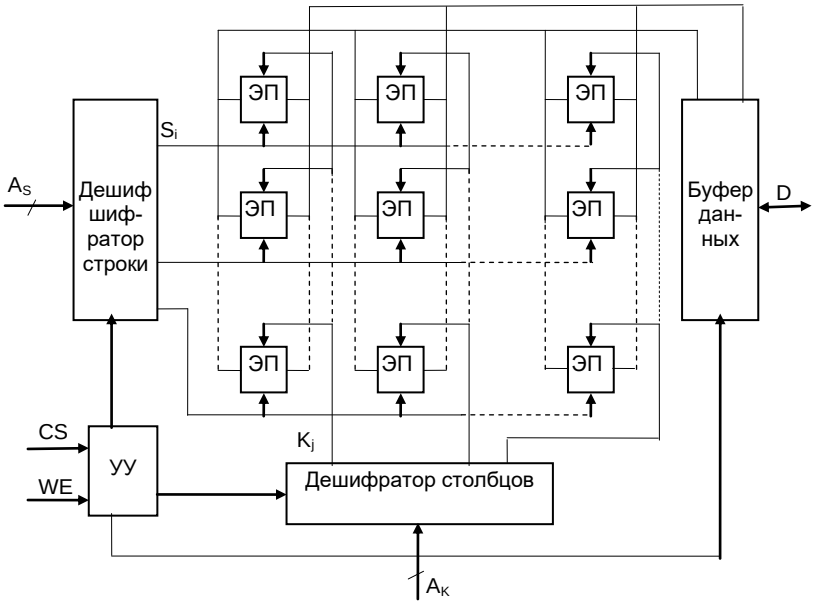


Рис. 4.2. Структура БИС запоминающего устройства

Разряды кода адреса делятся на две группы: одна группа из s разрядов определяет двоичный номер строки, в которой расположен ЭП, другая группа из k разрядов - номер столбца. Каждая группа разрядов адреса подается на соответствующий дешифратор, который вырабатывает управляющий сигнал (S_i, K_i), выбирающий адресованный ЭП.

При чтении, когда на БИС памяти подается сигнал выбора кристалла CS, содержимое ЭП выдается из буфера данных. При записи, кроме того, в устройство управления УУ посылается сигнал разрешения записи WE, и в результате, бит информации из буфера данных поступает в ЭП. При отсутствии сигнала CS буфер данных переходит в высокоимпедансное состояние.

4.2. Оперативные запоминающие устройства

Статическое ОЗУ. В качестве элементов памяти используются триггеры с непосредственными связями (см. рис. 4.3.а), которые могут неограниченно долго хранить информацию при включенном питании. Они очень просты в эксплуатации, обладают высокой надежностью и не требуют сложных схем обслуживания. При интегральной реализации памяти используются два вида запоминающих матриц: накопители высокого быстродействия со средней степенью интеграции без схем дешифрации (время цикла может достигать 3 нс) и накопители с повышенной информационной емкостью со схемами дешифрации**.**

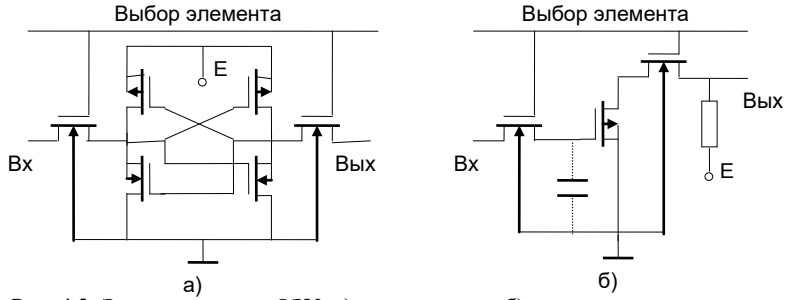


Рис. 4.3. Элемент памяти ОЗУ: а) статического, б) динамического типа

Временные диаграммы обмена данными с действующими временными ограничениями изображены на рис. 4.4. Из рисунка видно, что для статических ОЗУ длительность цикла записи равна времени выборки.

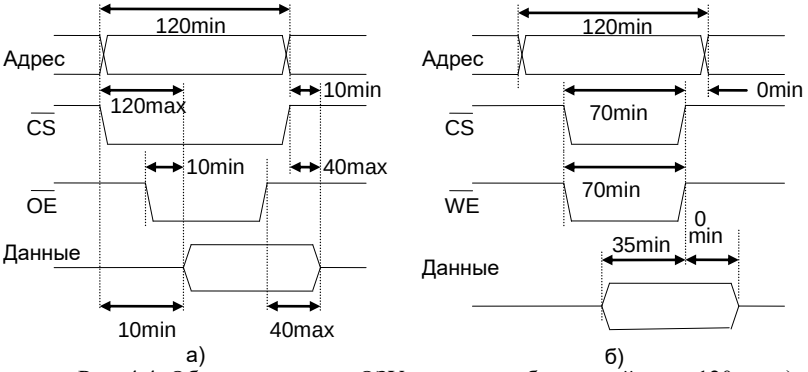


Рис. 4.4. Обмен данными с ОЗУ, имеющим быстродействие 120 нс: а) цикл чтения, б) цикл записи

В цикле записи на рисунке присутствует сигнал OE’ низкого уровня, который открывает буфер данных, а сигнал CS’ низкого уровня фиксирует, в данном случае, адрес элемента памяти.

В качестве примера можно привести быстродействующий БИС ЗУ (FAST SRAM) типа CY7C1041G: емкостью 4М, с быстродействием 10 нс, питанием 4,5-5,5 В, потреблением в активном состоянии 38 мА, в холостом - 6 мА, имеет 44 выводов.

Динамические ОЗУ. Увеличение емкости памяти связано с сокращением площади, занимаемой элементом памяти. Это достигается при использовании динамических ячеек, в которых информация хранится в виде заряда в конденсаторе. Элемент динамической памяти на МОП структурах показан на рис. 4.2б. Он представляет собой один МОП-транзистор, в котором роль конденсатора выполняет емкость затвор-подложка. Ток утечки накопителя составляет не более 10^{-10} А, а емкость - 0,1 пФ, следовательно, постоянная разряда конденсатора $t_{RC} > 1$ мс. Поэтому для сохранения заряда необходимо периодическое восстановление информации (регенерация) с периодом $t_r < 2$ мс.

Главные отличия динамических БИС ОЗУ от статических заключается в следующем:

- необходимо устройство, обеспечивающее регенерацию памяти в моменты, когда к ней нет обращений;
- обращение к БИС требует дополнительных схем;
- максимальная простота накопителя для обеспечения минимально занимаемой площади;
- меньшая потребляемая мощность, так как динамический элемент не потребляет тока, когда к нему нет обращения.

Сравнивая два элемента памяти на рис. 4.3, можно считать, что емкость БИС статических и динамических ОЗУ находится в пропорции 1:4. Следовательно, если емкость динамической БИС составляет 256 Мбит, то может потребоваться 28 выводов на ее корпусе только для адресации ячейки памяти. Поэтому, для этих микросхем при обращении к ним применяется мультиплексирование адреса, которое также позволяет за меньшее время выполнять регенерацию памяти.

На рис. 4.5 показан нормальный цикл обмена данными с динамическим ОЗУ.

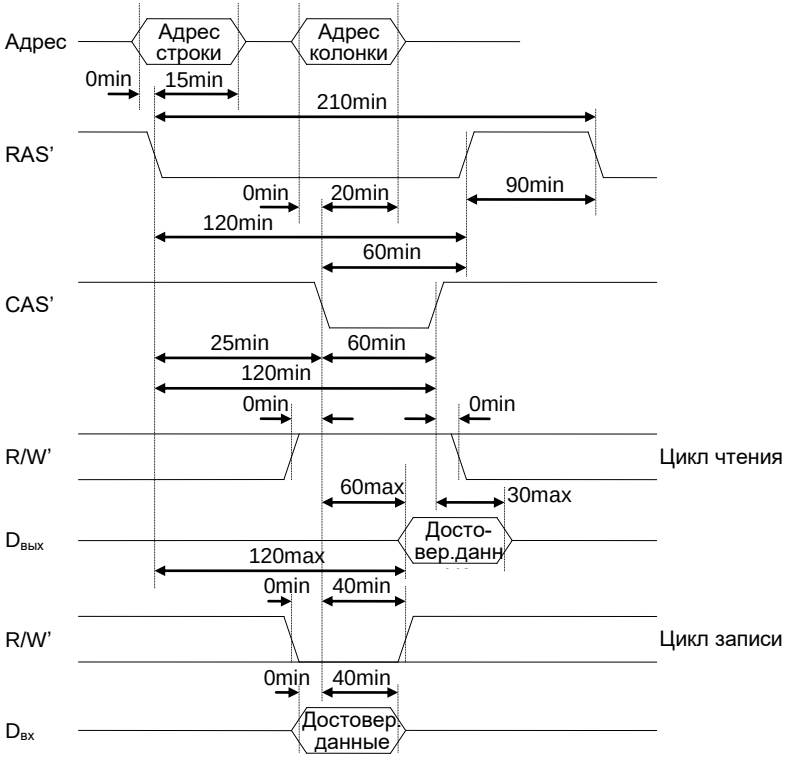


Рис. 4.5. Циклы чтения и записи динамического ОЗУ (Motorola, 120 нс)

Адрес, содержащий, например, для ОЗУ в 1 Мбайт 20 бит, разделяется на две группы и мультиплексируется на вдвое меньшее число выводов, сначала адрес строки, стробируемый сигналом RAS' (Row Address Strobe), а затем адрес колонки, стробируемый сигналом CAS' (Column Address Strobe). Данные записываются или читаются в соответствии с состоянием входа направления R/W' вслед за установкой CAS'. Перед началом следующего цикла памяти должно пройти некоторое время "выдержки RAS", поэтому длительность цикла записи больше времени доступа. Например, динамическое ОЗУ может иметь время доступа 120 нс и длительность цикла 210 нс. Цикл регенерации выглядит так же, но без сигнала CAS. Регенерация памяти выполняется при обращении к строкам, адреса которых вырабатывается отдельным счетчиком в устройстве регенерации, там же вырабатываются и управляющие сигналы.

Для управления динамическим ОЗУ существуют специальные микросхемы, например AM2968A, которые берут на себя не только мультиплексирование адресов и образование сигналов RAS/CAS, но также и арбитраж регенерации вместе с образованием адресов строк, кроме того, они включают в себя мощные усилители и согласующие резисторы, которые нужны для подключения больших матриц микросхем памяти.

Синхронная динамическая память

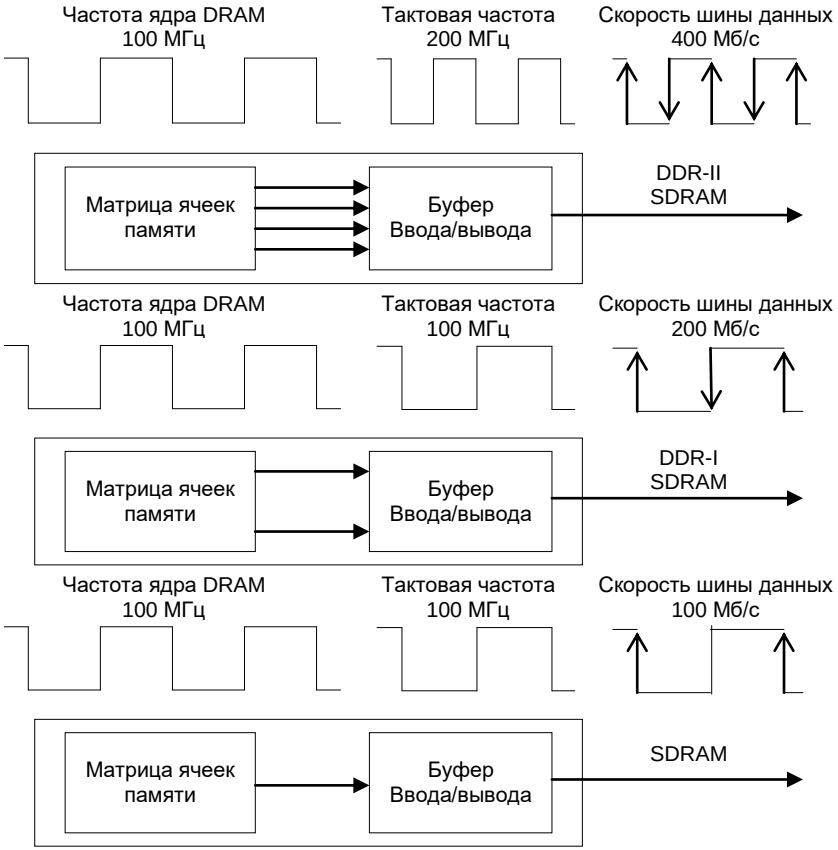


Рис. 4.6. Пропускная способность SDRAM

Для современных МПС требуется оперативная память большого объема с высокой скоростью обмена с процессором. Поэтому, в основном,

здесь применяются синхронные динамические устройства памяти SDRAM (Synchronous DRAM) разного типа DDR-I (Double Data Rate), DDR-II, DDR-III. На рис. 4.6 показано за счет чего достигается высокая скорость обмена современных модулей SDRAM.

Модули DDR-I обеспечивают удвоенную скорость обмена данными, DDR-II – учетверенную, а DDR-III в 8 раз быстрее за счет удвоения скорости работы выходных буферов.

4.3. Постоянные запоминающие устройства

ПЗУ относятся к неразрушаемой памяти, которая необходима в МПС для постоянного хранения информации, в том числе и при отключении питания. В ней может быть находиться программа начальной загрузки и инициализации системы, программа функционирования специализированного микропроцессорного контроллера, или же только данные, например, таблицы поправок для линеаризации функции отклика измерительной системы. Основными требованиями к ПЗУ являются не разрушаемое считывание, высокая надежность и энергонезависимость хранения информации.

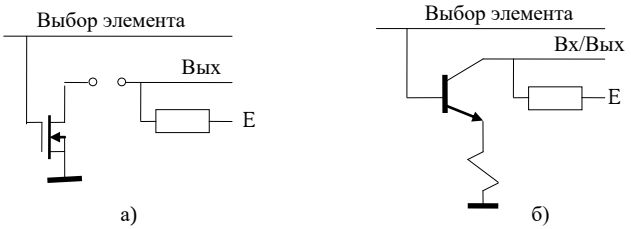


Рис. 4.6. Элементы памяти ПЗУ: а) масочного типа; б) с плавкими перемычками

Масочные ПЗУ. Масочно-программируемые ПЗУ относятся к категории заказных микросхем. В них информация записывается на стадии изготовления БИС при металлизации межсоединений в соответствии с выданной спецификацией, по которой создается маска (фотошаблон). Такие ПЗУ выгодны только при изготовлении больших партий для массового производства микропроцессорных устройств, например, калькуляторов, часов и тому подобное. В качестве элементов памяти в них используются диоды, биполярные и полевые транзисторы. Наибольшее распространение получили ПЗУ на МОП-транзисторах (см. рис. 4.6а) из-за технологической простоты изготовления и связанной с этим возможностью получения высокой степени интеграции, а также малой потребляемой мощности. Преимуществом ПЗУ, выполненных на биполярных транзисторах, является их более высокое быстродействие.

Однократно программируемые ПЗУ. К ним относятся ПЗУ с плавкими перемычками, элемент памяти которых изображен на рис. 4.6б. При выпуске все биты ПЗУ установлены и для сброса требуемых бит требуется подвергнуть элементы памяти действию электрического тока. Наибольшее распространение получили перемычки в виде плавких вставок из нихрома и поликремния, которые можно избирательно пережигать с помощью внешнего источника тока. Для программирования таких ПЗУ требуются специальные устройства - программаторы. Программируемые ПЗУ выпускаются на базе биполярных и КМОП структурах.

В настоящее время в качестве однократно программируемых ПЗУ (OTPROM - One Time Programming ROM) используются дешевые варианты РПЗУ с ультрафиолетовым стиранием без предназначенных для облучения кварцевых окон в корпусе. Производители гарантирует сохранность информации в них в течение не менее 10 лет.

Репрограммируемые ПЗУ (РПЗУ). На этапе реализации репрограммируемых БИС первыми были с ультрафиолетовым стиранием (UVEPROM). Они были созданы на основе лавинно-инжекционных МОП-транзисторов с плавающим затвором (см. рис. 4.7а), в которых запись информации осуществляется электрическим способом, а для стирания информации требуется ультрафиолетовое облучение.

Элемент памяти РПЗУ изображен на рис. 4.7. Правый транзистор служит для выбора элемента, а левый, структура которого показана на рис. 4.7а, - для хранения информации. Особенность его в том, что он имеет изолированный со всех сторон затвор, который можно зарядить с помощью лавинной инжекции (процесса пробоя слоя, изолирующего затвор, при напряжении свыше 12 В). Отрицательный заряд (около 10^6 электронов) в затворе, который можно рассматривать как конденсатор с постоянной времени порядка столетий, создает в подложке проводящий канал, тем самым переводит транзистор в состояние логической 1. Поскольку затвор электрически недоступен, стереть заряд можно лишь облучив его интенсивным потоком излучения с длиной волны менее 400 нм в течение 10 - 30 мин. При взаимодействии квантов с электронами в затворе энергия квантов передается электронам, которые, преодолевая потенциальный барьер, уходят в подложку кристалла. Вследствие этого отдельные байта РПЗУ выборочно стереть нельзя. Количество циклов перезаписи у РПЗУ ограничено.

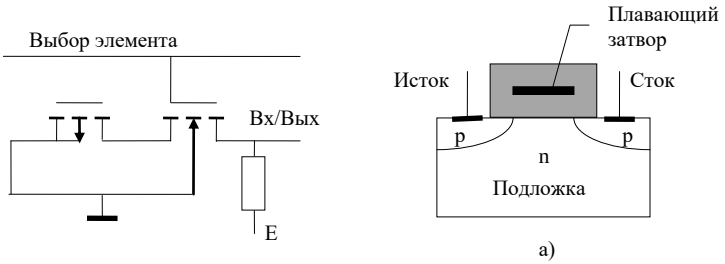


Рис. 4.7. Элемент памяти РПЗУ с ультрафиолетовым стиранием: а) МОП-транзистор с плавающим затвором

Наиболее распространены микросхемы с емкостью 512 Кбит. Например, БИС M27C512-12 имеет организацию памяти 64Кх8, быстродействие 120 нс, напряжение питания 5 В, потребление в активном состоянии 30 мА, в холостом - 200 мкА, имеет 28 выводов. Чтобы запрограммировать РПЗУ, к нему надо приложить повышенное напряжение (обычно 12,5 или 21 В) и увеличить цикл записи до нескольких миллисекунд. Используя специальные алгоритмы записи, время записи значительно сокращается. Так, программирование всей микросхемы с помощью алгоритма PRESTO II занимает меньше 6 с.

Электрически стираемые РПЗУ (ЭРПЗУ, EEPROM), ячейки в которых могут быть выборочно стерты и запрограммированы непосредственно в устройстве, где они используются как память. Они необходимы в том случае, когда требуется хранить и корректировать данные, поступающие в процессе работы МПС, например, при хранении констант конфигурации системы или параметров калибровки. При изготовлении ЭРПЗУ так же, как и для РПЗУ, используется технология МОП с плавающим затвором.

Современные ЭРПЗУ не требуют внешнего повышенного напряжения питания и работают практически так же, как и статическое ОЗУ - можно перезаписать любой байт с помощью одного цикла записи. В БИС предусмотрены внутренние схемы для генерации повышенного напряжения при программировании, а также увеличения длительности цикла записи до нескольких миллисекунд.

В качестве примера рассмотрим ЭРПЗУ типа AT28C256. БИС имеет организацию памяти 32Кх8, время доступа 150 нс, единственное напряжение питания 5 В, низкое электропотребление в активном состоянии - 80 мА, в холостом - 200 мкА, циклов записи до 100000, сохранность данных в

течении 100 лет. Корпус имеет 28 выводов, совместимых с промышленным стандартом ОЗУ.

БИС поддерживает 64-байтную страничную операцию записи, обеспечивая цикл записи в 78 мкс/байт, и тем самым запись всей микросхемы меньше, чем за 2,5 с. Для ранней индикации окончания цикла записи используется опрос инвертированных данных или специального статусного бита. Дополнительно к аппаратной защите данных по выбору пользователя добавляется и программная защита данных.

Страничная запись позволяет записать последовательно от 2 до 64 байт до начала внутреннего автоматического программирующего цикла, если каждый цикл записи байта будет начинаться не позже 100 мкс после предыдущего цикла. Для определения конца цикла записи можно опрашивать на шине данных старший бит, который до завершения операции имеет инвертированное значение, или следующий после него, который не будет менять свое значение при последовательном его опросе.

Аппаратная защита заключается в автоматическом запрещении записи в том случае, если длительность сигнала разрешения записи меньше, чем 20 нс, или если напряжение питания не больше, чем 3,5 В, или если во время включения или выключения питания любой из сигналов находится в состоянии: разрешения чтения OE’ - в низком, разрешения записи WE’ и выбора кристалла CE’ - в высоком.

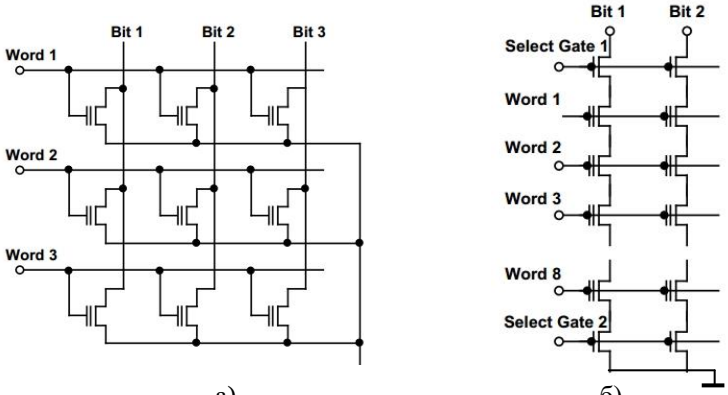
Программная защита записи данных устанавливается после применения специального алгоритма последовательной записи кодов в ячейки с определенными адресами. Другая последовательность кодов снимает программную защиту записи.

Моментальная память (FLASH) - разновидность РПЗУ с электрическим стиранием, которая изначально была разработана как альтернатива дисковой памяти для портативных компьютеров. В отличие от ЭРПЗУ стирание информации в них осуществляется блоками, а не отдельными байтами.

Флэш-память типа NOR (рис. 4.8а), например, БИС 28F008 фирмы Intel, имеют однотранзисторную ячейку с двухслойным поликремниевым затвором и блочную структуру накопителя - 16 блоков по 64К байт каждый с организацией памяти 1Мх8 бит. Они выдерживают не менее 100000 циклов записи/стирания. Запись данных в память производится побайтно за 9 мкс, при чтении максимальное время доступа составляет 85 нс. Стирание каждого блока выполняется независимо от остальных блоков за 1,6 с.

Автоматика записи байта и стирания блока позволяет записывать байт и стирать блок, используя командную двухтактную последовательность, посылаемую из командного интерфейса пользователя. Внутренний автомат записи автоматически выполняет алгоритм записи байта или стирание блока с последующей проверкой результата, освобождая от этих операций ресурсы процессора.

Другой тип флэш-памяти NAND (рис. 4.8б) отличается тем, что и запись, и чтение осуществляется блоками, и поэтому, он не годится для записи в них программ.



а) б)

Рис. 4.8. Структура NOR-flash (а); NAND-flash (б)

Емкость флэш-памяти зависит от технологии изготовления ячейки с разделением инжектированного заряда в затворе на несколько уровней. Изначально были одноуровневые SLC (Single-level cell), потом появились MLC (Multi-level cell) с 2 битами на ячейку, TLC (Triple-level cell) с 3 битами на ячейку, QLC (Quad-level cell) с 4 битами на ячейку и наконец, разрабатываемая в настоящее время PLC (Penta-level cell) с 5 битами на ячейку. Все они имеют пониженное количество циклов стирания/записи по сравнению с SLC.

Неудобство в потребности для ЭРПЗУ второго источника питания для выполнения операций записи и стирания частично снимается наличием большого количества микросхем преобразователей напряжения. Например, фирма Maxim выпускает ИС MAX732 с входным напряжением 4 - 7,5 В и выходным 12 В, позволяющую подключать до 9 внешних компонентов.

Среди других вариантов РПЗУ следует отметить последовательные электрически перепрограммируемые КМОП ЗУ (SEEPROM) фирмы Micro-

chip. Отличительной их особенностью является последовательный доступ к данным. Они имеют одно напряжение питания, аппаратную защиту записи, 2-х проводной последовательный интерфейс I^2C, внутреннее тактирование циклов записи и стирания, страничный буфер, 1 млн. циклов записи/чтения, время сохранения данных - не менее 40 лет. Микросхемы выпускаются емкостью от 1К бит до 64К бит с байтовой организацией шины данных в 8- и 14-выводных корпусах.

Фирма Xicor выпускает электрически стираемый потенциометр (EEPOT). В этой микросхеме имеется 99 последовательно соединенных, равных по величине резисторов, а положение отвода от них задается программно, сохраняя это положение в энергонезависимой памяти, входящей в эту же микросхему.

Среди прочих применений РПЗУ с электрическим стиранием их удобно использовать при разработке встроенного программного обеспечения, так как они могут быть так же, как и другие РПЗУ, запрограммированы в программаторе, но при этом им не требуется много времени на стирание в процессе отладки программ.

Энергонезависимые ОЗУ. Иногда существует потребность в энергонезависимой памяти с доступом к данным в темпе, характерным для ОЗУ, когда требуется неограниченное число циклов чтения/записи.

Данную проблему можно решить, если применить статическое КМОП ОЗУ с резервным батарейным питанием. Для этой цели выпускаются микросхемы, например, фирмой Maxim MAX703, которые предназначены в качестве супервизора микропроцессора, где помимо других полезных функций она выполняет переключение ОЗУ на питание от литиевой батарейки, если основное питание снижается до 4,65 В.

Некоторые фирмы, например, Dallas Semiconductor выпускают энергонезависимые ОЗУ, размещая в обычном корпусе вместе с КМОП ОЗУ литиевую батарейку, которая автоматически подключается при снижении напряжения питания. Однако при такой “энергонезависимости” данные исчезают при выходе из строя батарейки, срок службы которой не превышает 10 лет.

Другая возможность - в использовании энергонезависимого ОЗУ NOVRAM (NOnVolatile RAM) фирмы Xicor, в котором на одном кристалле объединены статическое ОЗУ и ЭРПЗУ. Входной сигнал STORE’ переносит содержимое ОЗУ в ЭРПЗУ при полной длительности цикла записи 10 мс, извлекаются данные быстрее, чем за 1 мкс. Для выработки управляю-

щего сигнала можно использовать одну из микросхем фирмы Maxim MAX690, предназначенную для контроля питающих напряжений. Объявлено, что NOVRAM выдерживают 10000 операций сохранения и неограниченное число операций записи/чтения в ОЗУ.

4.4. Организация памяти в микроконтроллере AT89LP51/52

Микроконтроллеры AT89LP51/52 используют гарвардскую архитектуру с раздельными адресными пространствами для памяти программ и памяти данных. Память программ имеет регулярное линейное адресное пространство с поддержкой 64K байт прямой адресации кода приложений. Память данных имеет 256 байт внутреннего ОЗУ, включая 128 байт регистров специального назначения пространства ввода/вывода. AT89LP51/52 поддерживают до 64K байт внешней памяти данных с частью внешней памяти данных, расположенной на чипе энергонезависимой флэш-памяти данных. Внешняя память программ поддерживается для адресов выше 4K/8K. Адресные пространства памяти AT89LP51/52 приведены в табл. 4.1.

Табл.4.1. Адресные пространства памяти AT89LP51/52

Название	Описание	Диапазон
DATA	Прямо адресуемое внутреннее ОЗУ	00H–7FH
IDATA	Непрямо адресуемое внутреннее ОЗУ и стековое пространство	00H–FFH
SFR	Прямо адресуемое пространство регистров ввода/вывода	80H–FFH
FDATA	Внутренняя энергонезависимая флэш-память данных	0000H–00FFH
XDATA	Внешняя память данных	0100H–FFFFH
CODE	Внутренняя энергонезависимая флэш-память программ	0000H–0FFFH (AT89LP51) 0000H–1FFFH (AT89LP52)
XCODE	Внешняя память программ	1000H–FFFFH (AT89LP51) 2000H–FFFFH (AT89LP52)
SIG	Внутренняя энергонезависимая флэш-память для сигнатуры	0000H–01FFH

Примечание. Массив сигнатур пользователя предназначен для идентификационных кодов пользователя или данных постоянных параметров.

4.4.1. Память программ

AT89LP51/52 содержит 4K/8K байт программируемой на кристалле флэш-памяти для хранения программ, а также поддержку до 60K/56K байт внешней памяти программ. Флэш-память может выдержать не менее 10000 циклов записи/стирания и имеет минимальное время хранения данных в течение 10 лет. Векторы прерываний расположены в пределах первых 83 байт памяти программ. Таблицы констант могут быть расположены во всем 64K адресном пространстве памяти программ для доступа с инструкцией MOVC. Карта памяти программ AT89LP51/52 показана на рис. 4.9.

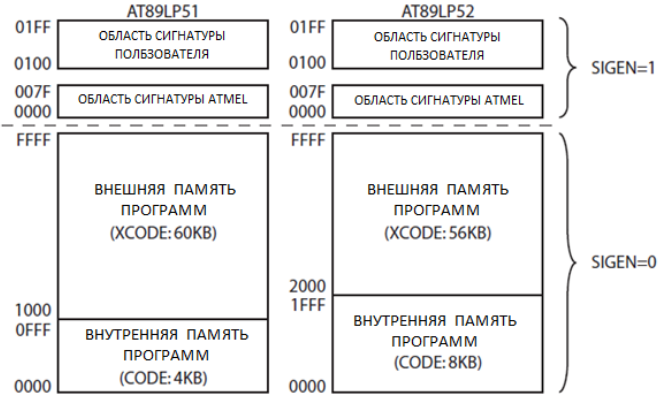


Рис. 4.9. Карта памяти программ AT89LP51/52

4.4.2. Интерфейс памяти программ

Микроконтроллеры AT89LP51/52 используют стандартный интерфейс внешней памяти программ 8051 со старшим адресом в порте P2, младшим адресом и данными в мультиплексированном порте P0 и стробами ALE и PSEN. Адреса памяти программ всегда 16-битовые, даже если фактическое количество памяти, используемой программой, может быть меньше 64 Кбайт. При выполнении внешней программы используются два полных 8-битовых порта P0 и P2 для адресации памяти программ.

Рис. 4.10 показывает аппаратную конфигурацию для доступа к 64 Кбайт внешнего ПЗУ с использованием 16-битового линейного адреса. Порт P0 служит для мультиплексирования шины адреса/данных. Строб разрешения фиксации адреса (ALE) используется для фиксации младшего байта адреса во внешнем регистре LATCH, после чего порт P0 может быть освобожден для ввода/вывода данных. Порт P2 обеспечивает старший байта адреса в течение всей операции. PSEN стробирует внешнюю память.

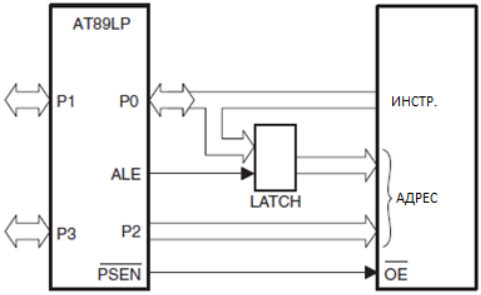


Рис. 4.10. Доступ к внешней памяти программ

На рис. 4.11 показан тайминг интерфейса внешней памяти программ. Сигнал ALE выводится с постоянной скоростью из расчета 1/3 системной частоты с длительностью 1/3 рабочего цикла. Сигнал PSEN выводится с одинаковой скоростью с коэффициентом заполнения 50%. Новый адрес устанавливается в середине импульса ALE для фиксации в LATCH по заднему фронту и переходит в третье состояние по заднему фронту PSEN. Данные инструкции вводятся в P0 и внутренне защелкиваются в высокой фазе тактового импульса до нарастающего фронта PSEN. Эти тайминги относятся как к режиму совместимости, так и быстрому режиму. В режиме совместимости нет никакой разницы в таймингах внутреннего и внешнего выполнения инструкций.

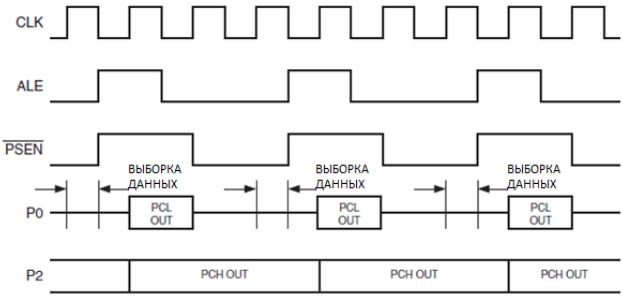


Рис. 4.11. Тайминг интерфейса внешней памяти

4.4.3. Внутренняя память данных

Микроконтроллеры AT89LP51/52, содержащие 256 байт основной памяти SRAM данных, включая 128 байт памяти ввода/вывода, отображаются в одном 8-битовом адресном пространстве. Доступ к внутренней памяти данных не требует настройки. Внутренняя память данных имеет, как показано на рис. 4.12, три адресных пространства: DATA, IDATA и SFR. Некоторая часть внешней памяти данных реализована как внутренняя.

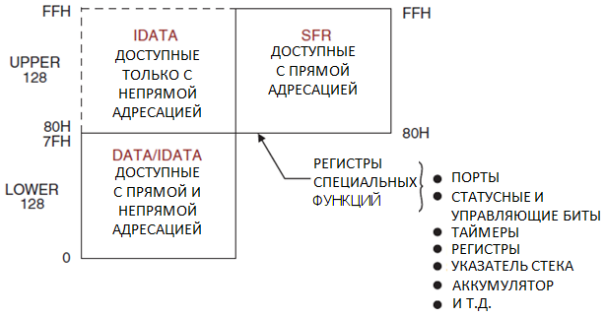


Рис.4.12. Карта внутренней памяти данных

DATA: Первые 128 байт ОЗУ, адресуемых 8-битовым адресом (00H-7FH), включены в инструкции. Самые младшие 32 байта памяти данных сгруппированы в 4 банка из 8 регистров каждый. Биты RS0 и RS1 (PSW.3 и PSW.4) выбирают, какой банк регистров используется. Инструкции с использованием регистровой адресацией будут доступны только к указанному в настоящее время банку. Также адресуются 128 бит (00H-7FH) в младших адресах данных 20H-2FH.

IDATA: Все 256 байт внутреннего ОЗУ могут быть косвенно адресованы с использованием 8-разрядных указателей R0 и R1. Первые 128 байт IDATA включены в пространство DATA. Аппаратный стек также расположен в пространстве IDATA.

SFR: Старшие 128 прямых адресов (80H-FFH) предназначены для доступа к регистрам ввода/вывода. Регистры ввода/вывода на устройствах AT89LP называются регистрами специального назначения (SFR). Регистры SFR доступны только через прямую адресацию.

Табл. 4.2. Карта памяти SFR и значения, устанавливаемые при сбросе

	8	9	A	B	C	D	E	F	
0F8H									0FFH
0F0H	B 0000 0000								0F7H
0E8H									0EFH
0E0H	ACC 0000 0000								0E7H
0D8H									0DFH
0D0H	PSW 0000 0000								0D7H
0C8H	T2CON 0000 0000	T2MOD 0000 0000	RCAP2L 0000 000	RCAP2H 0000 0000	TL2 0000 000	TH2 0000 0000			0CFH
0C0H	P4 1111 1111	PMOD (2)							0C7H
0B8H	IP xx00 0000	SADEN 0000 0000							0BFH
0B0H	P3 1111 1111							IPH xx00 0000	0B7H
0A8H	IE 0x00 0000	SADDR 0000 0000							0AFH
0A0H	P2 1111 1111		AUXR1 0000 00x0				WDTRST (write-only)	WDTCON 0000 0xx0	0A7H
98H	SCON 0000 0000	SBUF xxxx xxxx							9FH
90H	P1 1111 1111	TCONB 000x xxxx					MEMCON 0000 00xx		97H
88H	TCON 0000 0000	TMOD 0000 0000	TL0 0000 0000	TL1 0000 0000	TH0 0000 0000	TH1 0000 0000	AUXR 0000 0000	CLKREG (3)	8FH
80H	P0 1111 1111	SP 0000 0111	DP0L 0000 0000	DP0H 0000 0000	DP1L 0000 0000	DP1H 0000 0000		PCON 000x 0000	87H
	0	1	2	3	4	5	6	7	

Примечание:
1. Все SFR в крайнем левом столбце имеют побитовую адресацию.
2. Значение сброса: 0101 0101B, когда включена функция Tristate-Port Fuse, и 0000 0011B, когда она отключена.
3. Значение сброса: 0101 0010B при включенном режиме совместимости и 0000 0000B при отключенном.

4.4.4. Внешняя память данных

Микроконтроллеры AT89LP поддерживают 16-битовое адресное пространство до 64 Кбайт внешней памяти данных (XDATA). Внешнее пространство памяти доступно с инструкциями MOVX. Некоторые внутренние ресурсы памяти данных отображаются как часть внешнего адресного пространства, как показано на рис. 4.13. Эти ячейки памяти могут потребовать настройки, перед тем как процессор получит к ним доступ после установки

бит EXRAM, DMEN, IAP в регистрах SFR. В AT89LP51/52 реализовано 256 байт энергонезависимой флэш-памяти данных (FDATA).

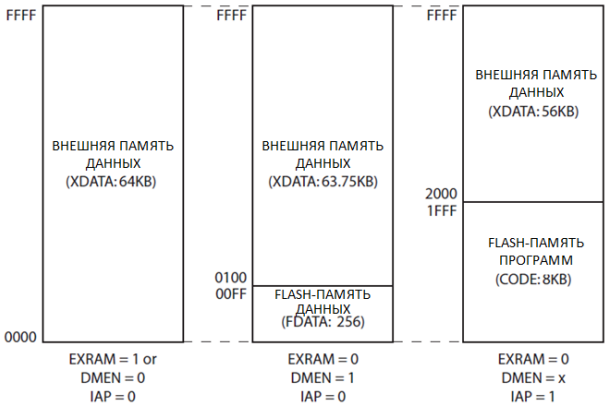


Рис. 4.13. Карта внешней памяти данных

XDATA: Пространство внешней памяти данных может вместить до 64 Кбайт. AT89LP51/52 используют стандартный интерфейс 8051 внешней памяти данных со старшим байтом адреса в порте P2, младшим байтом адреса и данными, мультиплексированными портом P0 и стробами ALE, RD и WR. XDATA могут быть доступно, как с 16-битовым (MOVX @DPTR), так и 8-битовым (MOVX @Ri) адресом.

Некоторое внутреннее пространство памяти данных отображается как часть адресного пространства XDATA. В этом случае нижний диапазон адресов будет доступен как ресурс внутренней, а не внешней памяти. Адреса выше этого диапазона по умолчанию будут относиться к XDATA. AT89LP51/52 поддерживают до 63.75K или 56K байт внешней памяти при использовании внутреннего отображения.

FDATA: Флэш-память данных, являющаяся частью внешней памяти, реализована как внутренняя энергонезависимая память данных. Инструкции MOVX с адресом диапазона будут иметь доступ к внутренней энергонезависимой памяти. В FDATA можно попасть только с 16-битовыми (MOVX @DPTR) адресами. Указание диапазона адресов FDATA инструкциями MOVX @Ri дают доступ только к внешней памяти. Адреса выше диапазона FDATA отображаются на XDATA.

4.4.5. Интерфейс внешней памяти данных

AT89LP51/52 используют стандартный интерфейс внешней памяти данных 8051 со старшим адресом в порте P2, младшим адресом и данными, мультиплексированными в порте P0, и стробами ALE, RD и WR. Интерфейс может быть использован в двух различных конфигурациях в зависимости от того, какой тип инструкции MOVX используется для доступа к XDATA.

Рис. 4.14 показывает конфигурацию аппаратных средств для доступа к 64K внешнему ОЗУ с использованием 16-битового линейного адреса. Порт P0 служит мультиплексированной шиной адреса/данных в ОЗУ. Строб разрешения защелки адреса (ALE) используется для фиксации младшего байта адреса во внешнем регистре, поэтому порт P0 должен быть освобожден для ввода/вывода данных. Порт P2 обеспечивает старший байт адреса в течение всей операции. Инструкции MOVX @DPTR используют режим линейного адреса.

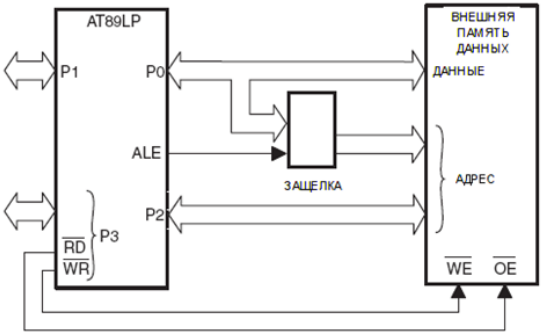


Рис.4.14. Режим 16-битного линейного адреса внешней памяти данных

На рис. 4.15 показана конфигурация аппаратных средств для доступа к 256-байтовым блокам внешнего ОЗУ с использованием 8-битового страничного адреса. Порт P0 служит мультиплексированной шиной адреса/данных для внешнего ОЗУ. Строб ALE используется для фиксации байта адреса во внешнем регистре, поэтому порт P0 может быть освобожден для ввода/вывода данных. Линии порта P2 (или другие порты), могут обеспечить управление линиями страницы памяти, однако эта операция не выполняется оборудованием автоматически. Программное приложение должно изменить регистр порта P2 в соответствующих случаях для доступа к различным страницам. Инструкции MOVX @Ri используют режим страничного адреса.

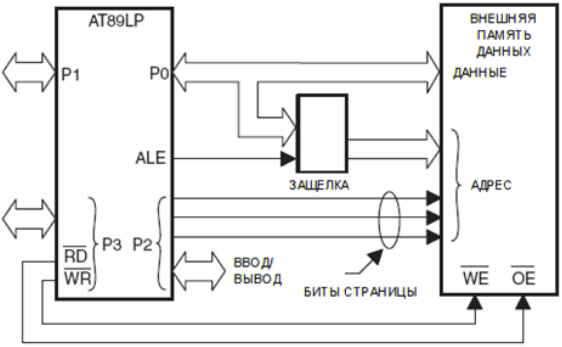


Рис.4.15. Режим 8-битного страничного адреса внешней памяти данных

На рис. 4.16. и рис. 4.17 показаны примеры циклов чтения и записи в быстром режиме, а на рис. 4.18 и рис. 4.19 - в режиме совместимости.

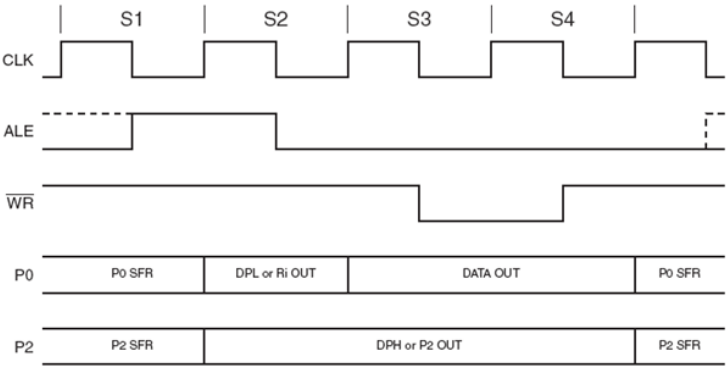


Рис.4.16. Цикл записи внешней памяти данных в быстром режиме

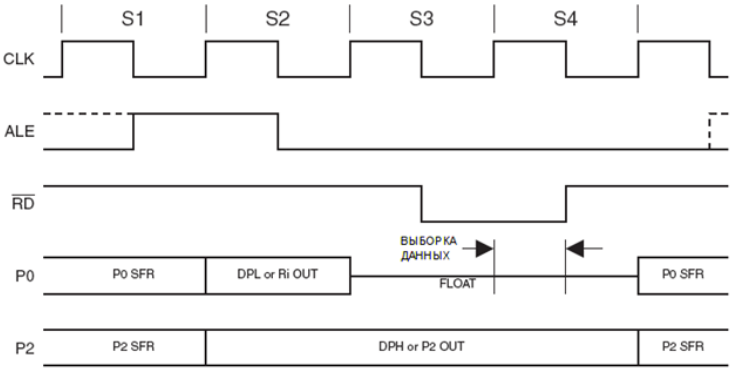


Рис.4.17. Цикл чтения внешней памяти данных в быстром режиме

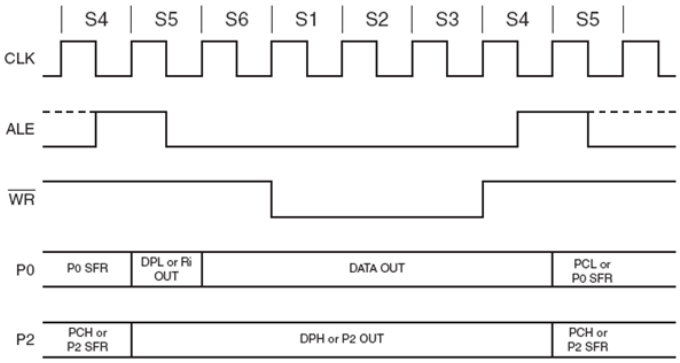


Рис.4.18. Цикл записи внешней памяти данных в режиме совместимости

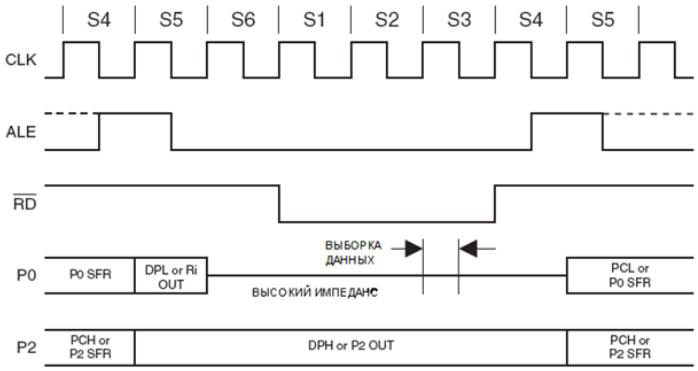


Рис.4.19. Цикл чтения внешней памяти данных в режиме совместимости

5. ИНТЕРФЕЙС ВВОДА-ВЫВОДА ДАННЫХ

5.1. Организация ввода-вывода данных

Микропроцессор осуществляет обмен данными с устройствами, которые располагаются на одной с ним плате. Пример такого обмена рассмотрен в предыдущем разделе. Другой вариант передачи с устройствами, подключаемые к нему через усилители посредством кабелей с разъемами. В этом случае должен существовать определенный стандарт на организацию каналов связи, обеспечивающих обмен данными с внешними устройствами.

Чаще всего обмен данными на небольшие расстояния, например, в пределах одной или нескольких плат МПС происходит по параллельным линиям связи, то есть каждый бит информации передается по отдельному каналу. Такой способ обмена обеспечивает наибольшее быстродействие, однако линии связи занимают на плате много места, а при передаче за пределы платы требуются дорогие многожильные кабели. В тех случаях, когда скорость обмена не велика, используются последовательные линии связи, по которым информационные биты передаются последовательно по одной линии в соответствии с тактовой частотой передачи. Как правило, они используются для удаленной связи, например, в информационных сетях, но могут применяться и на одной плате, как, например, последовательный двухпроводной канал связи I^2C.

Обмен данными может осуществляться в синхронном и асинхронном режимах. Синхронный режим передачи предполагает, что при каждом выполнении команд обмена устройство готово к приему или выдачи данных. Данные поступают на линии связи или считываются с них синхронно со строб сигналами, которые генерируются микропроцессором или другим активным устройством, например, контроллером прямого доступа к памяти. Однако в этом случае посылающее устройство не знает получены ли данные адресатом. Кроме того, при больших расстояниях сигналы могут устанавливаться недостаточно быстро, чтобы обеспечивать надежную передачу данных.

Асинхронный режим или режим передачи с квитированием обеспечивается дополнительными сигналами готовности и подтверждения передачи данных. Перед началом передачи данных микропроцессор проверяет сигнал готовности устройства к приему, а затем, другим сигналом оповещает устройство, что данные отправлены. При приеме данных МП посылает сигнал готовности к приему и опрашивает сигнал подтверждения от устройства, что данные переданы. Основное достоинство асинхронного

режима заключается в возможности использовать линии связи любой длины за счет лишь незначительного усложнения аппаратуры.

При организации связи с медленными устройствами в синхронном режиме передачи данных необходимо включение тактов ожидания кратное тактам стробирования. В асинхронном режиме увеличение цикла обмена происходит естественным образом.

Для организации системы ввода-вывода данных с внешними устройствами существует разнообразные стандартные правила или протоколы обмена, которые чаще называются стандартными интерфейсами ввода-вывода, включая в это понятие и аппаратные средства обеспечения ввода вывода. Интерфейсы внешних устройств разделяются на две большие категории: радиального (двухточечного) и магистрального (многоточечного) типа.

5.2. Интерфейсы радиальные параллельного типа

Рассмотрим реализацию протокола для обмена данными на примере стандартного интерфейса радиального параллельного (ИРПР) типа BS 4421, разработанного в Великобритании в 1969 г, Передача данных в ИРПР осуществляется между передающим устройством, называемым источником S (Source), и принимающим устройством - приемником A (Acceptance). На рис. 5.1 приведен пример использования интерфейса в системе связи контроллера с внешними устройствами, в составе которых имеются цифровой вольтметр с программным переключением режима работы (ЦВ), коммутатор измерительных каналов (К) и печатающее устройство (ПУ). Если контроллер одновременно выполняет функции приема и передачи, то требуется соответствующее количество сопряжений стандартного интерфейса источник-приемник.

Набор информационных, управляющих и вспомогательных сигналов стандартного интерфейса, а также направления их передачи изображены на рис. 5.2. Сигналы готовности источника SO (Source is On) и приемника AO (Acceptance is On) устанавливаются на все время связи устройств для индикации их рабочего состояния. Сигнал запроса данных приемником AC (Acceptance Control) и подтверждения истинности данных на линии источником SC (Source Control) предназначены для квитирования в процессе передачи данных по линиям D7...D0. Цикл работы системы повторяется до тех пор, пока не будет установлен сигнал ST (Source Terminate), свидетельствующий, что передано последнее слово в блоке данных, или пока приемник не распознает ошибку и не выдаст сигнал AE (Acceptance Error). Для надежности передачи данных сигнал PV (Parity Valid) включает выполне-

ние контроля четности, а передача сигнала по линии P приводит к нечетности суммы единичных сигналов в линиях данных. Линии S и Z выполняют функции общего провода и защитного заземления устройств.

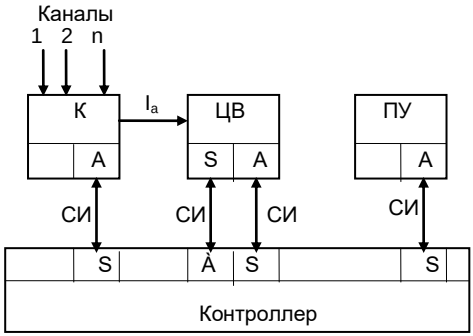


Рис. 5.1. Пример использования интерфейса ИРПР

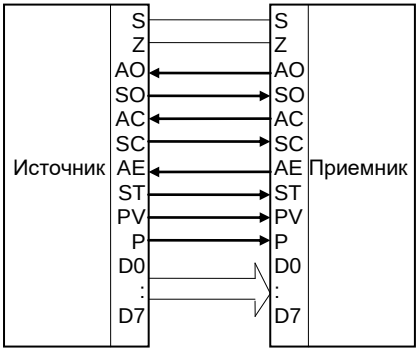


Рис. 5.2. Структура связей между источником и приемником сообщений

Временная диаграмма передачи данных (тайминг) изображена на рис. 5.3. Инициатором передачи данных является приемник, который в момент 1 информирует источник о своей готовности. Источник готовит данные и в момент 2 сообщает приемнику, что данные в линиях действительны. В мо-

мент 3 данные принимаются приемником, но к очередному обмену он еще не готов.

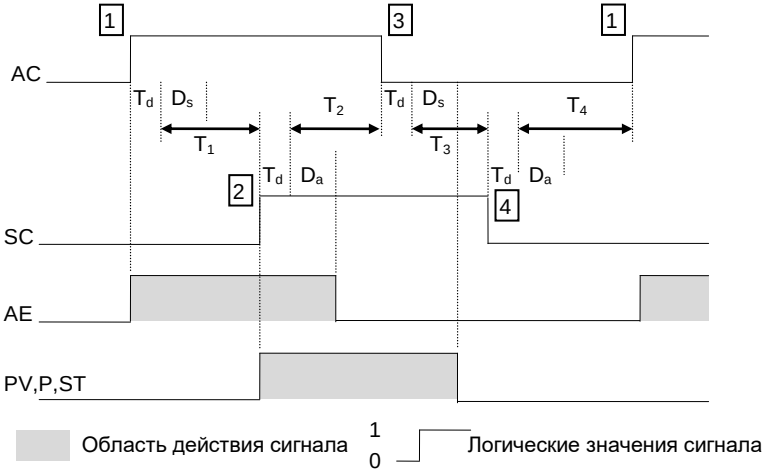


Рис. 5.3. Временная диаграмма передачи данных

С момента 4 сигналы кодов в линиях данных не действительны и затем может начаться следующий цикл передачи. Приемник распознает команду SC за время D_A, а источник команду AC за время D_S. Периоды T_1 и T_3 - время работы источника, T_2 и T_4 - приемника. Скорость передачи данных зависит от характеристик функциональных элементов системы и задержки прохождения сигналов в кабельных соединениях. Минимальное время передачи одного слова равно $T_1 + T_2 + T_3 + T_4 + 4T_d$. Время задержки кабеля T_d в стандарте указано для ограниченной длины кабеля до 1500 м.

5.3. Интерфейсы радиальные последовательного типа

Интерфейс обеспечивает асинхронную передачу постоянного тока по дуплексной линии. Допускается наличие цепи, определяющее состояние устройства ввода-вывода. Каждая цепь выполняется витой парой, и информация передается по токовой петле. Первая линия предназначена для передаваемых данных, вторая - для принимаемых данных и необязательная третья линия - для указания состояния приемника. В случае если устройство предназначено только для приема, первая линия разомкнута. Третья линия в состоянии логической 1 указывает готовность приемника принять новую информацию.

Последовательная передача данных осуществляется байтами по 5 - 8 бит со скоростью 9600 бит/с на расстояние до 500 м. Формат передаваемого кадра, изображенный на рис. 5.4, содержит помимо данных стартовый бит, необязательный бит контроля четности и один или два стоповых бита. Размещение данных в кадре начинается с младшего бита. Логические состояния линий кодируются значением тока в соответствии с табл. 5.1. В активном режиме линии запитываются со стороны передатчика, в пассивном - со стороны приемника.

Рис. 5.4. Формат кадра передаваемой информации

Табл. 5.1. Соответствие токов логическим состояниям

Логическое состояние	Токовая петля	
	20 мА	40 мА
0	0 - 3 мА	5 - 10 мА
1	15 - 25 мА	30 - 50 мА

Наиболее широко распространенным последовательным интерфейсом для связи между компьютером и внешними устройствами является стандарт RS232C, утвержденный ассоциацией электронной промышленности (EIA) в 1969 г. Для него предполагается наличие оборудования двух видов: оконечное оборудование данных DTE (Data Terminal Equipment) и аппаратура передачи данных DCE (Data Communication Equipment). Как правило, терминальным оборудованием является компьютер, а в качестве примера связного оборудования можно привести модем, который служит последовательным звеном в цепи компьютера и телефонной линией.

Табл. 5.2. Сигналы 9-контактного RS232C

Наименование	Номер контакта	Направление DTE↔DCE	Функция (со стороны DTE)
TxD	3	→	Передаваемые данные
RxD	2	←	Принимаемые данные
RTS	7	→	Запрос передачи (DTE готово послать данные)
CTS	8	←	Готов к передачи (DCE готово принять данные)
DTR	4	→	DTE готово
DSR	6	←	DCE готово
DCD	1	←	Детектор принимаемого сигнала
RI	9	←	Индикатор вызова
SG	5		Общий

Интерфейс RS232C включает в себя 25 линий, которые обеспечивают два независимых последовательных канала данных: главный и вспомогательный. Оба канала могут работать в дуплексном асинхронном режиме, т.е. одновременно принимать и передавать данные. Однако на практике вспомогательный канал применяется редко, и обычно используются только 9 линий, приведенных в табл. 5.2.

TxD и RxD - линии передачи и приема данных; RTS и CTS - пара квитирования передачи данных; DTR и DSR пара квитирования готовности оборудования; DCD и RI - разрешение входа DTE. На всех сигнальных линиях действуют биполярные уровни, причем установленные данные имеют отрицательную полярность (см. рис. 5.5): логический 0 (SPACE) имеет уровень напряжения от +3 до +25 В, а логическая 1 (MARK) - от -3 до -25 В, а установленные управляющие сигналы - положительную полярность того же уровня напряжения.

Сдвиг уровня, т.е. преобразование ТТЛ-уровней в уровни интерфейса RS232C и наоборот, производится специальными микросхемами драйвера линии и приемника линии, например, 1488 и 1489 соответственно или

75150 и 75154, аналогами которых являются отечественные микросхемы К170АП2 и К170ПУ2. Существенный недостаток этих микросхем в том, что им требуется двуполярное питание ±12 В. Фирма Maxim предлагает для этих целей семейство микросхем MAX220 - MAX249 с одним питающим напряжением +5 В, в которых вырабатываются необходимые уровни интерфейса до напряжения ±10 В.

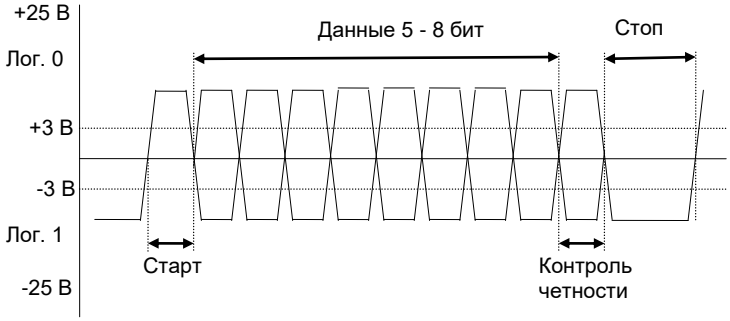


Рис. 5.5. Логические уровни передаваемого кадра данных в стандарте RS232C

Наличие двух типов оборудования для RS232C вносит некоторую сложность в разводке кабеля, осуществляющего связь между ними. На рис. 5.7 изображены некоторые варианты соединительных кабелей.

На рис. 5.6а показаны соединения для пары устройств DTE - DCE, когда оба устройства используют полный протокол квитирования. Одна пара квитирующих сигналов представлена линиями RTS, CTS, другая - линиями DTR, DSR. На рис. 5.6в показан кабель “нуль-модема” с перекрещенными входами и выходами для пары DTE - DTE. Тот же кабель годится и для пары DCE - DCE, только надо изменить направление стрелок и опустить соединение контакта DCD. Если одно устройство реализует протокол квитирования, а другое нет, то кабели распаиваются так, чтобы устройство само отвечало на свои же сигналы квитирования. Такое соединение для пары DTE - DCE показано на рис. 5.6б, и для пары DTE - DTE (или пары DCE - DCE, но тогда без соединения контакта DCD) - на рис. 5.6г.

Некоторые устройства используют аппаратные сигналы квитирования RTS, CTS для индикации начала и конца передачи данных, в то время как более медленные устройства (например, принтер) держит их установлен-

ными. Другие осуществляют программное квитирование: Ctrl-S (для остановки) и Ctrl-Q (для возобновления).

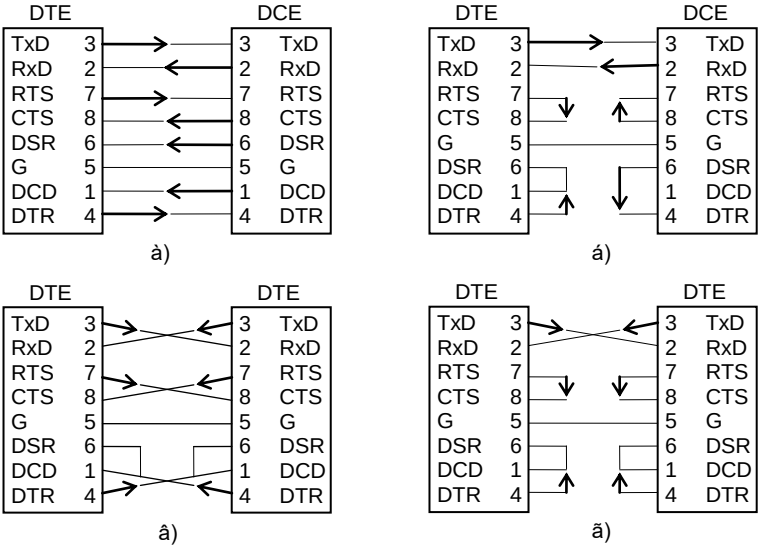


Рис. 5.6. Кабели 9-контактные RS232C: а) для пары DTE - DCE с линиями квитирования; б) для пары DTE - DCE без линий квитирования; в) для пары DTE - DTE с линиями квитирования; г) для пары DTE - DTE без линий квитирования.

Интерфейс RS232C обеспечивает дальность связи до 15 м при скорости передачи 19200 бит/с, линии связи представляют собой многожильный кабель. Для более высокой скорости передачи разработаны другие последовательные интерфейсы, как например, RS423, протокол передачи которого принципиально совместим с интерфейсом RS232. Он обеспечивает скорость до 100 кбит/с и дальность - до 1200 м (но не одновременно). В качестве линий связи здесь используются витые пары. Другой интерфейс RS422 представляет собой протокол для однополярных цепей с дифференциальным выходом сигнала с граничными возможностями 10 Мбит/с и 1200 м.

5.4. Последовательный интерфейс в микроконтроллере AT89LP51/52

Последовательный интерфейс в AT89LP51/52 реализует универсальный асинхронный приемник/передатчик (UART). UART имеет следующие характеристики:

• Полнодуплексный режим
• 8 или 9 бит данных
• Обнаружение ошибок кадра
• Многопроцессорный режим связи с автоматическим распознаванием адреса
• Генератор скорости передачи с использованием Таймера 1 или Таймера 2
• Прерывание при заполнении буфера приема или завершении передачи
• Эмуляция синхронной шины SPI (Serial Peripheral Interface) или шины TWI (Two-wire Serial Interface)

Последовательный интерфейс является полнодуплексным, что означает возможность одновременной передачи и приема. Он также имеет буферизованный прием, что означает, что он может начать прием второго байта до того, как ранее полученный байт будет прочитан из регистра приема. (Однако если первый байт все еще не был прочитан, когда прием второго байта завершен, один из байтов будет потерян.) Доступ к регистрам приема и передачи последовательного порта осуществляется через регистр специальной функции SBUF. Запись в SBUF загружает регистр передачи, а чтение SBUF обращается к физически отдельному регистру приема. Последовательный порт может работать в следующих четырех режимах.

Режим 0: Последовательные данные вводятся и выводятся через вывод RXD. Вывод TXD передает сдвиговые такты. Восемь бит данных передаются/принимаются, причем младший бит идет первым. Скорость передачи программируется на 1/6 или 1/3 системной частоты в режиме совместимости, на 1/4 или 1/2 частоты системы в быстром режиме или может изменяться в зависимости от Таймера 1.

Режим 1: 10 бит передаются (через TXD) или принимаются (через RXD): стартовый бит («0»), 8 бит данных (сначала младший значащий бит) и стоповый бит («1»). При приеме стоповый бит переходит в RB8 регистра специальных функций SCON. Скорость передачи зависит от Таймера 1 или Таймера 2.

Режим 2: 11 бит передаются (через TXD) или принимаются (через RXD): стартовый бит («0»), 8 бит данных (сначала младший бит), программируемый 9-й бит данных и стоповый бит («1»). При передаче 9-му биту данных (TB8 в SCON) может быть присвоено значение «0» или «1». Например, бит четности (P в PSW) можно переместить в TB8. При приеме 9-й бит данных помещается в RB8 регистра специальных функций SCON, а

стоповый бит игнорируется. Скорость передачи программируется на 1/16 или 1/32 системной частоты.

Режим 3: передаются (через TXD) или принимаются (через RXD) 11 бит: стартовый бит («0»), 8 бит данных (сначала младший бит), программируемый 9-й бит данных и стоповый бит («1»). Фактически режим 3 такой же, как и режим 2 во всех отношениях, за исключением скорости передачи данных, которая зависит от Таймера 1 или Таймера 2 в режиме 3.

Во всех четырех режимах передача инициируется любой инструкцией, использующей SBUF в качестве регистра назначения. Прием инициируется в режиме 0 условием RI = 0 и REN = 1 в регистре специальных функций SCON. В остальных режимах прием инициируется поступающим стартовым битом, если REN = 1.

Табл. 5.3. Управляющий регистр последовательного порта SCON

7	6	5	4	3	2	1	0
SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI

Обозн.	Функция
FE	Бит ошибки кадра. Этот бит устанавливается приемником при обнаружении недопустимого стопового бита. Бит FE не сбрасывается действительными кадрами и должен сбрасываться программно.
SM0 SM1	SM0 SM1 Режим Описание 0 0 0 Сдвиговый регистр 0 1 1 8-битный UART 1 0 2 9-битный UART 1 1 3 9-битный UART
SM2	Включает функцию автоматического распознавания адреса в режимах 2 или 3. Если SM2 = 1, то RI не будет. 9-й бит данных (RB8) равен 1, что указывает на адрес, а полученный байт является заданным или широковещательным адресом. В режиме 1, если SM2 = 1, то RI не будет активирован, пока не будет получен действительный стоповый бит, а полученный байт является заданным или широковещательным адресом. В режиме 0 SM2 определяет состояние холостых тактов сдвига таким образом, что такты инвертируются в зависимости от SM2, то есть, когда SM2 = 0, активные холостые такты высокого уровня, а когда SM2 = 1, низкого уровня.
REN	Включает последовательный прием. Устанавливается программным обеспечением. Для отключения приема сбрасывается также с помощью программного обеспечения.
TB8	9-й бит данных, который будет передаваться в режимах 2 и 3. Устанавливается или сбрасывается программно. В режиме 0 установка TB8 включает Таймер 1 в качестве генератора тактовых импульсов.

RB8	В режимах 2 и 3 - 9-й полученный бит данных. В режиме 1, если SM2 = 0, RB8 является принятым стоповым битом. В режиме 0 RB8 не используется.
TI	Флаг прерывания передачи. Устанавливается аппаратно тактом в конце 8-го бита в режиме 0 или в начале стопового бита в других режимы в любой последовательной передаче. Сбрасывается программно.
RI	Флаг прерывания приема. Устанавливается аппаратно тактом в конце 8-го бита в режиме 0 или посередине такта стопового бита в других режимах при любом последовательном приеме (кроме см. SM2). Сбрасывается программно.

5.4.1. Обнаружение ошибок кадрирования и автоматическое распознавания адреса

В дополнение ко всем своим обычным режимам UART может выполнять обнаружение ошибок кадрирования путем поиска отсутствующих стоповых битов и автоматическое распознавание адреса. При использовании для обнаружения ошибок кадрирования UART ищет отсутствующие стоповые биты в кадре. Отсутствующий бит установит бит FE в регистре SCON.

5.4.2. Многопроцессорная связь

Режимы 2 и 3 имеют специальное обеспечение для многопроцессорной связи. В этих режимах принимается 9 бит данных, за которыми следует стоповый бит. 9-й бит сохраняется в RB8. Затем идет стоп-бит. Порт можно запрограммировать таким образом, чтобы при получении стопового бита прерывание последовательного порта активировалось только в том случае, если RB8 = 1. Эта функция активируется установкой бита SM2 в SCON.

В следующем примере показано, как использовать прерывание для многопроцессорной связи.

Когда ведущий процессор должен передать блок данных одному из нескольких ведомых, он сначала отправляет байт адреса, который идентифицирует целевое ведомое устройство. Байт адреса отличается от байта данных тем, что в байте адреса 9-й бит равен «1» и «0» в байте данных. При SM2 = 1 ни одно ведомое устройство не прерывается байтом данных. Однако адресный байт прерывает все ведомые устройства. Каждый ведомый может проверить полученный байт и проверить, адресован ли он. Адресованное ведомое устройство сбрасывает свой бит SM2 и готовится к приему следующих байтов данных. Не адресуемые ведомые устройства устанавливают свои биты SM2 и игнорируют байты данных. Бит SM2 можно ис-

пользовать для проверки достоверности стопового бита в режиме 1. При приеме в режиме 1, если SM2 = 1, прерывание приема не активируется, пока не будет получен стоповый бит.

6. ИНТЕРФЕЙСНЫЕ КОНТРОЛЛЕРЫ

Связь с внешними устройствами, представляющими собой законченные автономные средства, находящиеся за пределами микропроцессорной системы, например, измерительные приборы и другая аппаратура, осуществляется встроенными в систему преобразователями, которые одновременно с относящимися к ним портами ввода/вывода, получили название интерфейсных контроллеров. В их функции помимо ввода/вывода данных входит: формирование интерфейса, физическое управление внешними устройствами в соответствии с командами процессора, перенос протокола обмена данными с процессором на более высокий интеллектуальный уровень, электрическое согласование сигналов.

Передача функции физического управления внешним устройством интерфейсному контроллеру освобождает процессор от низкоуровневого управления устройствами. Функции процессора сводятся к общему управлению контроллерами внешних устройств с помощью высокоуровневых команд и слов состояния, что упрощает программное обеспечение, уменьшает загрузку процессора и повышает общую производительность микропроцессорной системы.

Существуют разнообразные способы реализации интерфейсных контроллеров: от устройств на основе логических микросхем до контроллеров, размещенных на одном кристалле. Стандартизация интерфейсов позволила создать универсальные программируемые интерфейсные однокристальные контроллеры, обеспечивающие разнообразные протоколы передачи данных. В том случае, когда они не удовлетворяют условиям сопряжения нестандартных внешних устройств, применяются заказные интерфейсные контроллеры на основе программируемых логических интегральных схем или специализированных периферийных интерфейсных контроллеров.

6.1. Программируемый контроллер параллельного ввода-вывода

Программируемое устройство ввода-вывода параллельной информации (например, КР1834ВВ55А, MS82C55A) применяется в качестве элемента ввода-вывода общего назначения, сопрягающего внешние устройства с системной магистралью микропроцессора.

Структурная схема контроллера изображена на рис. 6.1. Обмен информацией между шиной данных магистрали и БИС осуществляется через 8-разрядный двунаправленный трехстабильный канал данных D. Для связи с внешними устройствами используются 24 линии ввода-вывода, сгруппированные в три 8-разрядных канала PA, PB и PC, направление передачи информации и режимы работы, которые определяются программным способом.

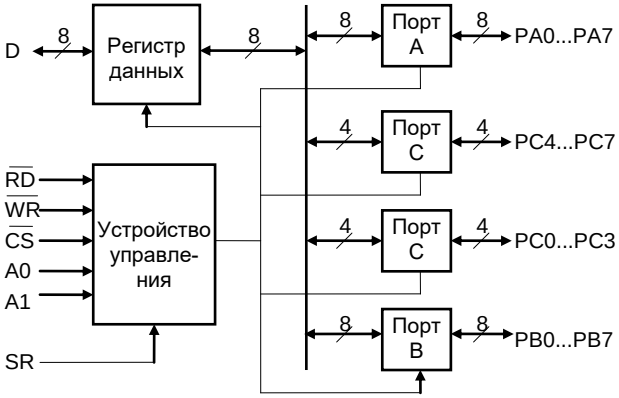


Рис. 6.1. Структурная схема БИС программируемого параллельного интерфейса КР1834ВВ55А

Контроллер может работать в трех основных режимах:

В **режиме 0** обеспечивается возможность синхронной программно управляемой передачи данных через два независимых 8-разрядных канала PA и PB и два 4-разрядных канала PC.

В **режиме 1** обеспечивается возможность асинхронной однонаправленной передачи данных через два независимых канала PA и PB. Порт PC используется при этом для приема и выдачи сигналов квитирования.

В **режиме 2** обеспечивается возможность обмена информацией с внешним устройством через двунаправленный порт PA по сигналам квитирования. Для передачи и приема сигналов управления обменом используются пять линий канала PC.

Выбор соответствующего канала и направление передачи информации через канал определяется сигналами на линиях A0, A1, соединенных обычно с младшими разрядами адресной шины, и сигналами RD', WR', CS' в соответствии с табл. 6.1.

Табл. 6.1. Управляющие сигналы

Сигналы на входах					Направление передачи данных
A1	A0	RD'	WR'	CS'	
					Операции ввода (чтение):
0	0	0	1	0	PA → регистр данных
0	1	0	1	0	PB → регистр данных
1	0	0	1	0	PC → регистр данных
					Операции вывода (запись):
0	0	1	0	0	регистр данных → PA
0	1	1	0	0	регистр данных → PB
1	0	1	0	0	регистр данных → PC
1	1	1	0	0	регистр данных → регистр управляющего слова
					Операции блокировки:
X	X	X	X	1	Регистр данных в высокоомном состояние.
1	1	0	1	0	Запрещено

Режим работы каждого канала определяется содержимым регистра управляющего слова (РУС), формат которого приведен на рис. 6.2. При подаче сигнала сброса SR РУС устанавливается в состояние, при котором все каналы настроены на режим 0 для ввода информации. Режим работы каналов можно изменять как в начале, так и в процессе выполнения программы, что позволяет обслуживать различные внешние устройства в определенном порядке одним контроллером.

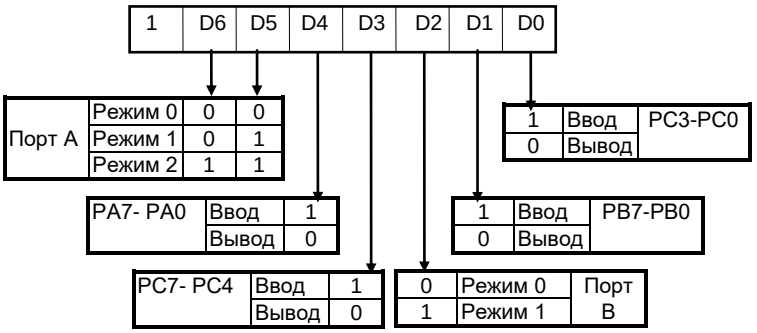


Рис. 6.2. Формат управляющего слова, задающего режим работы контроллера

При изменении режима работы любого канала все входные и выходные регистры каналов и триггеры состояний сбрасываются.

Графическое представление режимов работы каналов показано на рис. 6.3. Контроллер предусматривает также комбинированные режимы работ.

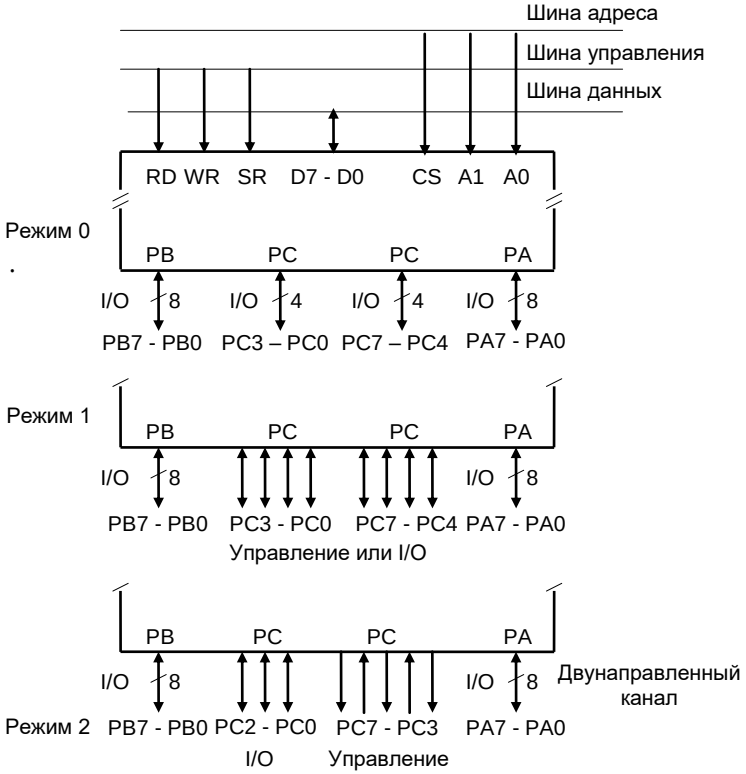
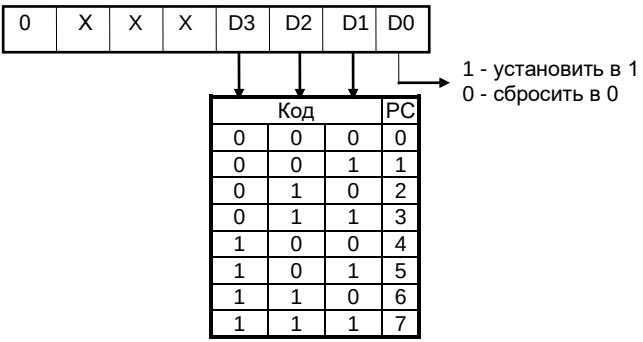


Рис. 6.3. Графическое представление режимов работы

В дополнение к основным режимам работы контроллер обеспечивает возможность независимой программной установки в 1 и сброс в 0 любого из разрядов регистра канала PC. Формат управляющего слова для установки/сброса разрядов регистра показан на рис. 6.4.



Код			PC
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Рис. 6.4. Формат управляющего слова для установки/сброса разрядов регистра PC

В качестве примера использования контроллера параллельного ввода-вывода данных взята система сбора информации, структурная схема которой приведена на рис. 6.5. Опрос датчиков происходит по прерыванию от таймера, который, кроме того, сбрасывает коммутатор, подключая датчик под номером 1. Сигнал с датчиков в виде уровня напряжения в диапазоне от 0 до 10 В поступает через коммутатор на аналого-цифровой преобразователь (АЦП). 10-разрядный АЦП после завершения преобразования величины напряжения в код вырабатывает сигнал готовности, после чего микропроцессор принимает код, загружает его в буфер памяти и выдает управляющий сигнал для переключения коммутатора к следующему датчику.

Программа ввода-вывода организует цикл обработки сигналов от всех датчиков системы, осуществляет запись полученной информации в выделенную область памяти и с помощью режима поразрядного управления каналом PC генерирует сигнал переключения датчиков.

Программа предполагает использование в системе сбора данных 10-разрядного АЦП, 8 младших разрядов которого принимаются по каналу PB, а 2 старших разряда по линиям PC0 и PC1. По линии PC7 поступает сигнал готовности АЦП, а сброс и установка разряда PC6 используется для выработки сигнала переключения коммутатора.

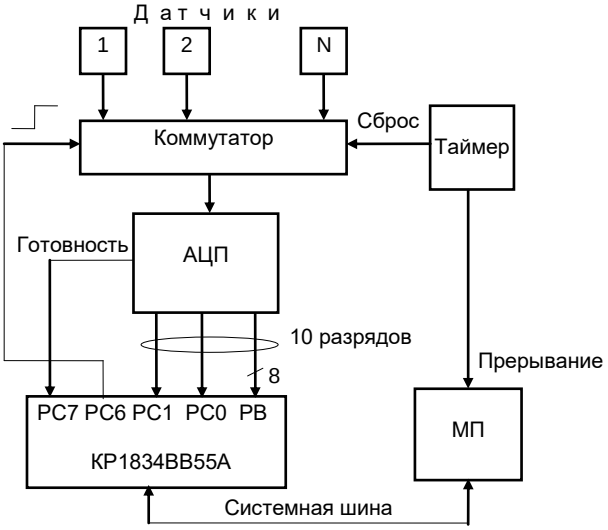


Рис. 6.5. Структурная схема системы сбора данных

Код ВВН задает следующие режимы работы контроллера:

канал РВ - ввод в режиме 0

линии РС1, РС0 - ввод в режиме 0

канал РА - не используется, но настраивается на ввод в режиме 1, чтобы использовать линию РС7 для ввода, а линию РС6 для формирования сигнала переключения коммутатора.

Канал РС в программе имеет адрес FEH, канал РВ - FDH, регистр управления - FFH. Начальному адресу области памяти (буферу) для размещения данных присвоено символическое имя ‘array’. Текст программы обслуживания прерывания на ассемблере 8051 имеет следующий вид:

```
IOAqpar: clr EA           ;Запрещение прерывания
         mov R1,0xFF      ;Адрес РУС
         mov A,#0xBB      ;Код режима
         movx @R0,A       ;Загрузка режима в РУС
         mov R7,#N        ;Установка счетчика датчиков
         mov DPTR,#array  ;Загрузка адреса начала буфера

wait:    mov R0,#0xFE     ;Адрес порта РС
         movx A,@R0       ;Ввод из порта РС
         mov C,ACC.7      ;Проверка готовности АЦП
```

```
jnc wait          ;Если АЦП не готов

mov R0,#0xFD      ;Адрес порта PB
movx A,@R0        ;Ввод 8 младших разрядов АЦП из
                  ;порта PB
movx @DPTR,A      ;Запись в буфер памяти
inc DPTR          ;Следующая ячейка памяти
mov R0,0xFE       ;Адрес порта PC
movx A,@R0        ;Ввод из порта PC
anl A,#3          ;Выделение двух старших разрядов
                  ;АЦП
movx @DPTR,A      ;Запись в буфер памяти
inc DPTR          ;Следующая ячейка памяти
mov R0,#0xFF      ;Адрес РУС
mov               ;Сигнал для переключения
A,#00001100B      ;коммутатора
movx @R0,A        ;Сброс разряда PC6
mov               ;Сигнал для переключения
A,#00001101B      ;коммутатора
movx @R0,A        ;Установка разряда PC6
djnz R7,wait      ;Переход на следующий датчик
setb EA           ;Разрешение прерывания
END
```

6.2. Программируемый контроллер последовательного ввода-вывода

Контроллер, реализованный в БИС M1821ВВ51А, является универсальным синхронно-асинхронным приемопередатчиком (УСАПП), предназначенным для аппаратного обеспечения последовательного протокола обмена между микропроцессором и каналами последовательной передачи информации.

УСАПП преобразует параллельный код, получаемый от центрального процессора, в последовательный поток данных со служебными битами и выдает его в последовательный канал связи с заданной скоростью, а также выполняет обратное преобразование последовательного потока данных в поток параллельных 8-разрядных слов. Передаваемая и принимаемая информация при необходимости может проверяться на четность (нечетность).

Контроллер программируется на выполнение почти всех применяющихся протоколов последовательной передачи данных и работает в двух режимах: синхронном и асинхронном. Программирование на тот или иной режим работы выполняется записью в соответствующие регистры инструкции режима, служебных синхросимволов и инструкции команд.

Контроллер обеспечивает максимальную скорость передачи информации в синхронном режиме работы до 64 Кбит/с, (минимальная не ограничена и определяется внешними устройствами).

Структурная схема контроллера приведена на рис. 6.6. Он состоит из буфера шины данных, а также устройств передачи, приема и управления.

Основу передатчика составляет 13-разрядный сдвиговый регистр, хранящий очередное выходное слово, последовательно сдвигаемого в линию TxD. Управление работой передатчика осуществляется схемой, которая отслеживает прием новых данных, при необходимости добавляет к ним контрольный бит, обрамляет старт-стоповыми битами и синхронизирует вывод из сдвигающего регистра с тактовыми сигналами, поступающими по линии TxC. Выходные сигналы TxRDY и TxEND информируют внешние устройства о готовности передатчика и конца передачи соответственно.

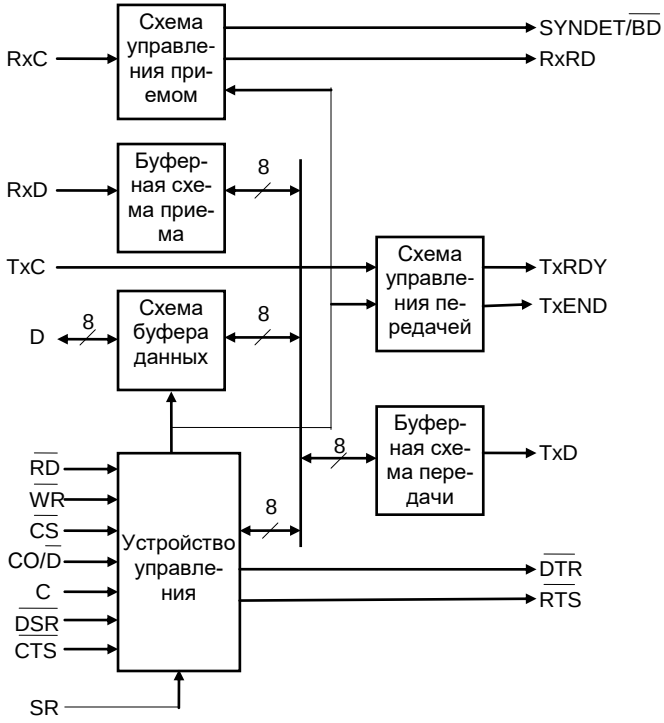


Рис. 6.6. Структурная схема УСАПП М1821ВВ51А

Приемник содержит два 9-разрядных сдвиговых регистра. Информация с входа RxD последовательно поступает на один из четырех входов в зависимости от длины передаваемого слова первого регистра, и затем, на один из входов второго регистра. Управление записью входной информации осуществляется схемой управления приемом, содержащей логику формирования синхроимпульсов приема (сигналы RxC, SYNDET/BD'), счетчик числа принятых бит, схему контроля четности, триггер ошибки четности, триггер ошибки кадра и триггер ошибки переполнения.

Буфер шины данных представляет собой 8-разрядное трехстабильное двунаправленное устройство для связи с процессором, которое состоит из входного и выходного регистров данных, а также регистра слова состояния УСАПП.

Устройство управления включает 8-разрядные регистр режима, регистр команд и два регистра синхросимволов. Синхросимволы предназначены для обеспечения работы УСАПП в синхронном режиме. Встроенный компаратор сравнивает слова, хранящиеся в регистрах синхросимволов и регистрах приемника, и результат сравнения используется для управления синхронизацией.

Перед началом работы контроллер должен быть установлен в исходное состояние либо с помощью сигнала сброса SR длительностью больше шести периодов тактовой частоты C, либо программным способом.

Возможные варианты сочетаний основных управляющих сигналов RD', WR', CO/D', CS, и соответствующие направления передачи информации приведены в табл. 6.2.

Табл. 6.2. Управление направлением передачи информации

Сигналы на входах				Направление и вид информации
CO/D'	RD'	WR'	CS'	
1	1	0	0	Канал данных → УСАПП (управление)
0	1	0	0	Канал данных → УСАПП (данные)
1	0	1	0	УСАПП → канал данных (состояние)
0	0	1	0	УСАПП → канал данных (данные)
X	1	1	0	Высокоомное состояние канала данных
X	X	X	1	Высокоомное состояние канала данных

В синхронном режиме контроллер формирует посылку кадров в формате, изображенном на рис. 6.7. В поток передаваемой информации вводятся один или два синхросимвола (устанавливается программно), данные могут иметь от 5 до 8 бит и если запрограммирован контроль данных по четности или нечетности, то добавляется бит контроля. Временные длины

синхросимволов и данных равны. Если между словами данных имеются временные промежутки, то они для предотвращения потери синхронизации заполняются синхросимволами.

Для управления модемом предназначены следующие сигналы устройства управления:

DSR' (Data Set Ready) - Готовность приемника модема (вход). Проверяется программно.

DTR' (Data Terminal Ready) - Запрос готовности приемника модема (выход). Управляется программно.

RTS' (Request To Send) - Запрос готовности передатчика модема (выход). Управляется программно.

CTS' (Clear To Send) - Готовность передатчика модема (вход). Разрешает УСАПП передачу данных.

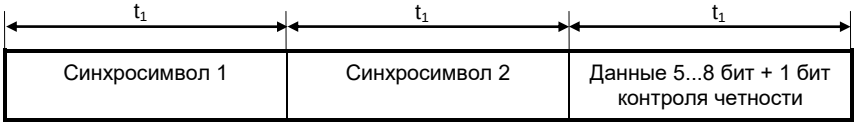


Рис. 6.7. Формат кадра при синхронном режиме обмена данными

Формат и содержание инструкции режима при синхронном обмене данных приведены на рис. 6.8.

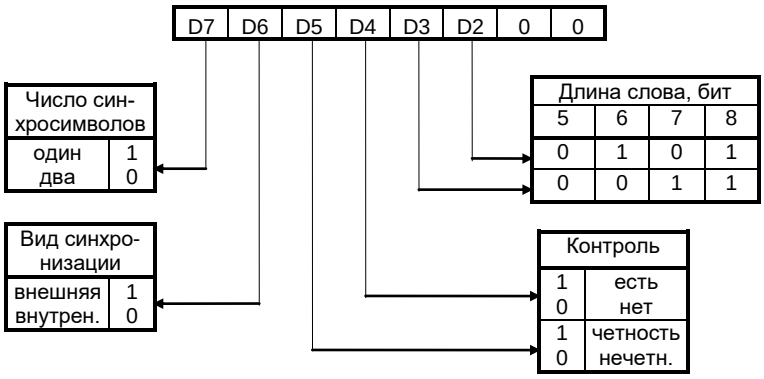


Рис. 6.8. Формат инструкции синхронного режима

В асинхронном режиме УСАПП формирует кадр в формате, показанном на рис. 5.4, причем количество бит в слове данных, длительность стопбита и контроль четности устанавливается программно в инструкции ре-

жима работы, формат и содержание которой приведены на рис. 6.9. Асинхронный режим имеет три подрежима, отличающиеся друг от друга различными соотношениями численных значений частот синхронизации приема f_{RxC} и передачи f_{TxC} к тактовой частоте f_C: 1:1, 1:16 и 1:64, также задаваемых программным способом.

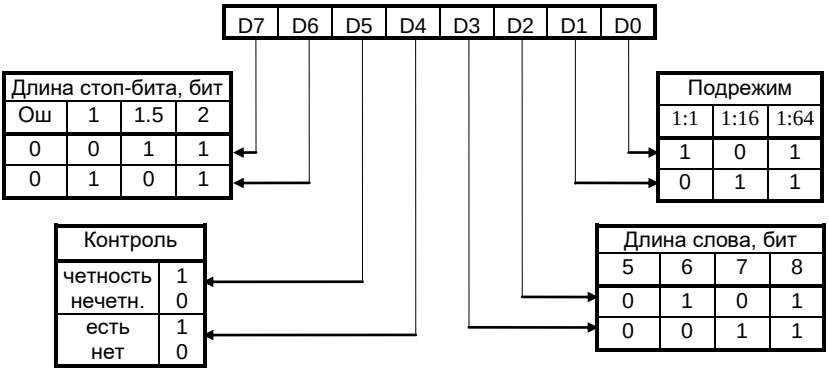


Рис. 6.9. Формат инструкции асинхронного режима

Для оперативного управления работой контроллера используется инструкция команды, изображенная на рис. 6.10, которая включает в себя биты разрешения /запрещения передачи и приема (биты D0 и D2), перехода в режим поиска синхронизации (бит D7), а также для программной установки УСАПП в исходное состояние и сброса триггеров ошибок (биты D6 и D4). Вывести контроллер из состояния паузы (бит D3) можно внешним сигналом SR, программным сбросом (D6) и паузой (D3). Биты D1 и D5 инструкции команды соответствуют внешним выводам контроллера.

D7	D6	D5	D4	D3	D2	D1	D0
Поиск синхр.	Сброс УСАПП	RTS	Сброс ошибки	Пауза	Прием	DTR	Передача

Рис. 6.10. Формат командной инструкции УСАПП

Для организации программно-управляемого обмена по условию имеется слово состояния, чтение которого возможно в любой момент времени. В состав слова состояния входят флаги готовности передатчика и приемника D0 и D1. После выдачи слова данных флаг TxRDY устанавливается в «0» и снова в «1» после передачи данных в буфер передатчика. Аналогично работает флаг RxRDY при заполнении и считывании буфера принятых

данных. Флаг окончания передачи (бит D2) указывает, что передатчик пустой и можно переключиться на прием. Флаги ошибок (биты D3, D4, D5) не прерывают работу УСАПП и обрабатываются программно. Биты D0, D1, D2, D6 и D7 соответствуют внешним выводам контроллера.

D7	D6	D5	D4	D3	D2	D1	D0
DSR	SYNDET	Ошибка формата	Ошибка переполнения	Ошибка четности	TxEND	RxRDY	TxRDY

Рис. 6.11. Формат слова состояния УСАПП

Фрагмент программы на языке ассемблера 8051, приведенный ниже, представляет один из вариантов организации асинхронной передачи элементов массива по последовательному каналу.

Код 7DH инструкции режима обеспечивает асинхронный режим с частотой передачи, равной частоте синхронизации, контролем по четности, одним стоп-битом и 8-разрядным словом данных. Для ввода в УСАПП инструкций и данных используются символические адреса ‘instr’ и ‘dat’. Идентификатор N обозначает число элементов массива, а ‘array’ - адрес начала буфера передаваемых данных.

```
IOAqser:  clr EA           ;Прерывания запрещены
          mov R0,#instr    ;Запись инструкции программного
                           ;сброса
          mov A,#0x40
          movx @R0,A
          mov A,#0x7D      ;Запись инструкции режима
          movx @R0,A
          mov A,#0x31      ;Запись инструкции команды
          movx @R0,A       ;передачи
          mov R7,#N        ;Установка счетчика массива
                           ;данных
          mov R1,#array    ;Загрузка начального адреса мас-
                           сива

trans:    mov A,@R1        ;Передача элемента массива в ак-
                           кумулятор
          mov R0,#dat      ;Запись элемента массива в УСАПП
          movx @R0,A
          mov R0,#instr    ;Слово состояния УСАПП

wait:     movx A,@R0
          rr A             ;Контроль готовности передатчика
          jnc wait         ;Если не готов, то повтор
```

```
inc R1          ;Следующий элемент массива
djnz R7,trans   ;Если все, то конец
mov R0,#instr   ;Запись инструкции команды конца
mov A,0x38      ;передачи
movx @R0,A
setb EA         ;Прерывания разрешены
END
```

6.3. Периферийные интерфейсные микроконтроллеры

Для построения контроллеров ввода-вывода перспективными являются микроконтроллеры PIC (Peripheral Interface Controller), созданные фирмой Microchip. Они предназначены для разработки несложных устройств, так называемого “нижнего уровня”, когда универсальные микроконтроллеры для этой цели становятся избыточными. Конкурентоспособность PIC на рынке обеспечивается сочетанием высокого быстродействия, низкого энергопотребления, высокой степени интеграции памяти и периферийных устройств в малом корпусе с невысокими ценами.

Микроконтроллеры PIC семейства 16CXX имеют аппаратные особенности, отмеченные в табл. 6.3.

Табл. 6.3. Основные характеристики PIC семейства 16CXX

PIC	16C54	16C55	16C56	16C57	16C64	16C71	16C74	16C84
Тактовая частота	16	16	16	16	25	16	20	10
Память программ EPROM	0,5x12	0,5x12	1x12	2x12	2x14	1x14	4x14	-
килослов EEPROM	-	-	-	-	-	-	-	1x14
Память данных, байт	25	25	25	72	128	36	192	36
Память данных EERAM	-	-	-	-	-	-	-	64
Глубина стека	2	2	2	2	8	8	8	8
Таймер0 (8+8 бит)	+	+	+	+	+	+	+	+
Таймер1 (16 бит)	-	-	-	-	+	-	+	-
ШИМ выход	-	-	-	-	+	-	+	-
Последовательный порт	-	-	-	-	+	-	+	-
АЦП(8 бит),число каналов	-	-	-	-	-	4	8	-
Число прерываний	-	-	-	-	8	4	12	4
Порты ввода-вывода	12	20	12	20	33	13	33	13
Число выводов корпуса	18	28	18	28	40	18	40	18

Все PIC обладают встроенными ПЗУ (0,5 - 4 килослов) и ОЗУ (32 - 256 байт). Основная часть контроллеров поставляется с однократно программируемым ПЗУ, но некоторые имеют РПЗУ с электрическим стиранием. Кроме того, PIC снабжены таймерами (от 1 до 3 шт.), встроенной системой

сброса, системой защиты от сбоев (watchdog timer), внутренним тактовым генератором, который может запускаться, как от кварцевого резонатора, так и от RC-цепи, в широком диапазоне частот - от 32 Кгц до 25 МГц. Число портов ввода-вывода - от 12 до 33, каждый из них может быть запрограммирован на ввод или вывод с нагрузочной способностью до 25 мА. PIC 16C64 и 16C74 дополнительно имеют выход с ШИМ, с помощью которого можно реализовать ЦАП с разрешением до 16 разрядов, а также последовательный двунаправленный синхронно-асинхронный порт. PIC 16C71 и 16C74 имеют внутренний 8-разрядный АЦП с устройством выборки-хранения и входным аналоговым мультиплексором.

Микроконтроллеры работают в широком диапазоне напряжений от 2,5 до 6,25 В, при этом ток потребления составляет 15 мкА на частоте 32 Кгц, 7 мА - 16 МГц, 20 мА - 20 МГц, а ток покоя (Sleep mode) - до 3 мкА.

С программной точки зрения PIC представляет собой 8-разрядный RISC-процессор с гарвардской архитектурой (память команд и данных разделены). Память команд имеет разрядность от 12 до 14 бит в зависимости от типа процессора. Благодаря сокращенному числу команд (от 33 до 35) все они имеют одинаковую длину в одно слово и выполняются, кроме команд ветвления, за один машинный цикл, состоящий из 4 тактов. ОЗУ представляет собой, в сущности, регистровый файл с возможностью адресации в коде команды к любой ячейки. Стек реализован аппаратно и имеет максимальную глубину или 2, или 8 слов. Почти во всех контроллерах имеется система прерываний, источником которых может быть таймер, а также изменение сигнала на некоторых входах. В PIC предусмотрен “плавающий” бит защиты памяти программ, который не позволит нелегально копировать устройство.

Для разработки устройств на базе PIC необходимы инструментальные средства программирования и отладки. Пакет PIC-Master фирмы Microchip для этих целей включает в себя макроассемблер, компоновщик, отладчик-симулятор, внутрисхемный эмулятор и программатор.

6.4. Программируемые логические интегральные схемы фирмы Xilinx

Для заказных устройств интерфейса ввода-вывода нестандартного типа предпочтительно использование программируемых логических интегральных логических схем (ПЛИС), например, фирмы Xilinx. Основное преимущество их в том, что разработка устройства полностью автоматизирована. В течение часа можно разработать принципиальную электрическую схему устройства, реализовать ее в виде функционирующей интегральной схемы и затем протестировать, в то время как с обычными логическими

матрицами на это потребуется месяцы. Причем, код конфигурации ПЛИС хранится в ОЗУ, что позволяет процесс разработки сделать интерактивным, легко модифицируя электрическую схему по мере необходимости, например, для адаптации к различным внешним устройствам, или на основе этого же кристалла разрабатывать новое устройство.

Фирма Xilinx начала производство ПЛИС, выпустив четыре семейства: XC2000, XC3000, XC3100 и XC4000, отличавшихся количеством вентилей на кристалле (см. табл. 6.4) и архитектурными особенностями. Архитектура этих ПЛИС базируется на трех типах программируемых элементов - блоках ввода-вывода IOB (Input/Output Block), конфигурируемых логических блоках CLB (Configurable Logic Block) и ресурсах связи. IOB обеспечивают внутренний интерфейс между логическими элементами и выводами на корпусе интегральной микросхемы, CLB выполняют заданные разработчиком логические функции, а ресурсы связи снабжают эти блоки сигналами. Логические функции и связи определяются программой, находящейся в ячейках статической памяти, которая загружается туда либо из последовательного или параллельного внешнего ПЗУ при подаче напряжения питания на устройство (ведущий режим), либо в режиме программирования ПЛИС (ведомый режим).

Табл. 6.4. Количество элементов на ПЛИС

ПЛИС	**Количество вентилей**	**Количество блоков CLB**	**Количество блоков IOB**
XC2000	1200 - 1800	100	74
XC3000	2000 - 9000	320	144
XC3100	2000 - 9000	320	144
XC4000	2000 - 20000	900	240

Размещение элементов ПЛИС показано на рис. 6.12. Массив логических блоков размещается в центре кристалла, а по его периферии размещены блоки ввода-вывода. Линии связи различных назначений с помощью матриц переключений и программируемых точек соединений обеспечивают связь как глобальную - между блоками, так и местную - внутри блока.

Конфигурируемый логический блок в ПЛИС XC4000 с 9 входами и 4 выходами содержит три функциональных генератора, два D-триггера в качестве запоминающих элементов, а также логику быстрого переноса для арифметических операций. Функциональные генераторы G и F могут также использоваться, как быстродействующие 16x1 бит ОЗУ.

Каждый блок ввода-вывода управляет сигналом на ножке корпуса микросхемы и может программироваться на ввод или вывод, либо на двунаправленное направление сигналов. Сигналы могут направляться через управляемый усилитель с 3-мя состояниями прямо или через регистр-защелку. Выходной буфер обеспечивает ток до 12 мА, а два, соединенных наружи по схеме логического И, до 24 мА.

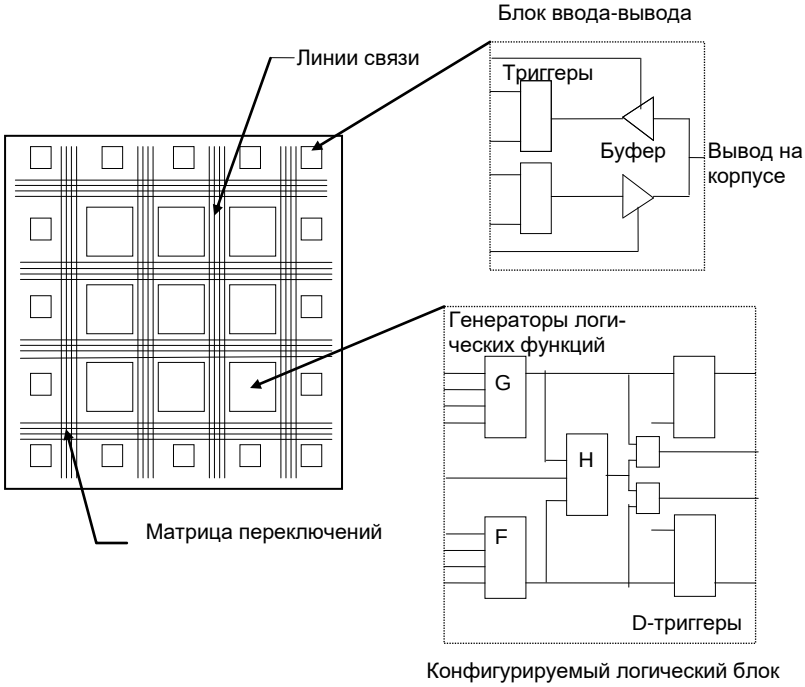


Рис. 6.12. Структура ППЛИС XC4000

Процесс разработки устройств на базе ППЛИС включает в себя три этапа:

- разработку принципиальной электрической схемы и ввод ее в компьютер,
- реализацию логики имеющимися в микросхеме средствами, посредством выделения на узел необходимого числа блоков,
- размещения их в конкретных областях кристалла и выполнения соответствующей маршрутизации сигналов,
- тестирование разработки и испытание устройства.

Фирма Xilinx предлагает для этих целей инструментальные средства XACT (Xilinx Automatic CAE (Cell Array Environment) Tools) с возможностью задания опций в командной строке, а также наличием выпадающих меню и контекстного справочника.

Ввод данных в компьютер может осуществляться с помощью либо графического редактора, изображая принципиальную схему разработки имеющимися графическими примитивами, либо текстового редактора, описывая ее средствами соответствующего языка. Для графического ввода можно использовать систему автоматизированного проектирования OrCAD фирмы OrCAD Systems Corporation, программного ввода - язык ABEL-HDL или VHDL. В результате первого этапа после трансляции создается файл в формате XNF (Xilinx Netlist Format).

Используя полученный файл для аппаратной реализации разработки, после оптимального распределения ресурсов ПЛИС для логических узлов разрабатываемого устройства, размещения их и задания между ними связей с помощью программ пакета XACT получают двоичный файл конфигурируемых данных для загрузки его в микросхему.

Тестирование является неотъемлемой частью процесса разработки устройств для гарантирования их работоспособности в наихудших условиях. Система разработки XACT поддерживает три способа тестирования разработки:

- интерактивное тестирование на этапе начального ввода данных разработки,
- проверку функционирования узлов и определение временных характеристик с помощью симулятора после этапа реализации устройства,
- схемное тестирование устройства на рабочей плате после загрузки микросхемы конфигурируемыми данными.

Одной из основных тенденций развития современных ПЛИС является их энергонезависимая “прошивка” по технологии ISP (In System Programmable). При этом программирование и перепрограммирование микросхемы осуществляется через 4-разрядный интерфейс в стандарте JTAG, а напряжение программирования (12 В) формируется непосредственно внутри кристалла. ПЛИС семейства MAX9000 фирмы Altera изготавливаются по технологии EEPROM, обеспечивающей до 100 циклов перезаписи, а фирма Xilinx в семействе XC9500 применила FLASH-технологию компании Seiko-Epson со значительным увеличением количества циклов - до 10000 раз.

Другим направлением развития является повышение их универсальности. Наиболее ярко эта тенденция выразилась в возможности формирования больших блоков памяти внутри микросхемы. В семействе ПЛИС FLEX10K фирмы Altera появились специализированные блоки EAB (Embedded Array Blocks), построенные по принципу “море вентилей”. Один такой блок имеет емкость два килобита и позволяет сформировать память 2048x1, 1024x2, 512x4 или 256x8 с циклом 12-14 нс.

7. ОБМЕН ДАННЫМИ ПО ПРЕРЫВАНИЮ

7.1. Особенности режима прерываний

В системах реального времени широко используется режим программного управления (программирование по готовности, как, например, в системе сбора данных, рассмотренной в предыдущем разделе), поскольку он обеспечивает максимальную скорость приема информации. Однако если регистрируемые события возникают относительно редко, такой режим становится неэффективным по отношению к выполняемым действиям процессора, так как он в основном тратит время на опрос готовности внешнего устройства (ВУ). В этом случае предпочтительней является режим прерываний.

Если внешнее устройство работает в режиме прерываний, то в памяти микропроцессорной системы находятся две программы: фоновая (основная), которая может не иметь отношения к внешнему устройству, и программа обработки прерывания, обеспечивающая обмен информацией между процессором и внешним устройством. Процессор выполняет основную программу до тех пор, пока внешнее устройство не будет готово к приему или передачи данных, о чем оно сообщит ему сигналом по линии запроса прерывания. Процессор прервет выполнение фоновой программы и перейдет к выполнению программы обработки прерывания. После завершения обработки прерывания процессор вернется к выполнению основной программы в том месте, где она была прервана сигналом от ВУ.

Режим прерываний обеспечивает высокую эффективность использования времени процессора: если ВУ не готово к обмену информацией, он выполняет основную программу. В режиме прерываний процессор может обслуживать параллельно любое количество внешних устройств. Каждому ВУ соответствует своя программа обработки прерывания. При этом возможны вложенные прерывания, когда сигнал прерывания от другого ВУ поступает на процессор, уже выполняющего программу обработки преры-

вания. Возврат в основную программу осуществляется в обратном порядке, по мере завершения выполнения обработки вложенных прерываний. Использование в системе прерываний приоритетов позволяет организовать обслуживание внешних устройств в порядке важности. В первую очередь обслуживаются устройства с максимальным приоритетом - сигнал прерывания от устройства с меньшим приоритетом не сможет прервать обработку его прерывания.

При переходе процессора на выполнение программы обработки прерывания от ВУ обычно реализуется следующая последовательность действий:

1) Внешнее устройство запрашивает прерывание.

2) После завершения текущей команды и при наличии разрешения процессор выдает сигнал подтверждения прерывания.

3) Запоминается содержимое счетчика команд (адрес следующей команды) и осуществляется переход по адресу подпрограммы обслуживания прерывания.

4) В подпрограмме запоминается, при необходимости, содержимое внутренних регистров (рабочих и состояния) и выполняется тело подпрограммы.

5) После выполнения подпрограммы восстанавливаются внутренние регистры, и осуществляется возврат к выполнению прерванной программы.

В микропроцессорных системах используется два метода реализации приведенной последовательности: прерывание с опросом и прерывание по вектору. В первом случае опрашивается каждое внешнее устройство, пока не обнаружится то, которое запрашивает прерывание. Далее происходит переход на соответствующую подпрограмму обслуживания прерывания. При этом методе приоритет устройства определяется его местом в последовательности опроса.

В случае прерывания по вектору при получении запроса от внешнего устройства управление передается непосредственно соответствующей подпрограмме обслуживания, т.е. устройство распознается сразу же после поступления сигнала подтверждения прерывания. Вектор прерываний представляют собой массив адресов программ обработки прерываний. При успешном выполнении прерывания внешнее устройство сообщает процессору адрес точки входа в вектор прерываний, а процессор, в свою очередь,

использует этот адрес для передачи управления соответствующей программе обработки.

7.2. Программируемый контроллер прерываний

Для организации связи внешнего устройства с микропроцессором по прерыванию в микропроцессорных системах используется специализированные контроллеры прерываний, позволяющие существенно сократить средства программного обеспечения режима прерываний, а также реальные затраты времени при выполнении многоуровневых прерываний. Одним из широко используемых подобных БИС является программируемый контроллер прерываний М1821ВН59А, обслуживающий до 8 запросов на прерывание с возможностью расширения до 64 путем их каскадного соединения и обеспечивающий сменяемую программно приоритетность поступающих запросов. Структурная схема контроллера приведена на рис. 7.1.

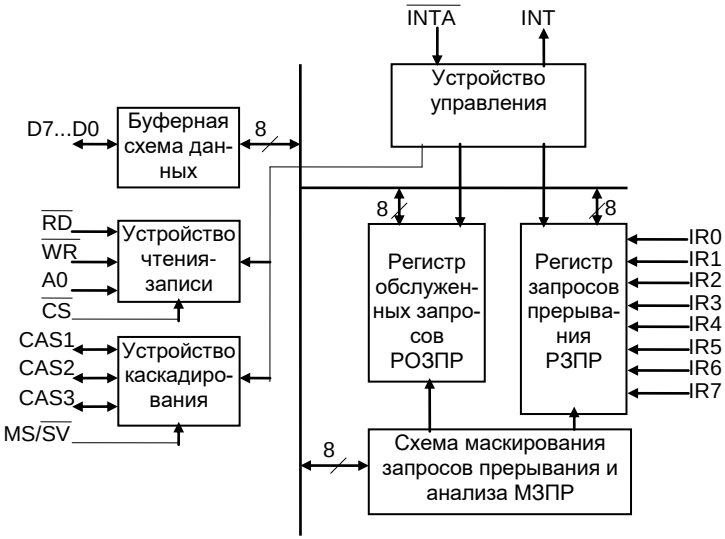


Рис. 7.1. Структурная схема программируемого контроллера прерываний

Запросы от внешних устройств IR поступают в регистр запросов прерываний (РЗПР), предназначенный для их записи и хранения. Запись в соответствующий разряд РЗПР происходит при изменении на соответствующем входе БИС напряжения от низкого до высокого уровня. IR0...IR7 - индивидуальные асинхронные входы, на которых напряжение высокого уровня должно удерживаться до получения первого импульса низкого уровня

на входе INTA’. Разряд РЗПР, соответствующий обслуживаемому запросу, при поступлении второго импульса INTA’ возвращается в исходное состояние. Содержимое регистра РЗПР может быть считано на шину данных.

Регистр обслуженных запросов (РОЗПР) предназначен для хранения сигналов, пропущенных схемой маскирования обслуживаемых в данный момент запросов прерываний, поступающих в РЗПР от внешних устройств. Соответствующий разряд РОЗПР устанавливается в 1 после поступления второго импульса INTA’, и он сохраняет свое состояние до получения команды *Конец прерывания*. Содержимое РОЗПР может также быть прочитано микропроцессором.

Схема маскирования запросов прерывания и анализа их по приоритету (МЗПР) взаимодействует с регистрами РЗПР и РОЗПР - она разрешает или запрещает прохождение сигналов с выхода РЗПР в соответствии с заданной маской или по уровню их приоритета. Код маски записывается в регистр и хранится там до записи новой маски или до установки микросхемы в исходное состояние. Сигналы, прошедшие схему маскирования, анализируются по уровню их приоритета. Запросы с более высоким уровнем приоритета, занесенные в РОЗПР по мере их обслуживания, запрещают прохождение через МЗПР равных или низших по уровню приоритета запросов. Для разрешения их прохождения необходимо выполнить или команду *Конец прерывания,* или команду специального маскирования.

Устройство чтения-записи, на которое подаются сигналы RD’, WR’, A0 позволяют записать команды в различные регистры контроллера и считать их содержимое на шину данных. Напряжение низкого уровня на входе WR’ позволяет записывать управляющие слова команд инициализации и операции. Напряжение низкого уровня на входе RD’ позволяет считать содержимое РЗПР, РОЗПР, либо двоично-десятичный код запроса прерывания.

Устройство управления предназначено для выдачи сигнала INT после поступления одного или нескольких запросов на входы IR0...IR7. Оно выдает также управляющие сигналы для формирования команды CALL.

Устройство каскадирование расширяет число входов обслуживаемых запросов до 64.

Буферная схема шины данных, 8-разрядная, двунаправленная с тремя состояниями, соединяет контроллер с системной шиной данных посредством выводов D7...D0. При программировании контроллера через нее записываются управляющие слова, и считывается служебная информация. В

режиме прерываний по запросу в процессе подтверждения (поступления трех импульсов INTA') через буферную схему в системную шину данных выдается трехбайтовая команда CALL. В остальное время выходы буфера находятся в высокоимпедансном состоянии.

Контроллер реализует оба указанных метода прерываний путем программной установки его в соответствующий режим работы. Причем, режим работы, соответствующий методу прерывания с опросом, называется обслуживанием по результатам опроса, а методу прерываний по вектору - обслуживанием по запросу.

В режиме обслуживания по результату опроса блокируется выход INT, поскольку микропроцессор сам является инициатором опроса. В этом случае по каждому сигналу RD', поступающему после команды *Обслуживание по результатам опроса*, при наличии запросов считывается код номера запроса, имеющего наивысший в данный момент уровень приоритета.

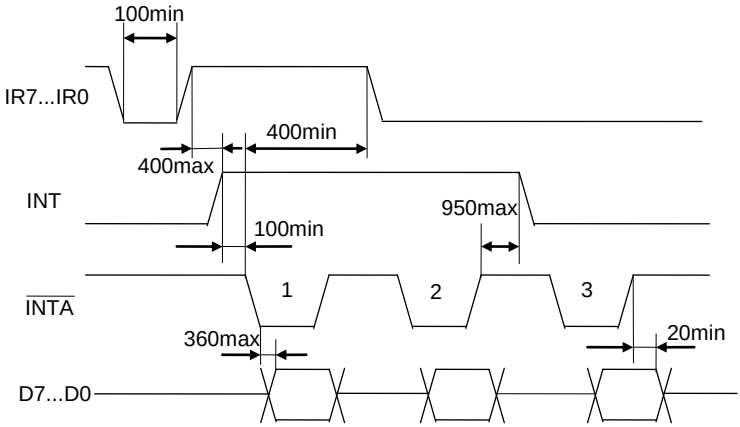


Рис. 7.2. Временные диаграммы режима обслуживания прерываний по запросу

В режиме обслуживания по запросу, временные диаграммы для которого изображены на рис. 7.2, контроллер, получая запросы от внешнего устройства, запоминает их, выделяет запрос с высшим уровнем приоритета и сравнивает его по уровню приоритета с зарегистрированными запросами

в РОЗПР. Если уровень приоритета выделенного запроса оказывается выше, чем у зарегистрированных запросах, выдает для микропроцессора сигнал INT. После получения со стороны МП сигнала подтверждения прерывания INTA’ контроллер вырабатывает адрес входа в вектор прерываний, т.е. начальный адрес подпрограммы обслуживания того устройства, которое вызвало выдачу сигнала INT. Это осуществляется путем посылки в МП трехбайтовой команды CALL.

При получении сигнала INTA’ контроллер выдает на шину данных код команды CALL. Этот код инициализирует со стороны микропроцессора еще 2 сигнала INTA’, которые позволяют послать на шину данных сформированный контроллером адрес - сначала младший, а затем старший байты.

Установка контроллера в исходное состояние, задание алгоритма обслуживания прерываний выполняется с помощью определенных команд. Контроллер прерываний М1821ВН59А выполняет также следующий набор операций:

1) операции маскирования: индивидуальное маскирование запросов, специальное маскирование (снятие действия этого запроса на запросы с более низким приоритетом);

2) операции установки статуса уровней приоритета: по установке в исходное состояние, по обслуженному запросу, по указанию;

3) операции конца прерываний: обычный конец прерывания (устанавливается в «0» разряд РОЗПР, соответствующий последнему обслуженному запросу), специальный конец прерывания (устанавливается в «0» указанный разряд РОЗПР);

4) операции чтения: чтение регистра запросов, регистра обслуживаемых запросов, регистра маски.

При поступлении запросов на контроллер он определяет, какое из них обладает наивысшим приоритетом. При этом уровни приоритетов входов IR0...IR7 заранее заданы, и находятся всегда в строго определенном отношении друг с другом, задаваемым приоритетным кольцом. При установке в исходное состояние наивысшим приоритетом обладает вход IR0, а самым низким, называемым дном приоритетного кольца - вход IR7. В процессе работы дно приоритетного кольца может произвольно устанавливаться.

7.3. Многоуровневое обслуживание прерываний

Число обслуживаемых запросов может быть увеличено до 64 путем соединения аналогичных микросхем с помощью специальной шины CAS0...CAS2, как изображено на рис. 7.3. При этом в зависимости от подачи соответствующих сигналов на выводы MS/SV', одна из них выступает в качестве ведущей, а другие - в качестве ведомых. Предварительно каждой ведомой микросхеме присваивается номер (посредством записи в нее командного слова), который должен быть равен номеру входа IR ведущей микросхемы, с которым соединен вывод INT ведомой микросхемы.

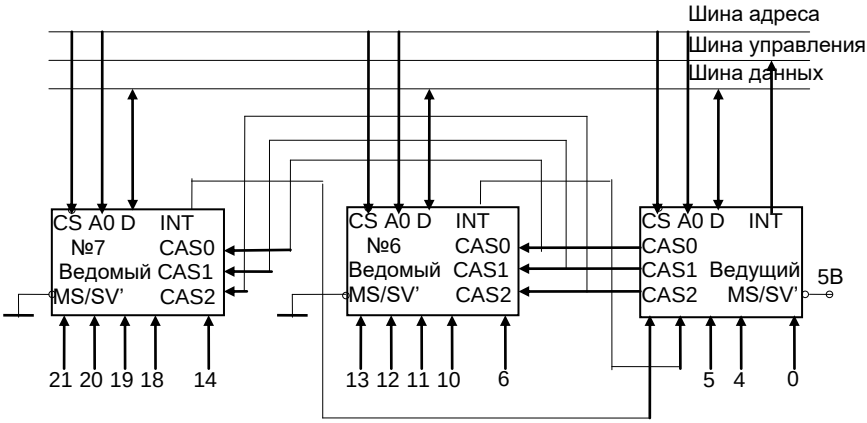


Рис. 7.3. Схема каскадного соединения БИС М1821ВН59А

Если сигнал INT, поступивший на процессор, выработан сигналом IR, поданным на вход ведущей микросхемы, то формирование трехбайтовой команды CALL осуществляется той же БИС. Если же прерывание процессора происходит от сигнала, поступившего на ведомую БИС, то формирование команды CALL происходит следующим образом. При поступлении первого подтверждающего сигнала INTA' ведущая микросхема выдает на шину данных код команды CALL, а на шину CAS0...CAS2 - код номера ведомой БИС. Поэтому, с приходом остальных двух сигналов INTA' код адреса подпрограммы обслуживания выдается на шину данных той ведомой БИС, запрограммированный номер которой совпал с кодом на шине CAS0...CAS2.

7.4. Обслуживание прерываний в микроконтроллере AT89LP51/52

7.4.1. Источники прерываний

AT89LP51/52 имеет 6 источников прерываний: два внешних прерывания, три прерывания по таймеру и прерывание от последовательного порта. Каждое из этих прерываний и сброс системы имеют отдельный программный вектор в начальной области памяти программ. Каждый источник прерывания может быть отдельно включен или отключен путем установки или сброса бита в регистре разрешения прерывания IE (Табл. 7.1). Регистр IE также содержит глобальный бит отключения EA, который запрещает все прерывания.

Табл. 7.1. Регистр разрешения прерываний IE

7	6	5	4	3	2	1	0
EA	-	ET2	ES	ET1	EX1	ET0	EX0

Обозначение	Функция
EA	Глобальное включение/выключение. Все прерывания запрещены, когда EA = 0. Когда EA = 1, каждый источник прерывания включается/выключается путем установки /сброса собственного бита разрешения.
ET2	Разрешение прерывания таймера 2
ES	Разрешение прерывания последовательного порта
ET1	Разрешение прерывания таймера 1
EX1	Разрешение внешнего прерывания 1
ET0	Разрешение прерывания таймера 0
EX0	Разрешение внешнего прерывания 0

Каждый источник прерывания может быть индивидуально запрограммирован на один из четырех уровней приоритета путем установки или очистки битов в регистрах приоритета прерывания IP и IPH. IP содержит младшие биты приоритета, а IPH - старшие биты приоритета для каждого прерывания. Выполняемая процедура обработки прерывания может быть прервана прерыванием с более высоким приоритетом, но не другим прерыванием с таким же или более низким приоритетом.

Прерывание с наивысшим приоритетом не может быть прервано никаким другим источником прерывания. Если в конце инструкции ожидается два запроса с разными уровнями приоритета, обслуживается запрос с более высоким уровнем приоритета. Если запросы того же уровня приоритета находятся в состоянии ожидания, в конце выполнения инструкции

внутренняя процедура опроса определяет, какой запрос обслуживается. Программа опроса основана на адресе вектора: прерывание с меньшим адресом вектора имеет более высокий приоритет, чем прерывание с большим адресом вектора. Следует отметить, что опрос используется только для разрешения ожидающих запросов того же уровня приоритета.

Табл. 7.2. Регистр младших битов приоритетов прерываний IP

7	6	5	4	3	2	1	0
-	-	PT2	PS	PT1	PX1	PT0	PX0

Обозначение	**Функция**
PT2	Младший бит приоритета таймера 2
PS	Младший бит приоритета последовательного порта
PT1	Младший бит приоритета таймера 1
PX1	Младший бит приоритета внешнего прерывания 1
PT0	Младший бит приоритета таймера 0
PX0	Младший бит приоритета внешнего прерывания 0

Табл. 7.3. Регистр старших битов приоритетов прерываний IPH

7	6	5	4	3	2	1	0
-	-	PT2H	PSH	PT1H	PX1H	PT0H	PX0H

Обозначение	**Функция**
PT2H	Старший бит приоритета таймера 2
PSH	Старший бит приоритета последовательного порта
PT1H	Старший бит приоритета таймера 1
PX1H	Старший бит приоритета внешнего прерывания 1
PT0H	Старший бит приоритета таймера 0
PX0H	Старший бит приоритета внешнего прерывания 0

Каждое внешнее прерывание (см. рис. 7.4) INT0’ и INT1’ может активироваться сигналом либо по его уровню, либо по фронту, в зависимости от битов IT0 и IT1 в регистре TCON. Флаги, которые фактически генерируют эти прерывания, это биты IE0 и IE1 в TCON. Когда выбирается программа обслуживания, аппаратное обеспечение сбрасывает флаг, сгенерировавший внешнее прерывание, только если прерывание было активировано фронтом. Если прерывание было активировано по уровню, то внешний источник запроса (а не аппаратное обеспечение на кристалле) управляет флагом запроса.

Прерывания от Таймера 0 и Таймера 1 генерируются флагами TF0 и TF1, которые устанавливаются в соответствующих регистрах Таймера/Счетчика (за исключением Таймера 0 в Режиме 3). Если генерируется прерывание от таймера, встроенное аппаратное обеспечение сбрасывает флаг, сгенерировавший его, когда выполняется служебная процедура его обслуживания.

Прерывание Таймера 2 генерируется логическим ИЛИ битов TF2 и EXF2 в регистре T2CON. Ни один из этих флагов не сбрасывается аппаратно, когда процессор переключается на процедуру обслуживания. Служебная процедура обычно должна определить, генерировал ли прерывание TF2 или EXF2, и этот бит должен быть очищен программным обеспечением.

Прерывание последовательного порта генерируется логическим ИЛИ RI и TI в SCON. Ни один из этих флагов не сбрасывается аппаратно, когда процессор обращается к подпрограмме обслуживания. Процедура обслуживания обычно должна определить, генерировал ли прерывание RI или TI, и этот бит должен быть очищен программным обеспечением.

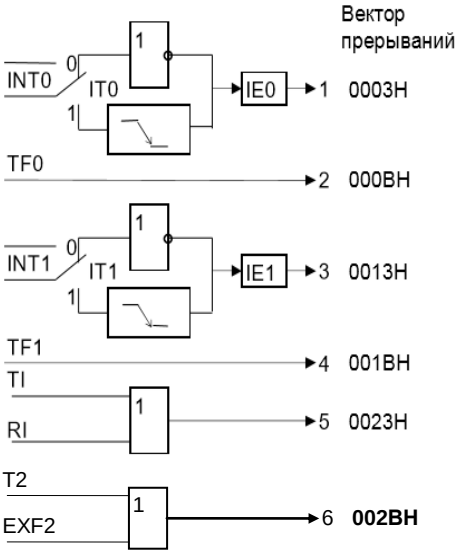


Рис.7.4. Прерывания в микроконтроллере AT89LP51/52

Все биты, которые генерируют прерывания, могут быть установлены или очищены программно с тем же результатом, как если бы они были установлены или очищены аппаратно. То есть прерывания могут быть сге-

нерированы, а ожидающие прерывания могут быть отменены в программном обеспечении.

Табл. 7.1. Адреса векторов прерываний

Прерывание	Источник	Адрес вектора
Сброс системы	RST или POR или BOD	0000H
Внешнее прерывание 0	IE0	0003H
Переполнение Таймер 0	TF0	000BH
Внешнее прерывание 1	IE1	0013H
Переполнение Таймер 1	TF1	001BH
Прерывание последовательного порта	RI или TI	0023H
Прерывание Таймера 2	TF2 или EXF2	002BH

7.4.2. Время реакции на прерывание

Флаги прерывания могут быть установлены аппаратно в любом такте цикла. Контроллер прерываний опрашивает флаги в последнем такте выполняемой инструкции. Если один из флагов был установлен в предыдущем такте, цикл опроса найдет его, и система прерываний сгенерирует код LCALL для вызова соответствующей процедуры обслуживания в качестве следующей инструкции, при условии, что прерывание не заблокировано ни одним из следующих условий:

- прерывание с таким же или более высоким уровнем приоритета уже выполняется;
- выполняется инструкция RETI или любая запись в регистры IE, IP или IPH;
- ЦП в настоящее время принудительно переходит в режим ожидания из-за записи IAP или FDATA.

Каждое из этих условий блокирует генерацию LCALL для процедуры обслуживания прерывания. Второе условие гарантирует, что если выполняемой инструкцией является RETI или какой-либо доступ к IE, IP или IPH, то, по крайней мере, еще одна инструкция будет выполнена до того, как будет обращение к вектору какого-либо прерывания. Цикл опроса повторяется в последнем такте каждой инструкции, а опрашиваемые значения - это значения, которые присутствовали в предыдущем такте. Если активный флаг прерывания не обслуживается из-за одного из вышеуказанных условий и больше не активен после снятия условия блокировки, запрещенное прерывание не будет обслуживаться. Другими словами, тот факт, что флаг прерывания когда-то был активен, но не обслуживался, не запоминается. Каждый цикл опроса новый.

Если запрос активен и выполнены условия для его подтверждения, следующей выполняемой инструкцией будет вызов аппаратной подпрограммы запрошенной процедурой обслуживания. Сам вызов занимает четыре цикла. Таким образом, между активацией запроса на прерывание и началом выполнения первой инструкции служебной программы обслуживания проходит не менее пяти полных тактов. Более длительное время отклика возникает, если запрос заблокирован одним из ранее перечисленных условий.

Если прерывание с таким же или более высоким уровнем приоритета уже выполняется, дополнительное время ожидания зависит от характера процедуры обслуживания другого прерывания. Если выполняемая инструкция не находится в своем последнем такте, дополнительное время ожидания не может превышать 4 такта, поскольку самая длинная инструкция имеет длину 5 тактов. Если выполняется инструкция RETI, дополнительное время ожидания не может превышать 9 тактов (максимум еще 4 такта для завершения выполняемой инструкции плюс максимум 5 тактов для завершения следующей инструкции). Таким образом, в системе с одним прерыванием время отклика всегда больше 5 тактов и меньше 14 тактов. (См. рис. 7.5 и рис. 7.6.)

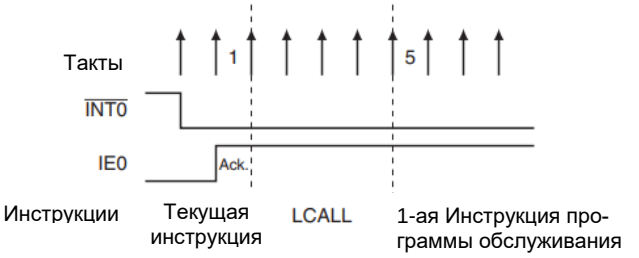


Рис.7.5. Минимальное время реакции прерывания в быстром режиме

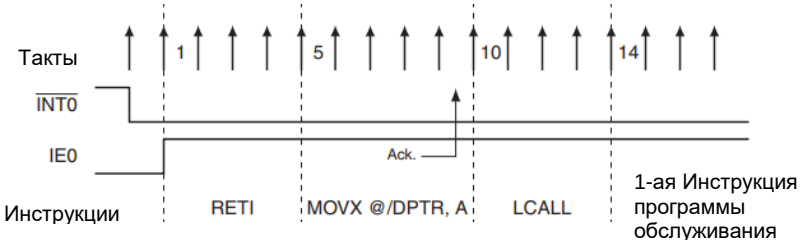


Рис.7.6. Максимальное время реакции прерывания в быстром режиме

Для совмещенного режима время реакции намного больше. (См. рис. 7.7 и рис. 7.8.)

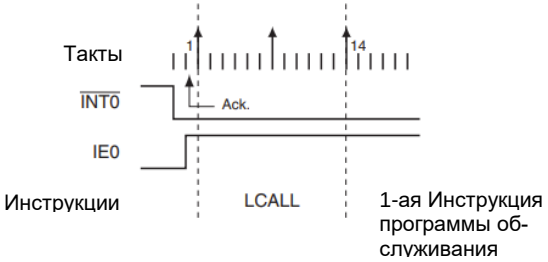


Рис.7.7. Минимальное время реакции прерывания в режиме совместимости

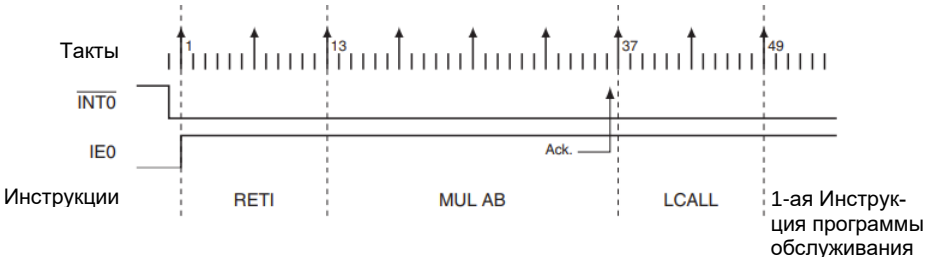


Рис.7.8. Максимальное время реакции прерывания в режиме совместимости