Практическая работа

**Микрокоманды**

Для управления трактом данных необходимо 29 сигналов. Их можно разделить на пять функциональных групп:

* 9 сигналов для записи данных с шины С в регистры;
* 9 сигналов для разрешения передачи регистров на шину В и в АЛУ;
* 8 сигналов для управления АЛУ и схемой сдвига;
* 2 сигнала, которые указывают, что нужно осуществить чтение или запись через регистры MAR/MDR;
* 1 сигнал, который указывает, что нужно осуществить вызов из памяти через регистры PC/MBR.

Значения этих 29 сигналов управления определяют операции для одного цикла тракта данных. Цикл состоит из передачи значений регистров на шину В, прохождения этих сигналов через АЛУ и схему сдвига, передачи полученных результатов на шину С и записи их в нужный регистр (регистры). Кроме того, если установлен сигнал считывания данных, то в конце цикла после загрузки регистра MAR начинает работать память. Данные из памяти помещаются в MBR или MDR в конце следующего цикла, а использоваться эти данные могут в цикле, который идет после него.

Мы загружаем регистр MAR в конце цикла тракта данных и инициируем работу памяти сразу после этого. Следовательно, мы не можем ожидать, что результаты считывания окажутся в регистре MDR в начале следующего цикла, особенно если длительность импульса небольшая. Этого времени будет недостаточно. Поэтому между началом считывания из памяти и использованием полученного результата должен помещаться один цикл. Конечно, во время этого цикла могут выполняться и другие операции - не только те, которым требуется слово из памяти.

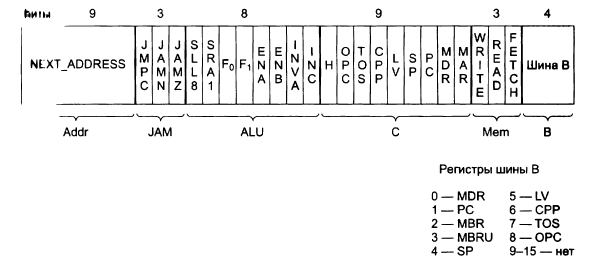
Так как регистры MBR и MDR загружаются на фронте синхронизирующего сигнала вместе с другими регистрами, их можно считывать во время циклов, в течение которых осуществляется передача нового слова из памяти. Они возвращают старые значения, поскольку прошло еще недостаточно времени для того, чтобы они сменились новыми.

Выходной сигнал шины С можно записать сразу в несколько регистров, однако нежелательно передавать значения более одного регистра на шину В. Существуют только 9 входных регистров, которые могут запустить шину В (регистры MBR со знаком и без знака учитываются отдельно). Следовательно, мы можем закодировать информацию для шины В в 4 бита и использовать декодер для порождения 16 сигналов управления, 7 из которых не нужны.

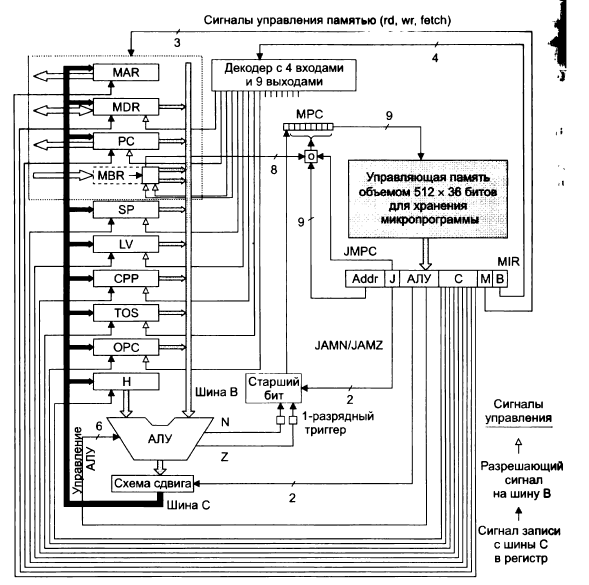
Теперь управление трактом данных происходит с помощью 24 команд, следовательно, нам требуется 24 бита. Однако эти 24 бита управляют трактом данных только в течение одного цикла. Задача управления - определить, что нужно делать в следующем цикле. Чтобы учесть это в конструкции контроллера, создадим формат для описания операций, выполняемых с использованием 24 бит управления и двух дополнительных полей: поле NEXT\_ADDRESS и поле JAM.

Один из возможных форматов (формат микрокоманды для Mic-1). В нем представлены следующие 6 групп, содержащие 36 сигналов:

* Addr - адрес следующей потенциальной микрокоманды;
* JAM - определение того, как выбирается следующая микрокоманда;
* ALU - функции АЛУ и схемы сдвига;
* С - выбор регистров, которые записываются с шины С;
* Меm - функции памяти;
* В - выбор источника для шины.



**Управление микрокомандами – микроархитектура Mic-1**



Для того чтобы понять какой сигнал управления и на каком цикле должен устанавливаться существует контроллер последовательности, который отвечает за последовательность операций, необходимых для выполнения одной команды. В каждом цикле он должен выдавать следующую информацию:

* Состояние каждого сигнала управления в системе;
* Адрес микрокоманды, которая будет выполняться следующей.

Диаграмма полной микроархитектуры Mic-1 состоит из двух частей: тракта данных и блока управления.

Самой большой и важной частью блока управления является управляющая память. Функционально она представляет собой память, в которой вместо обычных команд хранятся микрокоманды. Отличие от основной памяти в том, что команды, хранящиеся в основной памяти, всегда выполняются в порядке адресов, а микрокоманды – нет. Каждая микрокоманда сама указывает на микрокоманду.

Поскольку управляющая память функционально представляет собой ПЗУ, ей нужны собственные адресный регистр (назовем MPC) и регистр данных (назовем MIR). Процесс считывания происходит постоянно, поэтому сигналы чтения и записи не требуются.

Группа ALU содержит 8 бит, которые позволяют выбрать функцию АЛУ и запустить схему сдвига. Биты С загружают отдельные регистры с шины С. Сигналы М управляют работой памяти.

Наконец, последние 4 бита запускают декодер, который определяет, значение какого регистра будет передано на шину В. Задействуем стандартную схему, так как использование стандартных схем упрощает работу, и снижает вероятность ошибки.

Работа схемы заключается в том, что микропрограмме приходится не только управлять трактом данных, но и определять, какая микрокоманда должна выполняться следующей, поскольку микропрограммы не упорядочены в управляющей памяти. Вычисление адреса следующей микрокоманды начинается после загрузки регистра MIR. Сначала в регистр MPC копируется 9-разрядное поле NEXT\_ADDRESS (следующий адрес). Пока происходит копирование, проверяется поле JAM. Когда копирование поля NEXT\_ADDRESS завершится, регистр МРС укажет на следующую микрокоманду.

Принципы синхронизации машины, заключаются в том, что синхронизирующий сигнал делится на подциклы. Во время подцикла 1, который инициируется спадом сигнала, адрес, находящийся в регистре МРС, загружается в регистр MIR. Во время подцикла 2 регистр MIR устанавливает сигналы, и на шину В загружается выбранный регистр. Во время подцикла 3 работают АЛУ и схема сдвига. Во время подцикла 4 стабилизируются значения шины С, шин памяти и АЛУ. Регистр МРС не получает своего значения до тех пор, пока не будут готовы регистры MBR, N и Z, от которых он зависит. На спаде сигнала, когда начинается новый цикл, регистр МРС может обращаться к памяти.

Каждый цикл является самодостаточный. В каждом цикле определяется, значение какого регистра должно поступать на шину В, что должны делать АЛУ и схема сдвига, куда нужно сохранить значение шины С и, наконец, каким должно быть следующее значение регистра МРС.

MPC в действительности вообще не нужен, так как все его входные сигналы можно непосредственно связать с управляющей памятью. MPC может быть реализован в виде виртуального регистра, который представляет собой просто место объединения сигналов.