одним входом и одним выходом в отличие от координатного соединителя, который за один такт выполняет все соединения.

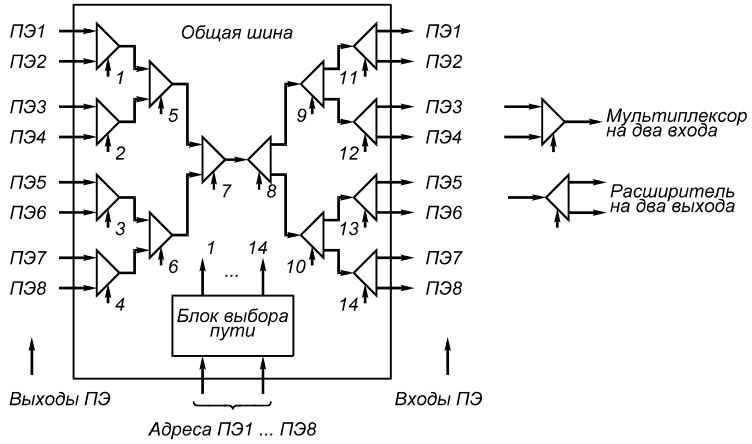

Рис. 2.20. Коммутатор типа “общая шина”

Следует четко разделять два понятия: связь и коммутатор. Связи различного вида возникают между процессорами в ходе выполнения вычислительного алгоритма, а коммутатор (коммутирующая сеть, система коммутации) есть среда, в которой реализуются эти связи. Не всякий коммутатор в состоянии реализовать требуемые виды связей.

Рассмотрим три варианта изображения некоторой многоэлементной связи:

а) с помощью рисунка (рис. 2.21). Это наиболее наглядный вид изображения связей, однако он не может быть использован в программе;

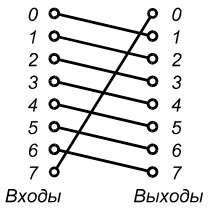

Рис. 2.21. Изображение связей в виде схемы соединений

б) с помощью таблицы (пакета) связей:

Входы	0	1	2	3	4	5	6	7
Выходы	1	2	3	4	5	6	7	0

Это наиболее общий вид изображения связи. Здесь позиция целого числа в таблице означает номер входа, а число в этой позиции — номер выхода, на который замкнута данная связь;

в) с помощью закона, позволяющего на основании номеров (адресов) входов вычислять номера (адреса) выходов.

В зависимости от характера различают регулярные и распределенные случайным образом связи. Для *регулярных связей* таблица связей не нужна, так как все связи вычисляются по определенному, достаточно простому закону. Такие связи называются *перестановками.* Они, как правило, используются для обработки регулярных структур данных при решении систем уравнений, выполнении матричных операций, вычислении быстрого преобразования Фурье и т. д.

Случайно распределенные связи возникают при обработке нечисловой информации (графов, таблиц, текстов) и могут задаваться только в виде таблицы случайных связей.

Рассмотрим некоторые виды перестановок, часто упоминаемые в литературе по параллельным ЭВМ.

1. Перестановки с замещением (рис. 2.22). Закон получения номера выхода следующий:

$$\{b_n, \ldots, b_k, \ldots, b_1\}_j = \{ a_n, \ldots, \overline{a}_k, \ldots, a_1\}_i,$$

где b_k, a_k — k-е двоичные разряды номера i-го входа и j-го выхода связи; $\overline{a}_k$ — двоичная инверсия k-го разряда.

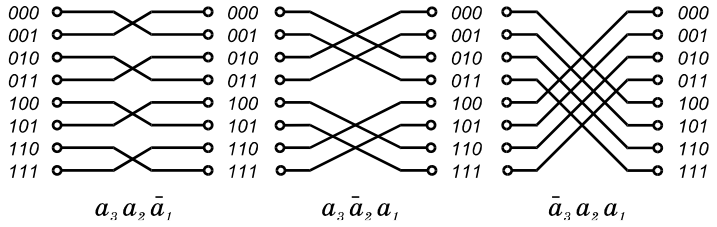

Рис. 2.22. Перестановки с замещением

2. Перестановки с полной тасовкой (рис. 2.23). Они соответствуют циклическому сдвигу влево всех разрядов или группы разрядов номера нужного входа. Полученный двоичный результат дает номер соответствующего выхода.

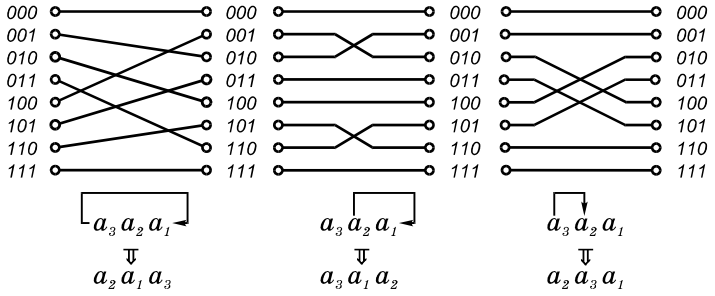

Рис. 2.23. Перестановки с полной тасовкой

3. Перестановки со смещением (рис. 2.24). Они выполняются по формуле

$$i_B = (i_A + r)_N ,$$

где r — величина сдвига; N — число входов. Вычисления производятся по модулю N. Для случая $i_B = (i_A + 1)_4$ (см. рис. 2.24) в разрядном представлении номеров входов и выходов рассматриваются только два младших разряда.

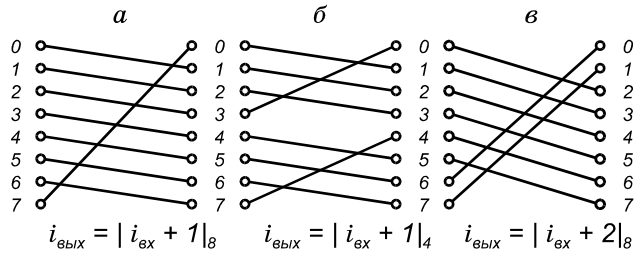

Рис.2.24. Престановки со смещением
величина сдвигов и число входов различны

Если в некотором коммутаторе предусмотрено (конструктивно) использование m перестановок, т. е.

$$КМ = \{П_1, П_2, ..., П_m\},$$

где $П_k$ — перестановка типа k, то для задания кода любой перестановки в КМ для использования необходимо управляющее слово с разрядностью $\log_2 m$.

Как уже говорилось, произвольные парные связи описываются таблицей из N чисел по $\log_2 N$ разрядов на одно число, т. е. требуется всего $N \log_2 N$ бит.

Координатный переключатель в состоянии реализовать за один такт все виды связей как регулярных, так и случайных. Однако объем оборудования такого коммутатора растет слишком быстро.

Поэтому надо искать другие типы коммутаторов, имеющих несколько меньше возможностей по числу устанавливаемых связей, но требующих значительно меньшего объема оборудования. Критерии при поиске таких коммутаторов могут быть разными: минимальное время реализации пакета связей t; минимальный объем оборудования коммутатора H; минимальная величина произведения $t \cdot H$ и др.

Не проводя поиска коммутаторов по указанным критериям, перечислим основные типы коммутаторов и оценим их наиболее важные характеристики.

По структуре (рисунку соединений) коммутаторы можно разделить на две группы: среды и многокаскадные коммутаторы.

Среда — коммутатор, каждый узел которого связан только со своими ближайшими соседями.

Однокаскадные коммутаторы, которые спроектированы для выполнения значительного числа перестановок, имеют большой объем оборудования. Поэтому часто строятся более простые однокаскадные коммутаторы, а перестановки реализуются путем итераций во времени, т.е. за несколько тактов.

В коммутаторе с односторонним сдвигом (рис. 2.25, а) каждый ПЭ связан с двумя соседними. Этот коммутатор строго соответствует перестановке, показанной на рис. 2.24, а. Однако, чтобы выполнить перестановку, как на рис. 2.24, в, требуется уже два такта работы такого коммутатора. В общем случае коммутатор (см. рис. 2.25, а) в состоянии выполнить за N тактов любую перестановку, где N — число ПЭ. При этом каждую единицу передаваемой информации можно снабдить номером процессора прием

ника. По мере сдвигов, когда информация достигнет соответствующего ПЭ, она должна “осесть” в этом ПЭ.

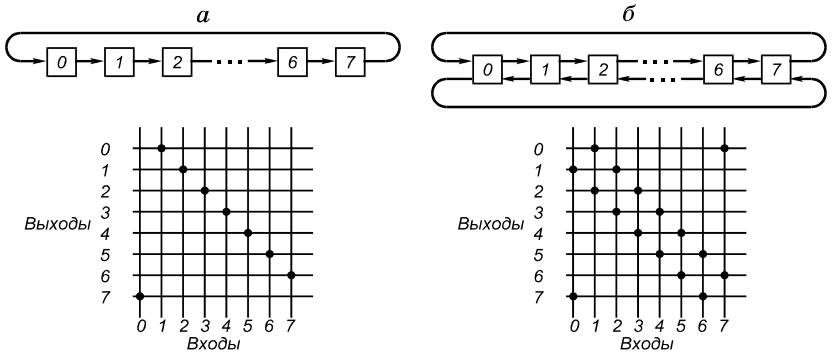

Рис.2.25. Структура кольцевых коммутаторов

Коммутатор с односторонним сдвигом (см. рис. 2.25, а) представлен в сетке координатного переключателя. Точки на пересечениях означают, что все эти соединения могут осуществляться одновременно. Такое представление коммутаторов удобно для теоретических исследований. По данному представлению видно, что коммутатор с односторонним сдвигом — часть полного координатного переключателя и поэтому не является соединительной сетью.

Любая совокупность перестановок в коммутаторе с двусторонним сдвигом (рис. 2.25, б) будет выполнена за более короткое время, чем в коммутаторе на рис. 2.25, а. В частности, чтобы передать единицу информации по связи ПЭ0 $\rightarrow$ ПЭ7, в первом коммутаторе требуется семь тактов, во втором — один такт.

В двумерном коммутаторе каждый ПЭ соединен двусторонними связями с четырьмя соседними (справа, слева, снизу, сверху) (рис. 2.26, а). Внешние связи могут в зависимости от решаемой задачи программно замыкаться по-разному: вытягиваться в цепочку (рис. 2.26, б), замыкаться в виде вертикально (рис. 2.26, в) и горизонтально (рис. 2.26, г) расположенных цилиндров, в виде тороида (рис. 2.26, д). Коммутационный элемент (КЭ) здесь обычно соединен с оборудованием ПЭ, и объем оборудования всего коммутатора растет пропорционально числу процессоров *N*.

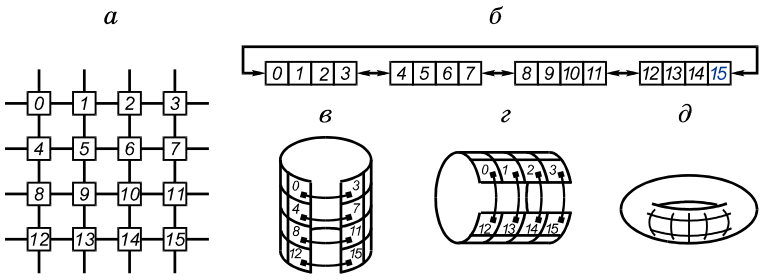

Рис. 2.26. Коммутатор с матричной схемой соединения

Коммутатор с матричной схемой соединений выполняет некоторый набор перестановок быстрее, чем рассмотренные выше коммутаторы с кольцевыми схемами.

Быстродействие любого коммутатора во многом определяется максимальным расстоянием *D,* под которым понимается число промежуточных узлов или тактов передачи информации между самыми удаленными процессорами.

Под кубическим коммутатором понимается пространственная трехмерная структура, где каждый процессор соединен с шестью соседними: четырьмя в своей плоскости и двумя в прилегающих плоскостях.

В общем случае все среды можно назвать *n*-мерными кубами, тогда кольцевой коммутатор имеет размерность $n = 1$, а матрица — $n = 2$ и т. д. Большое распространение в параллельной вычислительной технике получили среды с размерностями 2 и 3.

Чем больше *n* при неизменном числе процессоров, тем больше радиус связей и тем меньше *D.* Под радиусом связи понимается число процессоров, напрямую связанных с данным ПЭ.

Величина *n* может быть и больше трех. Покажем общий прием построения *n*-кубов произвольной размерности (рис. 2.27).

Для примера рассмотрим возможные связи элемента 9 матрицы, изображенной на рис. 2.26, а. Этот элемент в своей строке связан с элементами 8 и 10, адреса которых отличаются

на -1 и +1. Если бы данная строка составляла независимый коммутатор с размерностью 4 (n = 1, N = 4), то связи любого ПЭi такого коммутатора описывались бы рангом 1. Если элемент 9 находится в матрице, то его связи с элементом 5 верхней строки и элементом 13 нижней строки отличаются на $+N$ и $-N$, где N = 4 — размер строки. Следовательно, связи элемента ПЭi матричного коммутатора описываются рангом 2 (см. рис. 2.27).

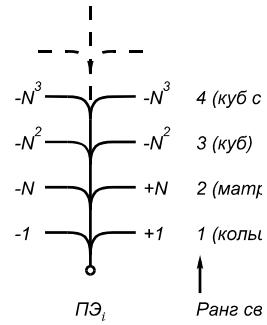

Рис. 2.27. Способ построения

Если бы элемент 9 находился в кубе, то номера смежных элементов соседних плоскостей отличались бы на $\pm N^2$, в кубе степени 4 — на $\pm N^3$ и т. д., что и показано на рис. 2.27. Такая методика позволяет построить куб любой степени и вычислить величину D.

Многокаскадные коммутаторы помогают создать более дешевые варианты соединительных и обобщенных соединительных сетей. Координатный переключатель $N \times N$ можно свести к двум ($N/2 \times N/2$) переключателям с замещением (рис. 2.28, а) [2]. В свою очередь переключатели $N/2 \times N/2$ могут быть разложены на более мелкие подобным же образом. Пример сети, приведенной до уровня стандартных КЭ размерностью 2×2,

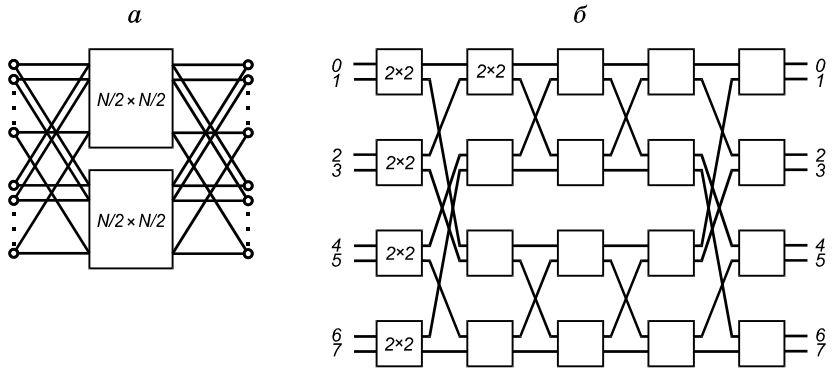

Рис. 2.28. Способ построения сети Бенеша

дан на рис. 2.28, б. По определению, это соединительная сеть (сеть Бенеша), обеспечивающая $N!$ возможных перестановок, каждая из которых выполняется за один такт работы сети. В такой сети отсутствуют конфликты для любых парных соединений.

Стандартный КЭ размерностью 2×2 и возможные в нем соединения показаны на рис. 2.29. Иногда для построения больших сетей применяются элементы 4×4 или 8×8. Основной недостаток сети Бенеша — большое количество элементов, необходимых для ее построения.

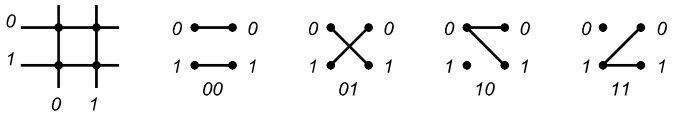

Рис. 2.29. Коммутационный элемент размерностью 2×2
и допустимые перестановки.
Внизу приведены коды нумерации перестановок

Вследствие этого на практике используются коммутаторы со значительно меньшим объемом оборудования, не являющиеся соединительными сетями в полном смысле, однако обеспечивающие широкий класс перестановок. В таких коммутаторах возможны конфликты, которые разрешаются последовательно во времени. Примером конфликта может быть ситуация, когда для двух входов требуется одна и та же выходная линия.

Широко используются неполные соединительные сети: омега-сеть, сеть Бенеша, *R*-сеть и др. Омега-сеть относится к типу сетей тасовки с замещением.

Коммутационные сети могут работать в двух режимах: коммутации каналов и коммутации сообщений. В первом случае сначала с помощью адресной системы устанавливается прямой тракт между парой соединяемых точек, а затем по этому тракту передается информация. Такой вид связи используется, например, в телефонных сетях. Во втором случае каждая передаваемая единица информации снабжается адресом требуемого выхода и перемещается от узла n_i к узлу m_{i+1}, где i — номер каскада; n и m — номера коммутационных узлов в каскадах. После того как информация принята узлом m_{i+1}, линия, соединявшая n_i и m_{i+1}, освобождается

для других соединений. Узел m_{i+1} по адресной части сообщения определяет дальнейший маршрут сообщения.

В матрицах процессоров используется как коммутация каналов, так и коммутация сообщений. Оценим основные характеристики коммутаторов, описанных выше. Оценка будет производиться для случая коммутации сообщений и перестановок двух видов: регулярных и случайных. Существует множество вариантов регулярных перестановок, и довольно трудно аналитически или методом имитационного моделирования определить число тактов, необходимое для реализации некоторой тестовой группы перестановок. Поэтому в качестве теста принята единственная, но очень широко используемая в процессорных матрицах операция — сдвиг на некоторое число разрядов. Минимальная дистанция сдвига равна единице, а максимальная совпадает с величиной максимального межпроцессорного расстояния D, в связи с чем за среднюю величину сдвига принято $D/2$ (см. табл. 2.2). Конечно, это только качественная оценка скорости выполнения регулярных операций коммутации.

Таблица 2.2.

Основные характеристики коммутаторов различных типов

Тип коммутатора	Время реализации пакета, такты		Число КЭ
	Регулярные перестановки	Случайный пакет	
Общая шина	$N/2$	N	$\frac{N}{2}\log_2 N$
Кольцо (ранг 1)	$N/4$	$N/2$	N
Матрица (ранг 2)	$\sqrt{N}/2$	$\sqrt{N}$	N
Куб (ранг 3)	$3\sqrt[3]{N}/4$	$\sqrt[3]{N}$	N
Омега-сеть	1	$\log_2 N$	$\frac{N}{2}\log_2 N$
Полный координатный переключатель	1	1	N^2

Для оценки времени реализации пакетов случайных связей обычно проводят имитационное моделирование работы коммутатора при подаче на вход наборов, распределенных по равномерному закону. При этом конфликты разрешаются последовательно в соответствии с некоторым приоритетом. Результаты такого моделирования и число КЭ в коммутаторе каждого типа приведены в табл. 2.2.

Сделаем следующие выводы:

1) коммутаторы типа полного координатного переключателя не могут использоваться для больших N из-за значительного объема оборудования, а общая шина — из-за большого времени реализации сообщения;

2) порядок быстродействия сред на регулярных и нерегулярных пакетах связей одинаков и равен $\sqrt[n]{N}$, где n — ранг среды, а для каскадных коммутаторов подчиняется логарифмическому закону;

3) омега-сеть превосходит в большинстве случаев среды на регулярных пакетах. Для качественного сравнения сред и многокаскадных коммутаторов построен график (рис. 2.30), из которого следует, что выбор того или иного типа коммутатора должен производиться строго в зависимости от типа ЭВМ, класса решаемых

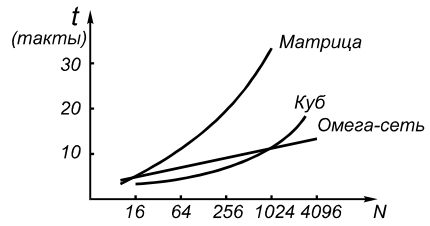

Рис. 2.30. Время реализации случайного пакета связей для различных комутаторов

задач (виды перестановок) и размера ПП.

Рассмотрим некоторые вопросы управления в коммутационных сетях. Для примера разберем управление в омега-сети для двух случаев: реализации регулярных перестановок и пакетов случайных связей.

Коммутатор для реализации регулярных перестановок (рис. 2.31, а) состоит из коммутационного поля и блока дешифрации и управления (ДУ). Выходы блока ДУ с номерами 0...11 подключены к соответствующим адресным входам коммутационных элементов (чтобы не загромождать рисунок, эти линии не показаны).

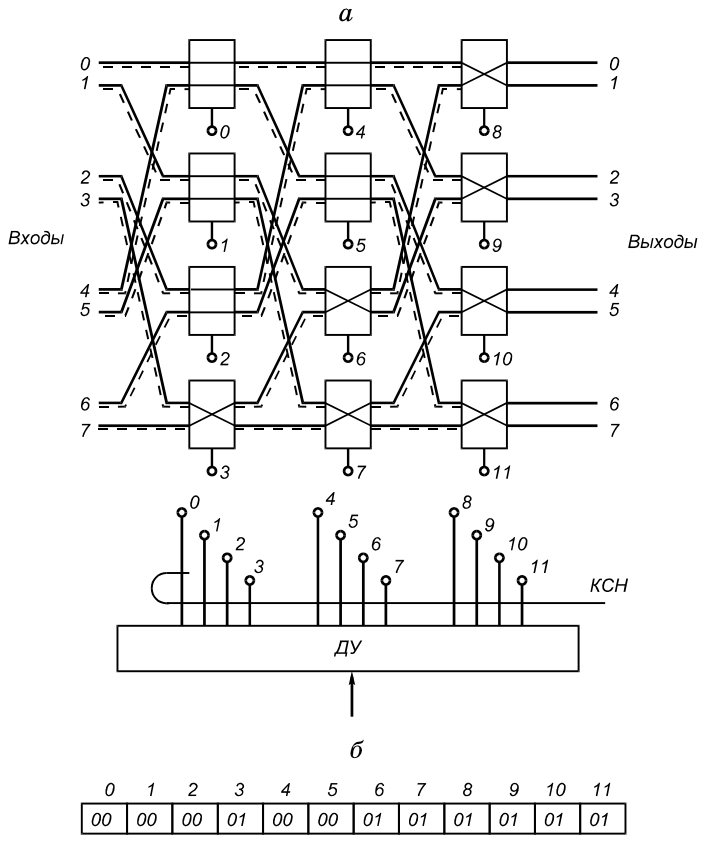

Рис. 2.31. Структура коммутатора для выполнения регулярных перестановок:
а — схема коммутатора; *б* — кодовое слово настройки, использованной в примере перестановки

При выполнении команды коммутации код коммутации подается на вход блока ДУ. Размерность кодового номера перестановки

невелика, зависит от числа реализуемых перестановок m и равна $\log_2 m$. Блок ДУ на основе кодового номера перестановки вырабатывает кодовое слово настройки (КСН), размер которого равен $(N / p)C_p^p \log_p N$, где N — число процессоров; p — размерность КЭ; N/p — число КЭ в одном каскаде; C_p^p — число перестановок в КЭ размерностью $p \times p$; $\log_p N$ — число каскадов. Например, КСН для коммутатора на 128 ПЭ при $p = 2$ содержит 1792 разряда, что сильно затрудняет создание коммутаторов с большой размерностью. Кодовые слова настройки могут храниться в памяти блока ДУ либо вычисляться при выполнении команды коммутации. В первом случае требуется большой объем памяти, во втором время вычисления будет слишком велико. Все КЭ (см. рис. 2.31, а) замкнуты в соответствии с КСН (см. рис. 2.31, б), которое обеспечивает смещение данных на один ПЭ (см. рис. 2.24).

После настройки коммутационного поля с помощью блока ДУ производится передача данных за один такт. Такой вид функционирования соответствует коммутации каналов.

Если при реализации некоторой перестановки блок ДУ обнаруживает, что в некоторых КЭ возможны конфликты, блок ДУ вырабатывает два или более КСН, которые реализуются в коммутационном поле последовательно.

Как правило, вид перестановок заранее не известен и генерируется в процессе решения задачи. Так происходит, например, при определении кратчайшего пути в графе, если каждая вершина графа расположена в отдельном ПЭ и представлена в виде списка вершин. Тогда выборка очередного слоя из списка связей (по одной из каждого процессора) приводит к образованию на входе коммутатора случайно распределенного пакета связей.

В этом случае каждая единица информации, передаваемая любым процессором, состоит из двух частей: А, И. Здесь А — адрес (номер) ПЭ, куда надо передать информацию И. Адрес может передаваться по отдельным шинам (штриховые линии на рис. 2.31, а) или по одним и тем же шинам с разделением во времени: сначала адрес, затем информация. В каждом ПЭ принятый адрес анализируется, чтобы определить на какой выход КЭ должна быть дальше передана информация. Таким образом, в КЭ дешифратор

должен дополнительно выполнять функцию анализа принятого адреса. Кроме того, для случайных пакетов конфликты являются обычной ситуацией, поэтому коммутационный элемент должен иметь память для хранения принятой информации, так как неизвестно, будет принятая информация передана в настоящем такте или послана с задержкой. В таком коммутаторе некоторые связи из случайного пакета могут быть реализованы раньше, чем другие, хотя процесс коммутации на входе был начат для всех связей одновременно.

Описанный способ функционирования соответствует коммутации сообщений. Здесь управление является децентрализованным и распределено по коммутационным элементам.

§ 2.4. Процессорные матрицы

Понятие “процессорная матрица” подчеркивает тот факт, что параллельные операции выполняются группой одинаковых ПЭ, объединенных коммутационной сетью и управляемых единым ЦУУ, реализующим единственную программу. Следовательно, ПМ являются представителями класса ОКМД-ЭВМ. Процессорная матрица может быть присоединена в качестве сопроцессора к БМ. В данном случае БМ обеспечивает ввод-вывод данных, управление графиком работы ПМ, трансляцию, редактирование программ и некоторые другие вспомогательные операции. В отдельных случаях все эти функции выполняет ЦУУ. Процессорные матрицы описаны в [2, 7].

Каждый ПЭ может иметь или не иметь собственную (локальную) память. Известно, что более половины обращений за информацией приходится на долю локальной памяти, поэтому в таких системах к быстродействию коммутатора предъявляются относительно низкие требования. Схема ПМ с локальной памятью приведена на рис. 2.32. Центральное устройство

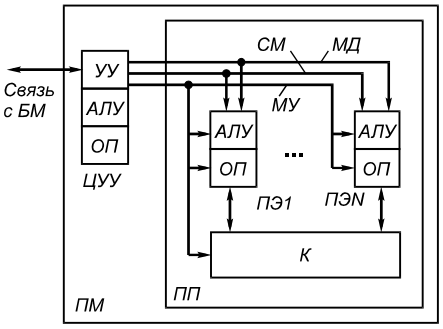

Рис. 2.32 Схема ПМ

управления является полноценным процессором для выполнения единственной программы, однако в этой программе используются команды двух типов: скалярные и векторные. Скалярные команды начинаются и заканчиваются в ЦУУ, векторные команды начинаются в ЦУУ, а продолжаются в ПП.

Процессорное поле содержит ряд ПЭ, каждый из которых является частью полного процессора, т. е. содержит АЛУ и память, но функции УУ в ПЭ весьма ограничены. В большинстве случаев управление заключается в том, что отдельный ПЭ может выключаться в связи с требованиями решаемой задачи на время выполнения одной или нескольких команд. Число ПЭ колеблется от нескольких десятков (в этом случае процессоры делаются мощными) до нескольких сотен и даже тысяч штук (в этом случае из-за ограничений объема оборудования можно использовать только маломощные в вычислительном отношении процессоры).

Центральное устройство управления связано с ПП тремя магистралями:

1) по магистрали данных МД из ЦУУ в ПП передаются общие адреса и данные, а из отдельных ПЭ в ЦУУ проводится выборка необходимых единиц информации;

2) по системной магистрали СМ, которая имеет одну шину на каждый ПЭ, в ПП передается (или из ПП считывается) вектор активности процессорных элементов;

3) по магистрали управления МУ из ЦУУ всем ПЭ передаются одни и те же микрокоманды для одновременного исполнения.

Как в ЦУУ, так и в ПЭ имеются регистры РОН, поэтому форматы и типы скалярных и векторных команд такие же, как и в последовательных ЭВМ: регистр — регистр (RR), регистр — память (RX) и др.

Векторные команды можно разделить на пять основных групп: арифметико-логические команды; команды пересылки между ЦУУ и ПП; команды коммутации; команды групповых условных переходов; команды активации процессорных элементов.

Рассмотрим функционирование ПП при выполнении команд этих групп.

Пусть имеется следующий фрагмент программы:

```
1   S  ЧТ          5, A1
```

2 V ПС 5, 3
3 V ЧТ 2, А2
4 V УМН 2, 3

Здесь S и V — скалярная и векторная команды соответственно.

По команде 1 в ЦУУ из локальной ОП (адрес А1) считывается константа Z1 и записывается в регистр 5. По команде 2 константа *Z*1 из регистра 5 ЦУУ передается в регистр 3 всех ПЭ по МД. По команде 3 все ПЭ считывают из локальной памяти (адрес А 2) константу Z 2 и помещают в регистр 2. Наконец, по команде 4 все ПЭ умножают содержимое регистров 2 и 3 и результат помещают в регистр 2, т.е. выполняется операция умножения на константу, которую на фортраноподобном векторном языке можно записать так:

$Z(*) = Z1 * Z2(*)$

Рассмотрим подробнее выполнение векторной команды 3 на микропрограммном уровне (табл. 2.3).

Таблица 2.3.

Выполнение векторной команды обращения к локальной памяти по тактам

Наименование микрооперации	Место выполнения микро-операции	Используемые	
		магистра-ли	блоки ПЭ*i*
Считывание команды из ОП	ЦУУ	-	-
Дешифрация полей команды в УУ ЦУУ	ЦУУ	-	-
Модификация адреса памяти	ЦУУ	-	-
Установка ОП*i* на прием адреса	ПП	МУ	ОП*i*
Прием адреса А1 из ЦУУ в ОП*i*	ПП	МУ, МД	ОП*i*
Запуск ОП*i* на чтение	ПП	МУ	ОП*i*
Установка АЛУ*i* на прием в регистр 2	ПП	МУ	АЛУ*i*
Операция в АЛУ*i*	ПП	МУ	АЛУ*i*
Возврат к чтению команды из ОП	ЦУУ	-	-

Из таблицы следует, что все микрокоманды генерируются в УУ ЦУУ, однако для команды векторного типа некоторые микро

операции выполняются не в ЦУУ, а в ПП (микрокоманды 4...8). При этом управление производится выборочно различными блоками ПЭ*i* и (как будет далее показано) коммутатором. В ПП все ПЭ*i* выполняют одну и ту же микрооперацию. Рассмотрение этой микропрограммы позволяет оценить разрядность МУ и МД.

Для большинства ЭВМ магистраль МД имеет 32...64 разряда, а магистраль МУ — 30...80 разрядов; разрядность магистрали СМ равна числу ПЭ.

Как и для конвейерных ЭВМ, в ЦУУ для повышения быстродействия обычно применяется совмещение операций. Например, при выполнении микрокоманд 4...8 для команды *Kj* производится выполнение микрокоманд 1...3 для следующей команды *K* (*j* + 1).

Совмещение может быть применено и в процессорных элементах.

В ПМ *индивидуальное управление* каждым ПЭ зависит от выключения этих ПВ на период выполнения одной (нескольких) задачи или команды в зависимости от исходных данных, внутреннего состояния ПЭ и некоторых внешних условий. Таким включением и выключением занимается двухуровневая система активации.

Первый уровень активации — реконфигурация — позволяет отключать ненужные или неисправные ПЭ на период выполнения нескольких задач и дает возможность проводить вычисления при наличии неисправных ПЭ. Триггер конфигурации имеется в каждом ПЭ. После выполнения тестовых программ процессорного поля в триггеры конфигурации исправных ПЭ из ЦУУ по СМ заносятся единицы, а в триггеры неисправных ПЭ — нули.

Процессорные элементы с нулевым значением триггера конфигурации не участвуют в вычислениях: в них не выполняются команды, поступающие из ЦУУ, становится невозможным считывание и запись в ОП таких ПЭ, состояние их триггеров активации не влияет на команды векторного условного перехода.

Кроме триггера конфигурации в каждом ПЭ есть несколько (до 16) триггеров активации. Триггер основной активности А0 управляет активностью ПЭ: если А0 находится в состоянии 1, то данный ПЭ активен, если в состоянии 0 — пассивен. Вспомогательные триггеры А1...А7 запоминают активность ПЭ перед различными внутренними циклами, если эта активность меняется; после выхода из цикла нужно восстановить исходное значение

активности. Триггеры А8...А15 отражают результат вычислений, проводимых в ПЭ: А8 — равенство результата нулю; А9 — знак результата; А10 — перенос; А11 — арифметическое переполнение; А12 — исчезновение порядка; А13 — переполнение порядка; А14 — некорректное деление; А15 — служебный триггер.

Активация бывает внешней и внутренней. В первом случае состояние триггеров активности каждого ПЭ устанавливается со стороны ЦУУ по СМ с помощью команд вида V ВША Ri, Aj. Здесь код ВША означает, что выполняется команда внешней активации, по которой содержимое 16 разрядов регистра Ri ЦУУ поразрядно записывается во всех ПЭ в регистры активации с номером j. Если j = 0, то это означает, что производится загрузка триггеров основной активности, которая может привести к выключению некоторых ПЭ, когда в соответствующих разрядах находятся нули.

Команда V ЧТА Ri, Aj выполняет обратную операцию: считывание состояния триггеров активации в разряды регистра Ri ЦУУ.

Коммутатор в ПП (см. рис. 2.32) может управляться командами двух типов:

V КМ1 Ri, Rj, Rk, F
V КМ2 Ri, Пk, F (2.3)

Команда коммутации КМ1 передает содержимое регистров Ri всех ПЭ в регистры Rk других ПЭ в соответствии с вектором адресов (номеров ПЭ-приемников), каждый элемент которого расположен в регистре Rj данного ПЭ. При выполнении команды содержимое Ri и Rj передается в коммутатор, прокладывающий по адресу из Rj нужный маршрут и затем передающий информацию из Ri.

Детальный алгоритм выполнения команды коммутации зависит от типа используемого коммутатора.

Функция F в команде КМ1 указывает на разновидность команды: запись или считывание данных. В случае считывания требуется обеспечить еще и обратную передачу информации.

Здесь описана только передача информации между регистрами. Если требуется произвести обмен данными, хранящимися в локальной памяти, то предварительно необходимые данные должны быть с помощью команд обращения в память приняты в нужные регистры.

Как упоминалось в § 2.3, команды коммутации с адресным вектором используются там, где заранее не известен вид перестановки, т. е. при обработке сложных структур данных (графов, деревьев, таблиц и т. д.).

Однако не всякий коммутатор пригоден для эффективной реализации команд типа КМ1. Такой коммутатор должен допускать работу в асинхронном режиме, внутренними средствами разрешать конфликты, если они возникают. Время передачи пакета случайных связей должно находиться в допустимых пределах. Назовем подобный коммутатор универсальным (УК). Для небольших размеров ПП ($N = 16...32$) в качестве УК может использоваться полный координатный переключатель, а для $N > 32$ должны использоваться, как было показано в § 2.3, многокаскадные коммутаторы.

В УК изменение конфигурации путем разделения ПП на независимые части или по причине неисправностей процессоров не вызывает трудностей, так как для продолжения решения задачи необходимо только переустановить адресные таблицы. Это обеспечивает повышенную живучесть ПП. В машинах с реконфигурацией число ПЭ задается как параметр. Это означает, что в библиотеках стандартных программ число процессоров не определено и конкретизируется в процессе трансляции пользователем или ОС. В синхронных же сдвиговых коммутаторах типа двумерной среды в ILLIAC-IV выход из строя одного узла разрушает всю систему сдвигов.

При выполнении команд типа КМ1 возможна ситуация, когда несколько ПЭ обращаются к одному, т.е. производится сбор информации (коллекция). В коллекционных командах коммутатор должен самостоятельно обеспечить последовательную запись или выдачу требуемых данных.

Команда типа КМ2 означает, что на коммутатор из ЦУУ через МД передается код перестановки Пk, в соответствии с которым настраивается коммутатор и производится бесконфликтная передача информации из регистров Ri или считывание через коммутатор в регистры Ri. Команда КМ2 может выполняться как на универсальных, так и на синхронных коммутаторах для всех видов регулярных перестановок (см. § 2.3).

Рассмотрим вопросы, связанные с эффективностью вычислений и методами программирования для ПМ, на примерах программирования некоторых операций.

Эффективность вычислений в первую очередь зависит от затрат времени на коммутацию и выборку информации. Время выполнения команды коммутации сравнимо со временем выполнения простых команд (логические команды, целочисленное сложение и т.д.), поэтому в большинстве случаев не коммутация, а *размещение данных* влияет на время выполнения программы.

Рассмотрим два варианта размещения квадратной матрицы в ПП (рис. 2.33). Предположим, что для решения некоторой задачи, например, умножения матриц, требуется параллельная выборка как строк, так и столбцов. Стандартное, принятое для большинства языков и ЭВМ размещение “по столбцам” (см. рис. 2.33, а) позволяет выбирать в АЛУ из ОП параллельно только строки, а элементы столбца расположены в одной ОП*i* и могут выбираться только последовательно. В результате при таком размещении параллелизм процессорного поля не будет использован в полной мере, поэтому для подобных задач характерно специальное размещение информации. При “скошенном” размещении матрицы (см. рис. 2. 33, б) за один такт могут быть выбраны как строка, так и столбец. Но при выборе столбца должна использоваться “фигурная” выборка, когда каждый ПЭi обращается в ОПi по собственному адресу, вычисляемому в зависимости от номера ПЭ.

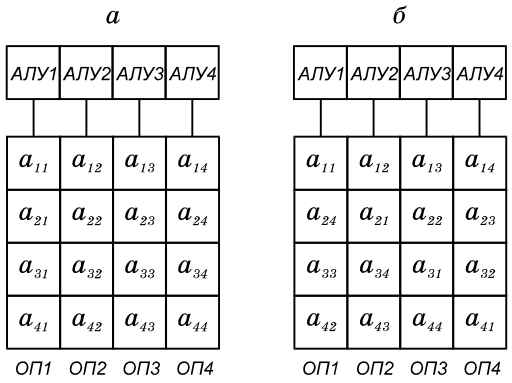

Рис. 2.33. Размещение матрицы в памяти процессорного по

Размещение информации влияет и на методы программирования. Рассмотрим операцию сложения двух векторов (рис. 2.34) для $L = N$ и $L > N$, где L — длина вектора; N — число процессоров. Про

граммы будут выглядеть различно:

а

Ячейка	ОП1	ОП2	ОП3	ОП4
1	a_1	a_2	a_3	a_4
2	b_1	b_2	b_3	b_4
3	c_1	c_2	c_3	c_4
4	—	—	—	—
5	—	—	—	—
6	—	—	—	—
7	—	—	—	—

б

	ОП1	ОП2	ОП3	ОП4
1	a_1	a_2	a_3	a_4
2	a_5	a_6	—	—
3	b_1	b_2	b_3	b_4
4	b_5	b_6	—	—
5	c_1	c_2	c_3	c_4
6	c_5	c_6	—	—
7	—	—	—	—

Рис. 2.34. Многослойное размещение коротких (а) и длинных (б) векторов

а) V ЧТ 1, $Я_1$ (считывание среза ячеек $Я_1$ памяти в регистры 1)

V СЛ 1, $Я_2$ (сложение содержимого регистров 1 со срезом ячеек $Я_2$ памяти)

V ЗП 1, $Я_2$ (запись содержимого регистров 1 в срез ячеек $Я_3$ памяти)

б) V ЧТ 1, $Я_1$

V СЛ 1, $Я_3$

V ЗП 1, $Я_5$

V ВША 5, А0 (из регистра 5 ЦУУ передан код активации 1100, что запрещает обращение к ОП3, ОП4)

V ЧТ 1, $Я_2$

V СЛ 1, $Я_4$

V ЗП 1, $Я_6$

Таким образом, несоответствие размеров L и N меняет не только длину, но и состав операторов программы. На программах на параллельных ЯВУ это не отражается: обе программы будут представлены в виде одного и того же оператора

C (∗) = A (∗) + B (∗)

Поскольку наиболее эффективные программы могут быть написаны только на ассемблере, то в языках подобного уровня следует предусматривать средства, позволяющие вне зависимости

от соотношения L и N получать одинаковые программы. Так как это не всегда возможно, то из-за необходимости учитывать расположение данных программирование на ассемблере для ПМ становится намного сложнее, чем для последовательных ЭВМ.

Некоторые операции в ПМ программируются проще и выполняются быстрее, чем в параллельных ЭВМ других типов. Например, операция сборки

V = V0 (INDEX (i)), i = 1, 2, ..., n

состоит в том, что из вектора V0 выбираются числа в порядке, указанном в целочисленном массиве INDEX(i) и помещаются в вектор V1. В конвейерной ЭВМ (см. § 2.2) данная операция выполняется в скалярной части (и поэтому медленно). В ПМ операция сборки отображается в одну команду коммутации согласно (2.3):

КМ1 Ri, Rj, Rk, F

Здесь в регистре Ri хранится вектор V0, в регистре Rj — массив INDEX(i), в регистр Rk будет записан вектор V1. Поскольку все операции в команде КМ1 выполняются параллельно, то время выполнения этой команды в простейшем случае ($L = N$) будет соответствовать одному циклу работы коммутатора.

Для многих параллельных ЭВМ принципиальное значение имеет команда SUM V, т. е. операция суммирования элементов некоторого вектора V. В процессорных матрицах с УК она выполняется в течение нескольких микротактов, которые строятся на основе команды КМ1. В каждом микротакте реализуется следующая операция (работают все ПЭi):

V СЛ Ri, Rj

Здесь операция СЛ выполняется в каждом ПЭl и производит сложение двух чисел, одно из которых расположено в регистре Ri данного ПЭk, а другое — в регистре Ri, но в ПЭl. В регистре Rj процессора ПЭk указан номер процессора ПЭl. Если в регистре Rj процессора ПЭk записан нуль, то этот процессор не производит суммирование, а выборка операнда из него через коммутатор не запрещается.

Операция SUM V для $N = 8$ (рис. 2.35) выполняется за три такта ($\log_2 8 = 3$) и для каждого такта по вертикали указано содержимое адресного регистра Rj для любого ПЭ.

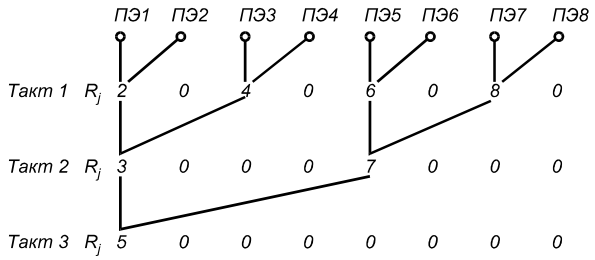

Рис. 2.35. Выполнение операции SUM V в ПМ

Для ПМ довольно сложным является выполнение циклов с оператором IF внутри цикла. В примере

```
      DO 5 I = 1, L
      IF (X (I) - R) 1, 2, 3
1     Y (I) = f1 (X (I))
      GO TO 5
2     Y(I) = f2 (X (I))                                    (2.4)
      GO TO 5
3     Y(I) = f3 (X (I))
5     CONTINUE
      STOP
```

как X, так и Y являются векторами, но Y вычисляется по трем различным формулам в зависимости от значений X(I) и R.

Следовательно, если даже длина векторов X и Y равна числу процессоров, т. е. $L = N$, то и тогда все три части вектора Y нельзя вычислять одновременно в ПП, так как в нем имеется только одно ЦУУ, а для одновременного вычисления всех элементов Y в (2.4) требуется три ЦУУ. Наиболее простой выход из положения заключается в том, что участки $f_1(X(I))$, $f_2(X(I))$, $f_3(X(I))$ вычисляются последовательно в единственном ПП.

Этот метод будет эффективным только в том случае, если функция внутри оператора проста и разбивает всю длину вектора

L на сплошные, примыкающие друг к другу участки, длина которых $l_1, l_2, l_3 >> N$.

Наиболее известным представителем процессорных матриц является ЭВМ ILLIAC-IV [2].

Контрольные вопросы.

1. Определите сущность конвейера. Что такое конвейер команд и арифметический конвейер?
2. Перечислите и поясните методы ускорения обращения к памяти.
3. Какое влияние на структуру и характеристики ЭВМ оказало введение в состав машины регистров общего назначения?
4. Что такое КЭШ, какова его организация?
5. Охарактеризуйте структуру скалярного конвейерного процессора.
6. Объясните, как выполняется программа в скалярном конвейерном процессоре.
7. Охарактеризуйте структуру векторного конвейерного процессора. Какова роль векторных регистров?
8. Назовите факторы, влияющие на быстродействие векторного конвейерного процессора.
9. Опишите структуру и особенности функционирования ЭВМ типа CRAY.
10. Назовите типы соединительных сетей и способы описания связей.
11. Охарактеризуйте основные виды регулярных связей.
12. В чем сущность коммутаторов типа сред, что такое многомерные кубы?
13. Что такое каскадные коммутаторы? Каковы их свойства?
14. Каковы принципы управления процессом коммутации? Приведите основные характеристики коммутаторов.
15. Опишите структуру процессорной матрицы.
16. Как выполняются векторные команды узлами процессорной матрицы?
17. Охарактеризуйте систему управления процессорного элемента. Что такое активация?
18. Как выполняются команды коммутации в процессорной матрице?
19. Какое влияние на быстродействие процессорной матрицы и ее программу оказывает способ размещения данных в оперативной памяти.