1. Предыстория вычислительной техники

1.1 Эволюция ВТ.

В истории вычислительной техники четко выделяются два периода:
1) простейшие механические и электромеханические приборы и машины для вычислении;
2) ЭВМ и параллельные вычислительные системы.
Первый период это *предыстория современной вычислительной техники,* его называют также древней историй ВТ. Второй период это *новая и новейшая история* ВТ.

Первое направление основывается на эволюционной модификации концептуальной последовательной ЭВМ фон Неймана. Достижения в области ЭВМ отражены в 3ех поколениях:
1. 1949г. Ламповая элементная база, последовательная обработка инф-ии.
2. 1963г. Транзисторная технология, интегральная схема.
3. 1965г. Транзисторная технология, интегральная схема. Отступление от последовательной обработки инф-ии. Допускалось совмещение выполнения нескольких операций, что делало ЭВМ простейшей выч-ой системой.
Пределом развития функциональной структуры ЭВМ яв-ся конвеер. Совершенствование архитектурных возможностей средств обработки инф-ии в этом направлении достигается за счет элементной базы. После 3-го поколения процесс развития средств ВТ был настолько бурным, что трудно выделить какой-либо заметный период и назвать его поколением.

Эволюция ВТ:
развитие человека и общества неразрывно связано с развитием техники вообще и выч-ой в частности. Человек постоянно стремится к созданию средств, повышая его технические и физические возможности. Выделяют 2 ряда развития техники:
1. Физический ряд. Рычаг и простейшие механизмы, из которых создаются машины.
2. Выч-ый ряд. Простейшие приборы, мех-ие и электромех-ие машины, ЭВМ, выч-ые сети и системы.
Простейшие выч-ые инструменты.
1. Абак - прибор дискретного действия. Счетная доска. 754-753г до н.э. Доска, разделенная на полосы, по которым передвигаются марки.
2. Логарифмическая линейка - прибор аналогового действия. 1594г. Вычисления основаны на замене операции. Они свидетельствуют о дуализме ВТ на ранних этапах.
3. Арифмометр - настольная машина мех-ая, с ручным или электрическим приводом. Яв-ся системой счетных колес. Дальнейшим продолжением арифмометра стали счетно-арифметические машины, которые использовались в начале XX века, для выполнения банковских и финансовых расчетах. Эффективно использовалась в механизации. Программу вычислений реализовал сам человек.
4. Машина Беббиджа. Универсальная механическая выч машина. По замыслу функциональная структура включала арифметическое устройство, память, средства ввода-вывода, устройство управления.
УУ работало на основе программы, т.е. последовательности инструкций занесенных на перфокарту. В этой машине была заложена возможность автоматизации вычислений. Для автоматизации вычислений было важно то, что автоматически мог меняться подпроцессор в зависимости от полученного результата.
5. Машина Цузе.

Семейство Z. Z1 (1938) – цифровой механический компьютер с программным управлением. Двоичная кодировка и система представления чисел с плавающей запятой. Такт частота = 1Гц, опытная модель. Z2 (1939) – тоже лишь опытная модель. впервые использованы электромеханические реле. Лишь Z3 (1941) использовалась в решении задач германского военно-исследовательского ракетного центра. Все три уничтожены войной.

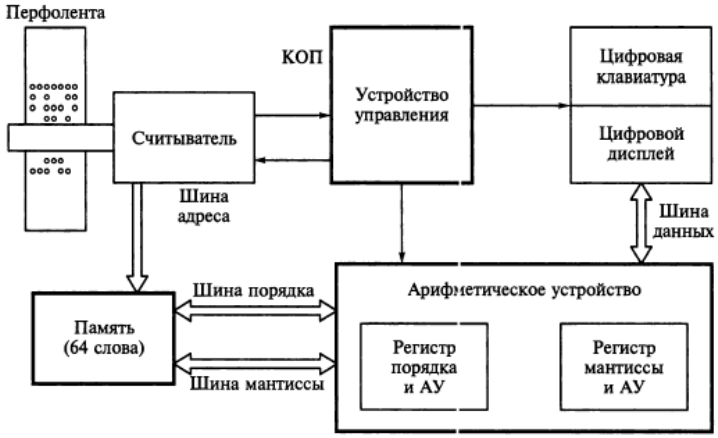


Рис. 1.1. Функциональная структура машины Z3:
→ — управляющие линии; КОП — код операции

1.2 ENIAC

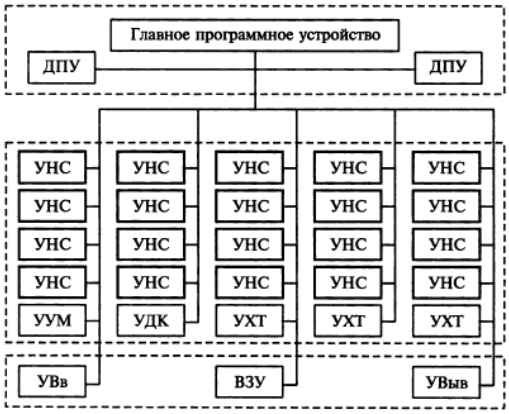
США, 1945. Вычислительная машина ENIAC кардинально отличалась не только от предшествующих ВМ, но и от машин, созданных позднее. В частности, от последних

ВМ ее отличало использование множества полуавтоматических электронных вычислительных устройств, работающих параллельно и полунезависимо, и реализация памяти только на электронных лампах.

Разработка машины ENIAC была поддержана Артиллерийским департаментом Армии США. С начала 1947 г. эта машина находилась на испытательном полигоне Баллистической исследовательской лаборатории в г. Абердине. Машина предназначалась прежде всего для расчетов по баллистике, в частности для составления таблиц стрельб.

Машина ENIAC это вручную перестраиваемая конфигурация, состоявшая из трех подсистем: управляющей, собственно вычислительной и ввода-вывода (рис. 1.2). Управляющая подсистема была представлена композицией из главного программного устройства (ГПУ) и двух дополнительных программных устройств (ДНУ). Вычислительная подсистема формировалась из 20 устройств накопления и суммирования (УНС), устройства умножения (УУМ), устройства деления и извлечения квадратного корня (УДК), и трех устройств хранения таблиц (УХТ).

Подсистема ввода-вывода состояла из устройств ввода (УВв) и вывода (УВыв) информации.



В ENIAC использовалась любопытная система представления чисел с фиксированной запятой, а не двоичная (см. разд. 1.1.5) и не двоично-десятичная (при которой каждый десятичный разряд представляется тетрадой, т. е. четырьмя двоичными разрядами). В данной системе каждый разряд десятичного числа (каждая из цифр от 0 до 9) представлялся десятью двоичными разрядами, из которых только один был равен единице. При этом номер позиции, в которой стояла единица, определял значение всей цифры. Например, цифра 3 выглядела как 0000001000, а цифра 9 как 1000000000.

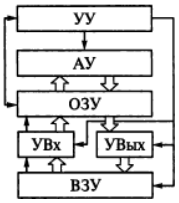
Вычислительная машина ENIAC имела следующие показатели: Тактовая частота 100 кГц; быстродействие 5000 и 350 опер./с соответственно при сложении и умножении десятиразрядных десятичных чисел; количество электронных ламп и электромагнитных реле 18 000 и 1500 соответственно.

1.3 EDVAC

США, 1950.

Автоматизация вычислений была одной из основных парадигм при проектировании машины: EDVAC это автоматический : Компьютер (Automatic Computer), т. е. ЭВМ, способная хранить в своей памяти программу вычислений. Отметим некоторые показатели EDVAC: тактовая частота 1 МГц (на порядок выше, чем в

ENIAC); быстродействие 1000 опер./с над 32-разрядными двоичными числами; емкость оперативной памяти 32 76;3 байт; количество электронных ламп —3000.



В отличие от ENIAC данная ЭВМ была последовательной машиной, она не могла выполнять две логические или арифметические операции одновременно. Особенности: автоматизация выч-ий, хранение программы в памяти и автоматическая модификация хода выч. процесса, конструктивная неоднородность.

1.4 Пути развития отечественных средств ВТ

Он сопровождался сопровождением не восприятия кибернетики в 50-х и параллельных выч технологий в 60-70х. Для важных отраслей ЭВМ развивались: МЭСМ-первая ЭВМ в СССР и Европе(маленькая эл-ая счетная машинка). Хар-ки: частота 5 кГц, 50 операций в секунду над 17-ти разрядными числами. Память на 94 слова. Система счисления двоичная, 6000 ламп, 25 кВт,S=60м^2. Команда условного перехода реализована программно. К машине подключается магнитный барабан на 5000 слов.

БЭСМ-быстродействующая эл-ая счетная машинка, 10000 операций в секунду, 5000 ламп, среднее время полезной работы 72%, 30кВт, S=100м^2. ОЗУ на ртутных линиях задержал 1024 32-разрядных слова. (1953 г.) В 1955 заменили на память на электронно-лучевых трубках.

БЭСМ-4 и М20. 20000 операций в секунду. Память на ферритовых сердечниках 4096 слов. 45 бит-разрядность. Элементная база – лампы и полупроводниковые диоды. Внешняя память на магнитных лентах и барабанах.

Особенности: зачатки параллелизма. Работа АУ совмещалась с выборкой команды и вводом/выводом информации.

БЭСМ-6. (1966 г.) 1 млн операций (одноадресных), длина слова 50 разрядов.

В Советском Союзе выпускались и другие оригинальные ЭВМ, например «Стрела» (1953), семейство машин « Минск», «Урал», «Мир» и др. В 1962-1965 гг. в Сибирском отделение: РАН были разработаны концептуальные основы построения вычислителі ных средств, основанных на новых принципах обработки информации (и.т:и, как сейчас говорят, с не-
фоннеймановской архитектурой). Эти средства стали называть вычислительными системами (ВС) или параллельным и ВС. Публикация в 1962 г. об американском проекте системы SOLOM0N появилась примерно через б месяцев после выхода в свет первой отечественной печатной работы по параллельным ВС. Первой параллельной ВС (причем с программируемой структурой) была система «Минск-222»

1.5 Современный уровень ВТ

Современный уровень выч техники. Около 10^5 операций в секунду до 1,5 млн. Наращивание за счет параллельных выч-ий и новых функциональных решений, и элементной базы. 1947 год – первый транзистор, 1959 год-первая интегральная схема, 1961 – четырех транзисторная схема, 1970-первый четырехразрядный чип

Intel4004. После 70-хгодов элементная база улучшилась за полгода в 2 раза. Современный уровень интегральных схем: частота 10^{10}гд, тр-ра до 10^{10}, S кристалла 100 мм^2, мощность несколько Вт.

Развитие средств ВТ идет по двум направлениям

1. *Электронные вычислительные машины и простейшие вычислительные системы.* Эти вычислительные средства основываются на эволюционных модификациях концептуальной последовательной машины дж. фон Неймана (1945). Их процесс развития отражен в трех поколениях. Функциональные структуры ЭВМ первого поколения (1949) полностью основаны на машине дж. фон Неймана и на ламповой элементной базе. Создание ЭВМ второго (1955) и третьего (1963) поколений сопровождалось не только отходом от принципа последовательной обработки информации, но и сменой элементной базы: переходом на транзисторы и интегральные схемы соответственно. Пределом в эволюционной модификации концептуальной ЭВМ дж. фон Неймана является конвейерный способ обработки информации в сочетании с векторизацией данных. Последний нашел воплощение уже в архитектурно развитых ЭВМ третьего поколения (допускающих одновременное или параллельное выполнение небольшого числа операций). А такие ЭВМ, по сути, представляют собой простейшие вычислительные системы.

   Для любого из трех поколений ЭВМ, для каждого из последующих этапов технического и технологического раз вития ВТ можно указать суперЭВМ или суперВС, т. е. вычислительные средства, обладающие предельными характеристиками по эффективности. Характерным для современного этапа является то, что архитектурные решения, которые были прерогативой суперЭВМ 1970-х и 1980-х годов, переместились с макроуровня на микроуровень, т. е. применялись г; современных микропроцессорах

2. Вычислительные системы. Эти средства базируются на принципе массового параллелизма при обработке информации. Вычислительные системы (в концептуальном плане) являются диалектической противоположностью ЭВМ, их функционирование основано на имитации работы не отдельных людей, занятых расчетами, а коллектив эв людей-вычислителей. Это позволяет преодолеть барьер производительности, существующий для ЭВМ, достичь высокой надежности и живучести, осуществимости решения задач, значительно улучшить технико-экономические показатели.

Вычислительные системы относятся к четвертому и последующим поколениям средств обработки информации. Современная ВТ представлена широким: спектром средств обработки информации от персональных компьютеров до ВС с массовым параллелизмом. Уровень быстродействия ЭВМ составляет миллиарды операций с плавающей запятой в секунду (GigaFLOPS).

2. Архитектура электронных вычислительных машин.

2.1. Каноническая функц. структура ЭВМ

Принципы фон Неймана:

1. Программное управление работой ЭВМ. Программы состоят из отдельных шагов команд; команда осуществляет единичный акт преобразования информации. Все разнообразие команд, использующихся в конкретной ЭВМ, составляет язык машины или ее систему команд. Таким образом, программа это последовательность команд, необходимая для реализации алгоритма.
2. Условный переход. Условный переход это возможность перехода в процессе вычислении на тот или инок участок программы. Условный переход позволяет легко осуществлять в программе циклы (с автоматическим выходом из них), итерационные процессы и т. п. Благодаря этому число команд в программе получается во много раз меньше, чем число выполненных машиной команд при исполнении данной программы.
3. Принцип хранимой программы. Этот принцип предопределяет запоминание программы вместе с исходными данными в одной и той же оперативной памяти. При функционировании ЭВМ команды выбираются из памяти в устройство управления, а операнды в арифметико-логическое устройство. В машине и команда, и число считаются словами. Если команду направить в АЛУ в качестве операнда, то над ней можно выполнять арифметические операции. Это открывает возможность преобразования программ в ходе их выполнения. Кроме того, принцип хранимой программы обеспечивает одинаковое время выборки команд и операндов из памяти, позволяет быстро менять программы или их части.
4. Использование двоичной системы счислений для представления информации в ЭВМ. В двоичной системе имеются только две цифры: о и 1, поэтому для их представления может быть использована любая система с двумя стабильными состояниями. Например, триод (открытое или закрытое состояния), триггер (с двумя устойчивыми состояниями), участок ферромагнитной поверхности (намагниченный или ненамагниченный), импульсная схема (наличие или отсутствие электрического импульса) и т. п. К логическим схемам (построенным по двоичной системе счисления) можно применять математический аппарат булевой алгебры. Итак, двоичная система счисления существенно упрощает техническую конструкцию ЭВМ.
5. Иерархичность запоминающих устройств (ЗУ). С самого начала развития ЭВМ существовало несоответствие между быстродействиями АУ и оперативной памяти. Путем построения памяти на тех же элементах, что и АЛУ, удавалось частично разрешить это несоответствие, но такая память получалась слишком дорогой и требовала значительного количества электронных компонентов (что снижало надежность ЭВМ). Иерархическое построение ЗУ позволяет иметь быстродействующую оперативную память сравнительно небольшой емкости . При этом следующий более низкий уровень представляют внешние ЗУ на магнитных лентах, барабанах и дисках. Внешние ЗУ имеют относительно малую цену, обладают большой емкостью, но меньшим быстродействием, чем оперативная память. Иерархичность ЗУ в ЭВМ является важным компромиссом между емкостью, быстродействием, относительной дешевизной и надежностью.

Процессор – это композиция из АЛУ, УУ и части ЗУ. если процессор имеет интегральное исполнение, то его называют микропроцессором.
Сопроцессор — специализированный процессор, расширяющий возможности центрального процессора компьютерной системы, но оформленный как отдельный функциональный модуль. Физически сопроцессор может быть отдельной микросхемой или может быть встроен в центральный процессор .
Главными характеристиками ЦПУ являются: тактовая частота, производительность, энергопотребление и архитектура.

2.2. Модель вычислителя. Hardware и Software
Вычислитель - это тот, кто вычисляет что-либо. Это либо человек занятый расчетом, либо тех-ое устройство.
Все вычислители основаны на примитивной имитации человека занятого расчетами.
Работа вычислителя не обходится без участия человека(оператора). Чем выше функциональные возможности, т.е. уровень автоматизации и механизации выч-ий, тем реже взаимодействие человека и вычислителя. Наилучшей степенью автоматизации обладает ЭВМ.
Следующая модель, представляющая основу функциональной организации ЭВМ:
C=<h,a>-описание.
h - конструкция, a - алгоритм работы вычислителя.
h=<U,g>
U-множество устройств, U={U_k}, k=5 в фон Неймана
g-структура связи между ними.
Под структурой g понимается граф, вершине которой сопоставлено устройство, а ребрам – каналы связи между устройствами.
Конструкция вычислителя основывается на трех функциональных принципах:
1. Последовательное выполнение операций на множество устройств U, взаимодействующих через структуру g.
2. Фиксированность структуры, невозможность автоматизировать изменения g.
3. Неоднородность состава устройств U, g.
a – алгоритм функционирования, допускает представление в виде суперпозиции.
a(P(D)) работы вычислителя определяется D-данными, которые он обрабатывает p-программой.
Алгоритм для P(D) должен приводить к однозначному результату.
Модель вычислителя:
C=<U, g, a(P(D))>
ЭВМ-средство обработки информации основанное на модели вычислителя.

Аппара́тное обеспе́чение - аппаратные средства, компьютерные комплектующие, жарг. железо (англ. hardware) — электронные и механические части вычислительного устройства, входящие в состав системы или сети, исключая программное обеспечение и данные (информацию, которую вычислительная

система хранит и обрабатывает). Аппаратное обеспечение включает: компьютеры и логические устройства, внешние устройства и диагностическую аппаратуру, энергетическое оборудование, батареи и аккумуляторы.
Програ́ммное обеспе́чение(допустимо также произношение обеспече́ние (ПО)) — Совокупность программ системы обработки информации.
Имеет место тенденция к вложению функций системного программного обеспечения в аппаратуру . Последнее поддерживается непрерывным совершенствованием технологии БИС, удешевлением элементной базы (в современных условиях микропроцессоров).

2.3 Понятие об архитектуре ЭВМ. SISD-архитектура.
Под архитектурой ЭВМ, как и вообще любых других средств обработки информации, в узком смысле понимают совокупность их свойств и характеристик, призванных удовлетворить потребности пользователей. Среди характеристик для юзера интересно:
-быстродействие
-форма представления чисел
-разрядность слов
-объем ОП
-характеристики устройств ввода вывода
-цена и показатели надежности
Архитектура вычислительного средства — концепция взаимосвязи и функционирования его аппаратурных (Hardware) и программных (Software) компонентов.
SISD (Single Instruction Stream – Single Data Stream) архитектура предопределяет такое функционирование ЭВМ, когда одиночный поток команд управляет обработкой одиночного потока данных. Примеры с архитектурой SISD: машина фон Неймана, EDVAC.

2.4 Понятие о семействе ЭВМ.
Совокупность архитектурно близких ЭВМ выделенную для фиксированного уровня развития ВТ и электронной технологии, называют семейством, или рядом, ЭВМ.
Границы семейства ЭВМ устанавливаются чисто условно, машины одгого семейства могут различаться по техническим характеристикам (например, по производительности) и по конструктивному исполнению.
Понятие семейство связано с понятием совместимости.
Совместимость-программа, приготовленная для какой-либо модели, дает один и тот же результат при ее использовании на любой из модели семейства. При этом следует учитывать направление совместимости сверху вниз или снизу вверх.
Совместимость ЭВМ в границах семейства проявляется в аппаратурном, программном и информационных планах.
Аппаратурная совместимость обеспечивается единством конструктивных решений, модульность построений ЭВМ, а также стандартизация связей и процедур

управления.
Снизу вверх - означает программа для младшей модели, может быть исполнена для любой другой модели.
Сверху вниз – на старшей модели могут быть приготовлены программы для реализации на любой из младших моделей.
Информационная совместимость обеспечивается использованием единым форматом представления данных, построения файлов одинаковых носителей данных.
Приведем примеры семейств ЭВМ. Самыми распространенными семействами «больших» машин третьего поколения в мире были IBM S/360 и IBM S/370, а в Советском Союзе ЕС ЭВМ и АСВТ-Д. Семейство ЕС ЭВМ включало в свой состав два подсемейства: «Ряд 1» и «Ряд 2». Машины «Ряд 1» были близки по архитектуре к моделям семейства IBM S/360, а машины «Ряд 2» к моделям IBM S/370.

2.5 Поколения ЭВМ

Поколения будем характеризовать совокупностью показателей эффективности и архитектурных свойств. Для представления используем вектор:
E={ω, ν,ϑ,σ}, где ω-показатель производительности (опер/с), ν –емкость опер. памяти (бит), ϑ – среднее время безотказной работы ЭВМ (ч), σ – «цена операции», отношение цены ЭВМ к показателю производительности.

Поколение ЭВМ	Годы появления	Возможности пользования	Алгоритм управления $a(p(D))$, структура	Элементная и логико-конструктивная базы	Производство	Программное обеспечение	Средства обмена	Показатели			
								Производительность ω, опер./с	Объем памяти ν, бит	Безотказность ϑ, ч	Цена 1 опер./с σ, долл.
1	1949–1951	Одна задача, пассивный режим	Последовательный алгоритм, фиксированная структура	Лампы, компоненты	Индивидуальное	Машинные языки	Устройства ввода-вывода (УВВ)	10^5	10^6	1…10	10
2	1955–1960	Набор задач, пассивный режим	Последовательно-параллельный алгоритм, фиксированная структура	Полупроводники, вентили	Мелкосерийное	Алгоритмические языки, трансляторы, диспетчеры	УВВ, каналы связи	10^6	10^7	10^2	10^0
3	1963–1965	Мультипрограммирование, активный режим	Последовательно-параллельный алгоритм, ручное изменение структуры	Интегральные схемы, группы вентилей	Серийное	Система языков, операционные системы	УВВ, каналы связи, оптические устройства	10^7	10^8	10^3	10^{-1}

2.6 Производительность ЭВМ. Понятие, показатели, единицы измерения.

Под производительностью ЭВМ понимается ее способность обрабатывать информацию. Как правило, когда говорят с производительности, то понимают под этим потенциальную возможность ЭВМ по обработке информации (а не реальную, учитывающую аномальности в работе ЭВМ, например простои из-за отказов, из-за профилактического обслуживания и т. п.).

Для оценки способности ЭВМ производить обработку информации используют количественные характеристики или показатели производительности. Распространенным и простейшим показателем производительности ЭВМ является тактовая частота. Она указывает, сколько элементарных операций может осуществить в единицу времени ЭВМ (точнее, ее процессор). Или, говоря иначе, время такта - время выполнения элементарной операции процессором ЭВМ.

Пусть {к1 , к2, ..., кn*}* часть набора операций, требующих обращение только к оперативный памяти. τ_j (тау) время выполнения операции кj, (время).

Пусть так же операции выполняются с равной вероятностью $\frac{1}{n}$

Выполнение одной такой операции = $\frac{1}{n}\sum_{i=1}^{n}\tau_i$

Номинальное быстродействие

$$\omega' = n * \left(\sum_{i=1}^{n} \tau_i \right)^{-1}$$

Очевидно, что при реализации на ЭВМ реальных программ имеет место не равновероятный выбор операций. Пусть ρ_i -вероятность выбора операции ki. Тогда *мат. ожидание* времени выполнения операции $\sum_{i=1}^{n}\tau_i\rho_i$

Обратная величина называется *быстродействием по Гибсону.*

$$\omega^{o} = \left(\sum_{i=1}^{k} \tau_i \rho_i \right)^{-1}$$ а значение ρ_i - весовые коэффициенты. Существуют несколько «смесей» Гибсона, которые отображают статистику задач, решаемых на ВМ. На практике, часто используют модифицированные значения показателей ω^{o} и ω', когда включаются только операции с фиксированной запятой. При решении задачи на ЭВМ, в общем случае, требуются затраты машинного времени на на ввод программы и данных, обращение к внешней памяти, работу ОС , вывод результатов и т.д.

Пусть {I1 , I2, ... , I i , ..., 1L } набор типовых(тестовых) задач, решаемых на ЭВМ- Benchmarks.

ν_i -число операций, непосредственно входящих в программу решения задачи Ii оно содержит в себе затраты на счет и доп. затраты.

$\omega_i = \nu_i / t_i$ -быстродействие при решении задачи I.

Обратные значения – 1/ Wi- среднее время выполнения одной операции при решении задачи I.

Пусть{ П1, П2 , ..., Пi , ... , ПL } распределение вероятностей спроса на типовые задачи Ii

,тогда $\sum_{i=1}^{L} \pi_i / \omega_i$ – среднее время выполнения одной операции при решен набора типовых задач.

Средним быстродействием ЭВМ называют величину

$$\omega = \left(\sum_{i=1}^{L} \pi_i / \omega_i \right)^{-1}$$

Существует множество тестовых наборов, одним из распространенных является LINPACK, предназначенный для решения задач линейной алгебры и позволяет оценить производительность ЭВМ на вычислениях с плавающей запятой.

Для измерения тактовой частоты ЭВМ используют мегагерцы МГц, а также гигагерцы (ГГц или GHz). Для оценки номинального быстродействия и быстродействия ЭВМ по Гибсону в случае, когда учитываются только операции с фиксированнои запяток, применяются следующие единицы измерения:

• MIPS (Million of Instructions Per Second), 1 МГР5 = 10^6 опер./с;

• GIPS, 1 GIPS = 10^9 опер./с.

Измерение производительности на тестовых наборах задач осуществляется в следующих единицах:

• 1 FLOPS (FLoating-point Operations Per Second), 1 операция с плавающей. запятой в секунду и ее производные (MFLOPS,GFOPS).

2.7 Показатели, характеризующие память ЭВМ. Кол-во информации, емкость.
Запоминающее устройство, или память (Metory, Storage) ЭВМ - функциональное устройство, предназначенное для приема, хранения и выдачи информации.

Для оценки возможностей памяти применяются показатели. Все они связаны с понятием *«количество информации»*, введенном в 1948 г. американским инженером и математиком К.Э. Шенноном (C.E. Shannon, 1916–2002).

Для оценки количества информации будем использовать формулу

$$H = -\sum_{i=1}^{n} P_i \log_2 P_i,$$

где i — одно из альтернативных устойчивых состояний памяти; P_i — вероятность нахождения памяти в состоянии $i \in \{1, 2, \ldots, n\}$, $\sum_{i=1}^{n} P_i = 1$; при этом считается, что $0 \log_2 0 = 0$.

Если память может находиться в любом состоянии с равной вероятностью, т. е. если $P_i = 1/n$, $i = \overline{1, n}$, то количество информации определяется формулой

$$H = \log_2 n.$$

Единицей количества информации называется *бит* (англ. bit от binary — двоичный и digit — знак). Бит — количество информации ($H = 1$), посредством которого выделяется одно из двух альтернативных и равновероятных состояний ($n = 2$) памяти. Слово bit используется также и для обозначения двоичной цифры и двоичного разряда.

Запоминающее устройство, способное хранить 1 бит информации, называется элементом (или ячейкой) памяти. Самым распространенным элементом памяти является триггер (trigger электронная схема с двумя устойчивыми состояниями).
Емкость памяти (Memory Capacity) максимальное количество информации, которое может в ней храниться. В качестве простейших единиц измерения емкости памяти применяют бит и байт. Существуют и укрупненные единицы емкости памяти ЭВМ :
1 Кбит (1 K bit) = 1024 бит = 2^10 бит;
1 Мбит (1 M bit) = 1024 Кбит = 2^20 бит;
Ширина выборки определяется количеством информации, записываемой в память или считываемой из нее за одно обращение. Время выборки промежуток времени с момента подачи сигналов чтения или записи до завершения соответствующей операции. Время обращения складывается из времени выборки и времени, которое расходуется на то, чтобы память была готова к реализации следующей операции обращения. Это время называют также длительностью цикла обращения к памяти. В течение цикла можно выбирать (считывать или записывать) информацию, обновлять или модернизировать состояние некоторых элементов памяти.
Быстродействие памяти характеризуется также пропускной способностью или скоростью обмена информацией между ней и другими устройствами. Эта скорость определяется количеством информации, которое можно записать в память или считать из нее в единицу времени. В качестве основной единицы измерения скорости обмена используют 1 бод = 1 бит/с (или 1 boud = 1 bit per second). К укрупненным единицам, характеризующим быстродействие памяти, относят:

1 Kboud = 1 Килобод = 1 Кбод = 10^3 бод;
1 Mboud = 1 Мегабод = 1 Мбод = 10^6 бод;

2.8. Надежность ЭВМ. Понятия, показатели, все гребаные функции.
Основополагающими понятиями теории надежности ЭВМ являются отказ и восстановление.
Отказом называется событие, при котором ЭВМ теряет способность выполнять заданные функции по переработке информации (включая функции по вводу и выводу информации, хранению и собственно преобразованию информации). Полный отказ нарушает работу всей ЭВМ, частичный – ухудшает функционирование. Устойчивый отказ – можно исправить только ремонтом, неустойчивый отказ – самоисправляющийся, временный.
Восстановлением называется событие, заключающееся в том, что отказавшая ЭВМ полностью приобретает способность выполнять заданные функции по обработке информации. Восстановление отказавшей ЭВМ может быть осуществлено автоматически (в общем случае с помощью аппаратурно-программных средств) или полуавтоматически (с участием бригады
технического обслуживания).
Прежде чем дать определения показателей надежности ЭВМ, введем случайные функцию ω(τ) и величину ξ. Пусть
ω(τ) =1, если в момент времени τ > =0 ЭВМ находится в работоспособном состоянии;
ω(τ) =0, если ЭВМ находится в неработоспособном состоянии.
Назовем ω(τ) производительностью ЭВМ в момент времени τ>=0, а ξ – моментом времени первого отказа.
Функции учить отдельно, вперед, спасти мир с учебником.

2.9. Предпосылки совершенствования архитектуры ЭВМ. Анализ возможностей совершенствования, архитектурные особенности параллельных вычислительных систем.
Если исходить из глобального трактования архитектуры ЭВМ, то легко заметить, что первые три поколения ЭВМ полностью основываются на модели вычислителя, на принципах, положенных в ее основу. Архитектуры ЭВМ, принадлежащих второму и даже третьему поколениям, являются модификациями архитектуры дж. фон Неймана.
Развитие средств обработки информации, направленное на достижение высокой производительности, надежности и живучести, натолкнулось в рамках модели вычислителя на серьезные препятствия.
Последовательность обработки информации.
Существуют следующие способы повышения производительности ЭВМ при обработке информации:

1) совершенствование и разработка алгоритмов решения задач;
2) создание эффективного ПО и оптимизация программ;
3) повышение быстродействия и улучшение физико-технических свойств элементов и внутримашинных информационных каналов;
4) улучшение алгоритмов выполнения машинных операций и соответствующая модификация структуры процессора;
5) модернизация алгоритма управления вычислительными процессами и канонической структуры ЭВМ.

Фиксированность структуры ЭВМ. Отсутствие возможности автоматического изменения структуры не позволяет в полной мере адаптировать ЭВМ к области применения (подобрать адекватную структуру и режим обработки), учесть особенности и характеристики задач при их программировании. Жесткость структуры ЭВМ в ряде случаев приводит к значительным трудностям программирования задач и не позволяет использовать эффективные методы их решения.

Неоднородность ЭВМ. Конструктивный принцип неоднородности в машине дж. фон Неймана реализован на нескольких уровнях: структура ЭВМ в целом нерегулярна, а состав гетерогенный: каждое из пяти устройств имеет свое функциональное назначение и свою логическую организацию, основывается на специфических принципах, обладает своими особенностями технической реализации. Построение неоднородных ЭВМ находится в резком противоречии с тенденцией развития микро- и наноэлектроники.

На современном этапе развития микроэлектроники (элементной базы для ВТ) и в перспективе технико-экономически оправдано создание средств обработки информации, функционирование которых должно быть основано на имитации работы не одиночных вычислителей, а коллективов вычислителей. Такие средства обработки информации получили название вычислительных систем (ВС). Параллельные ВС относят к четвертому, пятому и последующим поколениям средств обработки информации. Алгоритм управления вычислительными процессами в средствах четвертого поколения это универсальный параллельно-последовательный алгоритм с автоматическим изменением своей структуры. Структура вычислительных средств может также автоматически изменяться (программироваться) в зависимости от структуры и параметров решаемой задачи. Характерной особенностью средств четвертого поколения стало то, что многие функции программного обеспечения ЭВМ третьего поколения получили аппаратурную реализацию. Элементную базу ВС составили БИС (микропроцессоры и кристаллы памяти).

Пятое поколение вычислительных средств связано с решением еще более сложных (суперсложных) системных задач, известных под общим названием «проблемы искусственного интеллекта». Для решения задач такой сложности требуются самоорганизующиеся вычислительные средства, архитектура и функциональная структура которых должны допускать автоматические изменения универсального алгоритма

управления процессом вычислении в течение всего времени решения задачи.

5. Архитектура вычислительных систем

5.1 Модель коллектива вычислителей. Структура ВС, алгоритм функционирования, модель ВС.

Каноническую основу конструкции ВС и ее функционирования составляет модель коллектива вычислителей [5], которая представляется парой:

$$S = \langle H, A \rangle, \tag{3.1}$$

где H и A — описание конструкции (или конструкция) и алгоритм работы коллектива вычислителей.

Конструкция коллектива вычислителей описывается в виде

$$H = \langle C, G \rangle, \tag{3.2}$$

где $C = \{c_i\}$ — множество вычислителей c_i, $i = \overline{0, N-1}$; N — мощность множества C; G — описание макроструктуры коллектива вычислителей, т. е. структуры сети связей между вычислителями $c_i \in C$ (или структура коллектива).

Конструкция коллектива вычислителей есть отражение следующих основополагающих архитектурных принципов:

1) параллелизма (Parallelism, Concurrency) при обработке информации (параллельного выполнения операций на множестве С вычислителей, взаимодействующих через связи структуры G);

2) программируемости (Programmability, Adaptability) структуры (настраиваемости структуры G, достигаемой программными средствами);

з) однородности (Homogeneity) конструкции Н (однородности вычислителей с С и структуры G).

Структура (Structure, Topology) коллектива вычислителей представляется графом G, вершинам (узлам) которого сопоставлены вычислители $c_i \in C$, а ребрам — линии связи между ними.

Простейшие структуры ВС. Различают нульмерные, одномерные и двумерные простейшие структуры ВС . В первом случае структура сети межвычислительных связей «вырождена», взаимодействие между вычислителями ВС осуществляется через общую шину (Common bus, Uni bus). В случае одномерных структур (« инейки» Linear graph или «кольца» Ring) обеспечивается связь каждого вычислителя с двумя другими (соседними) вычислителями (рис. 3.1, а, б). В нульмерных структурах имеется общий ресурс шина, в одномерных же структурах этот ресурс трансформируется в распределенный, т. е. в локальные связи между вычислителями. Следовательно, архитектурные возможности (в частности, надежность) последних структур существенно выше, чем у нульмерных.

Увеличение размерности структуры повышает структурную надежность ВС. В самом деле, двумерные структуры предоставляют каждому вычислителю непосредственную связь с четырьмя соседними. В качестве примеров двумерных структур (рис. 3.1, в) может служить «решетка» (точнее, 2D-решетка Two-dimensional grid) и тор (2D-тор Two-dimensional torus). Следовательно, в системах с двумерной структурой при отказах некоторых вычислителей и (или) связей между ними сохраняется возможность организации связных подмножеств исправных вычислителей.

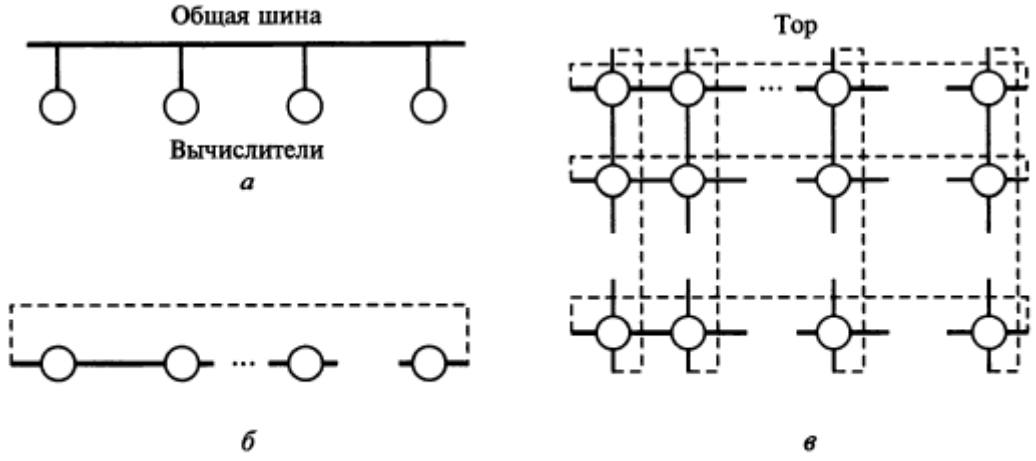


Рис. 3.1. Фрагменты простейших структур ВС:
а — нульмерная; *б* — одномерная; *в* — двумерная

Гиперкубические структуры ВС. Гиперкубы, или структуры в виде булевых n-мерных кубов, нашли широкое применение при построении современных высокопроизводительных ВС с массовым параллелизмом. Гиперкуб (Hypercube) по определению это однородный граф, для которого справедливо n = log 2 N, где N количество вершин; п число ребер, выходящих из каждой вершины; n называют также размерностью гип еркуба. Каждый вычислитель в гиперкубической ВС имеет связь ровно с n другими вычислителями.Гиперкуб размерности n называют также nD-кубом (D означает размерность (Dimension)).

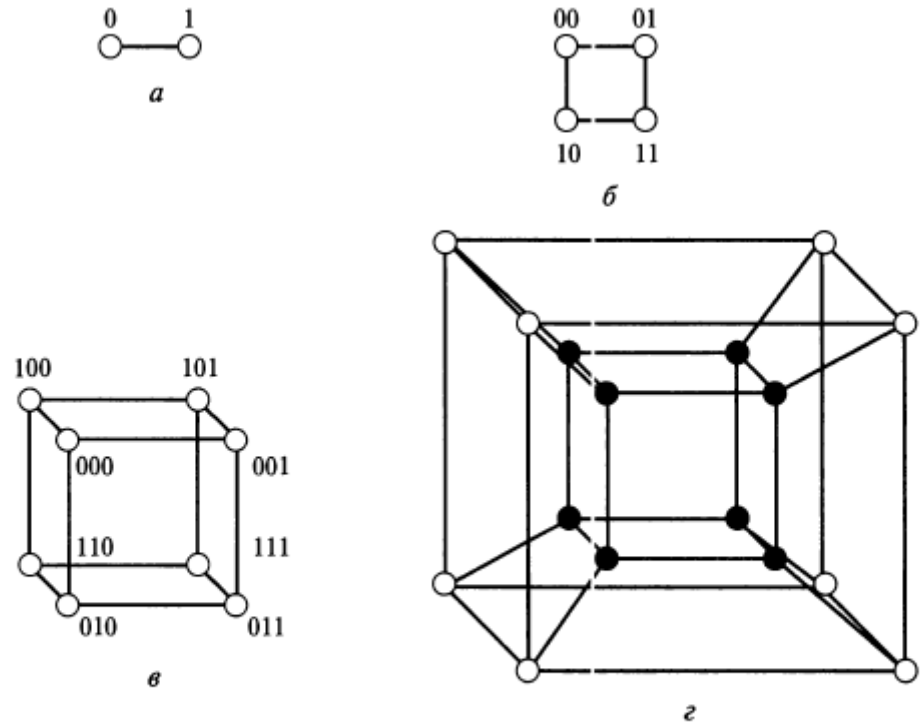


Рис. 3.2. Гиперкубические структуры ВС:

а — 1D-куб ($N = 2$, $n = 1$); *б* — 2D-куб ($N = 4$, $n = 2$); *в* — 3D-куб ($N = 8$, $n = 3$); *г* — 4D-куб ($N = 16$, $n = 4$)

3.1.3. Алгоритм функционирования вычислительных систем

Алгоритм A работы коллектива S обеспечивает согласованную работу всех вычислителей $c_i \in C$ и сети связей между ними (структуры G) в процессе решения общей задачи. Данный алгоритм может быть представлен в виде суперпозиции

$$A(P(D)), \tag{3.3}$$

где D — исходный массив данных, подлежащих обработке в процессе решения задачи,

$$D = \bigcup_{i=0}^{N-1} D_i, \tag{3.4}$$

D_i — индивидуальный массив данных для вычислителя $c_i \in C$, причем в общем случае

$$\bigcap_{i=0}^{N-1} D_i \neq \varnothing; \tag{3.5}$$

P — *параллельная программа* решения общей задачи,

$$P = \bigcup_{i=0}^{N-1} P_i, \quad \bigcap_{i=0}^{N-1} P_i = \varnothing, \tag{3.6}$$

P_i — ветвь i программы P.

Следует отметить, что формула (3.5) представляет собой *условие информационной избыточности*, оно позволяет организовать отказоустойчивые параллельные вычисления. В самом деле, за счет локальной информационной избыточности в каждом вычислителе имеется потенциальная возможность к восстановлению всей исходной информации (или, по крайней мере, необработанных данных) при отказе отдельных вычислителей.

3.1.4. Модель вычислительной системы

Формулы (3.1)–(3.3) позволяют записать для модели коллектива вычислителей или модели ВС

$$S = \langle C, G, A(P(D)) \rangle, \qquad (3.7)$$

где C — множество вычислителей; G — структура сети связей между вычислителями; A — алгоритм работы множества C как коллектива вычислителей (взаимосвязанных через сеть G) при реализации параллельной программы P обработки данных D.

Модель коллектива вычислителей допускает создание средств обработки информации разнообразных конфигураций. Представление коллектива вычислителей, в качестве макровычислителя делает возможным формирование сверхколлективов или систем коллективов, или макросы числительных систем. Модель коллектива вычислителей применима и на микроуровне; с ее помощью можно строить вы числитель как коллектив, составленный из микровычислителей.

5.2 Техническая реализация модели коллектива вычислителей и ее принципы. Архитектурные свойства ВС.

Каноническое описание модели коллектива вычислителей определяют принципы ее реализации. Выделим основные:

Модульность (Modularity) принцип, предопределяющий формирование ВС из унифицированных элементов (называемых модулями), которые ф ункционально и конструктивно закончены, имеют средства сопряжения с другими элементами, разнообразие которых составляет полный набор. Функциональные и конструктивные возможности модулей, разнообразие их типов определяются исходя из требований, предъявляемых к ВС, и, безусловно, из возможностей микроэлектронной базы.

Модульность вычислительной системы обеспечивает:

1) возможность использования любого модуля заданного типа для выполнения любого соответствующего ему задания пользователя;

2) простоту замены одного модуля на другой однотипный;

з) масштабируемость, т. е. возможность увеличения или уменьшения модулей без коренной реконфигурации связей между остальными модулями;

4) открытость системы для модернизации, исключающую ее моральное старение.

Следует заметить, что принцип модульности распространим и на средства программного обеспечения ВС.

При конструировании ВС с массовым параллелизмом достаточно ограничйться единственным модулем-вычислителем, который бы обладал вычислительной и соединительной полнотой. Следовательно, модуль должен иметь средства автономного

управления, располагать арифметико-логическим устройством и памятью и содержать локальный коммутатор - схему для связи с другими модулями. На практике принято такой модуль-вычислитель называть либо элементарным процессором (ЭП), либо элементарной машиной (ЭМ). При этом считается, что ЭП это композиция из процессора и локального коммутатора.

Близкодействие (short-range interaction) - принцип построения ВС, обусловливающий такую организацию информационных взаимодействий между модулями-вычислителями, при которой каждый из них может непосредственно (без «посредников») обмениваться информацией с весьма ограниченной частью модулей-вычислителей. Следовательно, структура ВС позволяет осуществлять информационные взаимодействия между удаленными вершинами-вычислителями лишь с помощью промежуточных вершин-вычислителей, передающих информацию от точки к точке (point-to-point). Удаленными считаются те вершины в структуре ВС, расстояние между которыми более 1 (число ребер между которыми более 1). Принцип близкодействия допускает реализацию механизма управления ВС, не зависящий от количества составляющих ее вычислителей.

Вычислительные системы, основанные на принципах модульности и близкодействия, удовлетворяют также требованиям асинхронности, децентрализованности и распределенности.

Асинхронность функционирования (Asynchronous functioning) ВС обеспечивается, если порядок срабатывания ее модулей определяется не с помощью вырабатываемых тем или иным образе м отметок времени, а достижением заданных значений определенны (кап с правило, логических) функций. Использование асинхронных схем позволяет достичь в системе алгоритмически предельного быстродействия: модули ВС срабатывают немедленно после выполнения соответствующего условия. Применение асинхронных схем обмена информацией между вычислителями позволяет не учитывать разброс в их тактовых частотах и колебания времени задержки сигналов в линиях связи.

Децентрализованность управления (Decentralized control) ВС достигается, если в системе нет выделенного модуля, который функционирует как единый для всей системы центр управления. децентрализованное управление системой основано на совместной работе всех исправных модулей системы, направленной на принятие решений, доставляющих оптимум выбранной целевой функции. Децентрализованное управление системой (в отличие от централизованного) позволяет:

1) достичь живучести ВС, т. е. ее способности продолжать работу при отказах модулей (в том числе и тек, которые предназначены для принятия решений);

2) избежать очередей при обслуживании заявок на управление.

Распределенность ресурсов (State of distribution) ВС позволяет создавать такую систему, в которой нет единого ресурса, используемого другими в режиме разделения времени. Под ресурсами ВС понимаются все объекты, которые запрашиваются, используются и освобождаются в ходе выполнения вычислений, например, процессоры или даже

модули, входящие в их состав, модули оперативной памяти, внешние устройства, линии межмодульных связей, шины, файлы данных, компоненты ПО. Вместе с этим каждый ресурс распределенной ВС рассматривается как общий, доступный любому потребителю.

Архитектурные свойства ВС:

1. Масштабируемость (Scalability) ВС. Под масштабируемостью ВС понимается их способность к наращиванию и сокращению ресурсов, возможность варьирования производительности. Сложность (трудоемкость) задач, решаемых на вычислительных средствах, постоянно растет. Для сохранения в течение длительного времени способности ВС адекватно решать сложные задачи необходимо, чтобы она обладала архитектурным свойством масштабируемости. Это означает, в частности, что производительность, достигнутую ВС на заданном количестве вычислителей, можно увеличить, добавив еще один или несколько вычислителей.

2. Универсальность (Genericity, Generality, Versatility) ВС. Вычислительные системы алгоритмически и структурно универсальны. Принято считать, что ЭВМ (основанные на модели вычислителя) являются алгоритмически универсальными, если они обладают способностью (без изменения своих структур) реализовать алгоритм решения любой задачи. С другой стороны, ВС это коллектив вычислителей, каждый из которых обладает алгоритмической универсальностью, следовательно, и система универсальна (в общепринятом смысле).

3. Производительность (Performance, Throughput, Processing power) ВС. В отличие от ЭВМ, построенных на основе модели вычислителя, ВС не имеют принципиальных ограничений в повышении производительности. Увеличение производительности в них достигается з а счет не только повышения физического быстродействия микроэлектронных элементов, а главным образом увеличения числа вычислителей. При этом достигается простота настройки ПО на заданное число вычислителей в системе

4. Реконфигурируемость (Reconfigurability) ВС. Структурная и функциональная гибкости ВС обусловлены широкими возможностями систем по статической и динамической реконфигурации. Статическая реконфигурация ВС обеспечивается: варьированием числа вычислителей, их структуры и состава; выбором дня вычислителей числа полюсов для связи с другими вычислителями; возможностью построения структур в виде графов, относящихся к различным классам; допустимостью применения в качестве связей каналов различных типов, различной физической природы и различной протяженности и т. п. Благодаря приспособленности ВС к статической реконфигурации достигается адаптация системы под область применения на этапе ее формирования.

Динамическая реконфигурация ВС поддерживается возможностью образования в системах таких (виртуальных) подсистем, структуры и функциональные организации которых адекватны входной мультипрограммной ситуации и структурам решаемых задач.

5. Надежность и живучесть (Reliability and Robustness) ВС.

Под надежностью ВС понимается ее способность к автоматической (программной) настройке и организации функционирования таких структурных схем, которые при отказах и восстановлении вычислителей обеспечивают заданный уровень производительности или, говоря иначе, возможность использовать фиксированное число исправных вычислителей (при реализации параллельных программ решения сложных задач).

Под живучестью ВС понимают свойство программной настройки и организации функционирования таких структурных схем, которые в условиях отказов и восстановления вычислителей гарантируют при выполнении параллельной программы производительность в заданных пределах или возможность использования всех исправных вычислителей.

6. Самоконтроль и самодиагностика (self-testing and self-diagnostics) ВС.

В системах-коллективах вычислителей может быть применен нетрадиционный подход к контролю и диагностике:

1) в качестве контрольно-диагностического ядра ВС могут быть использованы любые исправные вычислители и в пределе ядро любого произвольно выбранного вычислителя;

2) выбор ядра системы и определение ее исправности могут быть выполнены автоматически (с помощью средств ВС).

Предлагаемый подход позволяет говорить о самоконтроле и самодиагностике ВС.

7. Технико-экономическая эффективность (Technical-economical Efficiency) ВС.

5.3 Параллельные алгоритмы. Элементарные понятия параллельного программирования, параллельный алгоритм умножения матриц. Показатели эффективности параллельных алгоритмов. Парадокс параллелизма. Сложные задачи. Схемы обмена информацией между ветвями параллельной программы.

Параллельные алгоритмы

Описание процесса обработки информации, ориентированного на реализацию в коллективе вычислителей называют параллельным алгоритмом. Он в отличие от последовательного предусматривает реализацию на каждом временном шаге множества операций, и так же как и последовательный сохраняет зависимость последовательных этапов вычислений от результатов предыдущих. Запись параллельного алгоритма на языке, доступном коллективу вычислителей называют параллельной, а этот язык – параллельным языком программирования.

Методы вычислительной математики и соответствующие алгоритмы, как правило последовательный процесс приспособления алгоритмов к решению в коллективе вычислителей называют распараллеливанием.

Теоретическая и практическая деятельность, направленная на создание параллельных алгоритмов и программ называют параллельным программированием. Качество алгоритмов у такого программирования определяется методикой распараллеливания.

Существуют два подхода: Локальное и глобальное (крупноблочное) распараллеливание.

При первом подходе алгоритм решения задачи расщепляется на предельно простые блоки (операторы операция) и требуют максимального выделения этих операций для каждого шага вычислений. Такой подход считается не эффективным, т.к. он обеспечивает не однородную загрузку вычислителей. Его следует применять для оценки предельных возможностей по распараллеливанию. Практически эффективным считается глобальное распараллеливание. Оно требует расщепления вычислительного процесса на крупные блоки (подзадачи), между которыми существуют слабые связи. Последнее обеспечивает относительно незначительные расходы времени на реализацию обмена. Такие подзадачи называют ветвями параллельных программ.
$P = \bigcup_{i=0}^{n} p_i$ n-количество ветвей.

Основным приемом является распараллеливание по циклам. На практике такое распараллеливание по циклам, дает количество ветвей равное количеству повторений цикла. В параллельный программе всегда требуется выполнять обмены информацией между ветвями. Так же в этой программе имеются последовательные участки.

Например: в начале и в конце при вводе или выводе информации. В этой программе последовательные участки могут встречаться и внутри, они соответствуют негрупповым обменам информацией.

Параллельные алгоритмы умножения матриц.

Анализ прямых и итерационных методов в вычислительной математике показывает, что в них используется умножение векторов и матриц данных.

Выработаем алгоритм для умножения матриц большого размера:

A[1:m, 1:n]*B[1:n, 1:q]=C[1:m, 1:q]

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1q} \\ b_{21} & b_{22} & \cdots & b_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nq} \end{bmatrix} = c_{ij} = \sum_{r=1}^{n} a_{ir} b_{rj} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1q} \\ c_{21} & c_{22} & \cdots & c_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mq} \end{bmatrix}$$

Пусть эту операцию умножения матриц нужно выполнить на N-вычислителях. Пусть m*n и n*q – числа, много больше количества вычислителей. (m*n>>N, n*q>>N)Это условие является необходимым. Распараллеливание следует применять для тех задач, которые недоступны для их решения на ЭВМ. Такие задачи называют сложными (трудоемкими).

При параллельной обработке необходимо, чтобы каждый вычислитель рассчитывал N-ую часть элементов матрицы С. Следует заметить, что в матрицах А и В требуют размещения в каждом вычислителе n-ой части элементов соответствующей матрицы. Необходимо произвести однородное распределение элементов этих матриц по вычислителям.

Например, этого можно достичь, если матрицу А разделить на N горизонтальных полос, а матрицу В на N вертикальных полос. Каждая из этих полос будет находиться в своем вычислителе, т.е. в l-ом вычислителе будут размещены следующие матрицы А: (l-1)]m/N[+1,(l-1)]m/N[+2,…l]m/N[B:(l-1)]q/N[+1,(l-1)]q/N[+2,…l]q/N[

-ближайшее к х целоечисло, такое, чтобы выполнялось неравенство:]x[>= x

Параллельный вычислительный процесс можно реализовать следующим образом.

Сначала первый вычислитель рассылает свою первую строку из матрицы А по всем остальным вычислителям. После этого все вычислитель параллельно рассчитывают свои элементы первой строки матрицы С. Затем первый вычислитель рассылает остальным вторую строку из своей полосы матрицы А и снова по тому же циклу. Рассылка элементов из первого вычислителя заканчивается после передачи всем другим строки]m/N[. Затем рассылкой занимается второй вычислитель и т.д. Применяя такое однородное распределение исходных данных и процедуру обработки мы получили параллельный алгоритм, состоящий из идентичных ветвей.

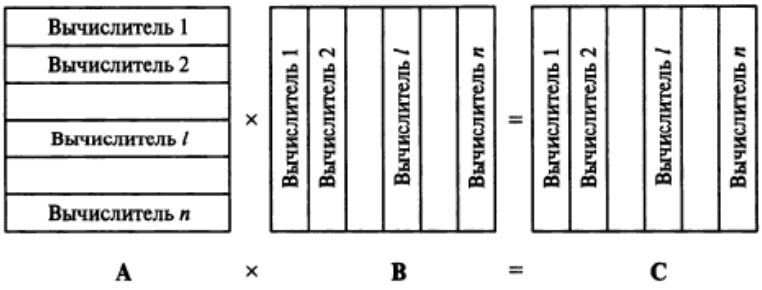


Рис. 3.4. Распределение данных по вычислителям ВС

Показатель эффективности параллельных алгоритмов.

Ясно, что каждая ветвь параллельной программы это последовательность параллельных операторов (логических), а так же операторов, обеспечивающих работу между ветвями. Чем выше эффективность параллельного алгоритма, тем меньше взаимодействие.

В качестве одного из показателей эффективности можно использовать коэффициент накладных расходов ε=t/T, где t-время, расходуемое на организацию и реализацию обмена, Т-время, расходуемое на вычисление. Чтобы получить эффективный алгоритм, нужно, чтобы каждый вычислитель перерабатывал как можно больше данных.

Следующий показатель-коэффициент ускорения $\chi=\tau_1/\tau_n$, τ_1- время решения этой же задачи на одном вычислителе (используется последовательная программа), τ_n-время решения этой же задачи на коллективе из n-вычислителей. Показатель χ показывает, насколько быстрее коллектив машин решает задачу быстрее, чем одна машина.

Коэффициент эффективности: E=χ/N, N – количество ветвей программы.

Парадокс параллелизма:

χ>n, E>1 - состоит в достижении ускорения и эффективности, превышающих n и 1 соответственно. Парадокс выражается в более, чем линейном росте производительности параллельной ВС с увеличением количества n ее вычислителей.

Этот эффект постоянно шокирует специалистов, но парадоксом, по сути не является, этот эффект объясняется следующим:

1. Эффект памяти

2. Возможность применения алгоритмов параллельных методов решения задач, реализация которых невозможна на последовательных ЭВМ из-за ограничения емкости памяти.

Пояснение к 1: Современная вычислительная система-коллектив вычислителей, каждый из которых имеет свою локальную память (ОЗУ). Сеть связей между вычислителями делает их локальную память общедоступной. Если суммарная локальная память всех вычислителей имеет достаточно большую емкость и допускает вложение параллельного алгоритма и данных. Следовательно, нет необходимости использовать более медленную внешнюю память, то будет достигнут минимум времени τ_n. С другой стороны, если сложность задачи достаточно высока и ее нельзя разместить в локальной памяти отдельного вычислителя и если она допускает решение на одном вычислителе только при использовании внешней памяти, то будет достигнуто значение τ_1, отличающееся от минимального и наконец, если время передачи между вычислителями сравнимо с временем обращения к локальной памяти, то будет иметь место следующее равенство: $\tau_1 = A_n * n * \tau_n$коэффициент $A_n > 1$, этим объясняется парадокс параллелизма.

Понятие сложной задачи.

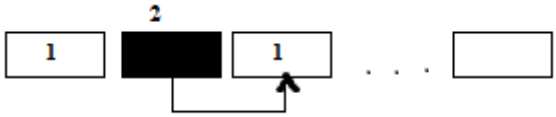
Ранее установлено, что эффективность реализации параллельных программ на вычислительных системах зависит от набора операций, которые следует выполнить при решении задачи, и от количества вычислителей, на котором реализуется параллельный алгоритм. В связи с этим, коэффициент накладных расходов можно написать в виде: $E(V,n) = {t(V,n)}/{T(V,n)}$где n-количество вычислителей, V-количество операций, которые

следует выполнить при решении задач. n=const$E(V,n) = 0$, при V->∞. Значение E будет практически удовлетворительным, если будет выполняться следующее: V=n*10^k, k-число большее, либо равное 1 – эмпирический коэффициент основанный на практике, зависит от быстродействия каналов связи между вычислителями. Если скорость каналов сравнима со скоростью обращения к локальной памяти, то k=1. Если количество операций превышает количество вычислителей, хотя бы на порядок, то такую задачу лучше решать на параллельной системе. Задачу, удовлетворяющую количество операций, принято

считать сложной. Следовательно, задача считается простой, если она может быть решена на одном вычислителе.

Схема обмена информацией между ветвями параллельного алгоритма.

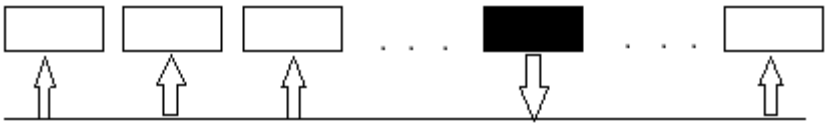
Параллельный алгоритм-композиция ветвей, связанных для осуществления обмена. Возникает вопрос о многообразии схем обмена. Ясно, что для схемы может быть использована с помощью дифференцированного обмена. При этом обмене информация из одной ветви передается в другую.



ДО:

Каждая ветвь реализуется на своем вычислителе. Ясно, что при таком обмене участвуют 2 вычислителя, а все остальные простаивают: т.е. этот обмен неэффективен, соответственно возникает вопрос: существуют ли групповые или коллективные обмены, при которых все вычислители работают одновременно. Анализ параллельных алгоритмов решения сложных задач показывает, что к таковым обменам относятся трансляционные(ТО), трансляционно-циклические, конвейерно-параллельные обмены.

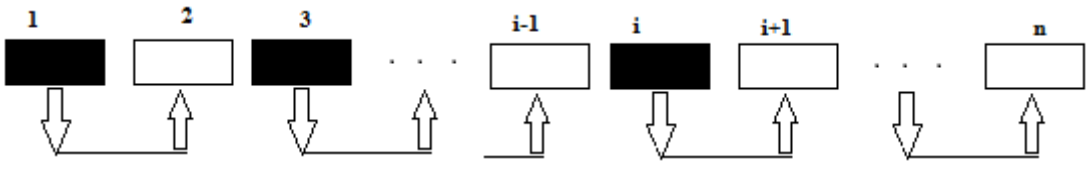
ТО (трансляционный)-из одного вычислителя информация передается всем остальным. Заметим, что данные схемы впервые были введены в 1960 году. Позднее они были обнаружены на Западе. В частности, трансляционный обмен получил название One-to-



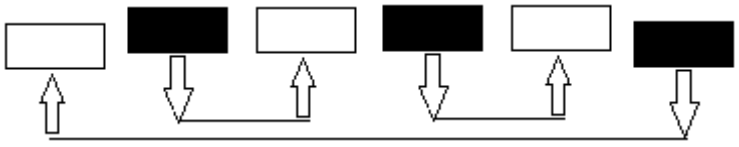
allBroadCast.

Т-Ц (трансляционно-циклический) (all-to-allBroadCast)-суть обмена заключается в повторении n-раз трансляционного обмена из каждой машины.

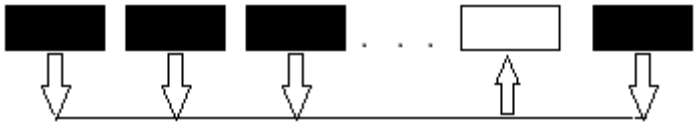
К-П (ковейерно-параллельный)–реализуется в два или несколько этапов. Например обмен, состоящий из двух этапов. При четном n на первом этапе информация из вычислителей с нечетными номерами показывается в соседнем вычислителе с четными номерами. На втором этапе, из четных вычислителей передаются в соседние не четные вычислители, с большим номером.



1 этап



2 этап

В этом случае, все вычислители работают параллельно. Помимо этих алгоритмов обмена существует также коллекторный обмен который, по сути, является инвертированным трансляционным обменом, при котором информация из всех вычислителей собирается в одно. 

Данный метод тяжело реализовывать технически, потому его реализуют при помощи n-1 дифференциальных обменов. Следовательно, в коллекторном обмене в любой момент времени, работают только два вычислителя, остальные простаивают. Могут быть и другие типы обменов, но вышеизложенные – явно характерные.

Опыт применения крупноблочного распараллеливания сложных задач.

Опыт эксплуатации советских вычислительных систем с программной структурой полученный в Сибирском отделении Советской академии позволяет сделать следующие выводы:

1. Сложные задачи допускают представление в виде параллельного алгоритма с идентичными ветвями, значит, проблема программирования сводится к написанию программы для одной ветви.
2. При распределении исходных данных между ветвями параллельного алгоритма эффективным является принцип однородного расчленения массива.
3. Для построения параллельного алгоритма достаточно использовать пять схем обмена.
4. Простота схем обмена между ветвями позволяет ограничиться только однородными структурами, что обеспечивает высокую технико-экономическую эффективность ВС.
5. Для записи параллельного алгоритма решения сложных задач эффективны версии широко распространенных языков высокого уровня, которые называют параллельными языками. Они отличаются от последних тем, что в них включены операторы для организации взаимодействия между ветвями, например могут быть введены операторы, соответствующие схемам обмена. Дополнение к транслятору для параллельного языка оценивается в несколько %.
6. Простота схем обмена и распределения данных по ветвям ведет к простоте написания параллельных программ, при этом трудозатраты на разработку параллельных программ, в отличие от последовательных, не превышает 10%.
7. Схемы ТО ТЦО и КПО составляют более 90% от всех схем. Параллельные алгоритмы, построенные на их основе и методике крупноблочного распараллеливания, характеризуется несколькими процентами затрат времени на обмены по сравнению с общим временем решения задачи.

Некоторые замечания о MPI

Message Passing Interface – интерфейс передачи данных.

Этот инструментарий получил широкое распространение в 90-е г. Позволяет организовать схемы обмена между ветвями параллельной программы. Процедуры MPI совмещаются с той или иной схемой обмена информации. Соответственно, схемы обмена информацией и других систем взаимодействия поддерживались даже в первой в мире системе с программируемой структурой:1965-66 гг.

Минск-222. В ней была реализована библиотека с соответствующими процедурамиMPI-это подобие выполненного стандарта. Эти схемы являются важнейшими инструментами при разработке параллельных программ.

5.4 Концептуальное понятие о вычислительных системах. Типы архитектур: MISD, SIMD, MIMD.

Дать четкое определение понятию «система» нельзя. С греческого это целое, состоящее из чего-то. Система-множество элементов, находящихся в отношениях и связях друг с другом, которые образуют определенную целостность и единство. Под системой, в рамках данного предмета, понимают, средства обработки информации, основанные на модели коллектива вычислителей. При достаточно общей трактовке под вычислительной системой понимается совокупность взаимосвязанных и одновременно функционирующих аппаратурно-программных вычислителей ,которая способна не только реализовывать параллельный процесс решения сложной задачи, но и в процессе работы автоматически настраиваться и перестраиваться с целью достижения адекватности между своей структурно-функциональной организацией и структурой и характеристиками решаемой задачи.

Модель коллектива вычислителей есть диалектическая модель развития вычислителя, следовательно, вычислительная система существенно отличается от машины Фон-Неймана. Существует классификация Флинна (1965). Предложенную классификацию вычислителей по потокам команд и данным, имеющие место в вычислительном средстве.

Поток команд-любая их последовательность, подлежащая исполнению вычислительным средством. Для использования команд требуются данные, следовательно поток команд порождает поток данных.

Четыре типа архитектур по Флинну: SISD - ЭВМ; MISD, SIMD, MIMD–вычислительные системы.

Разнообразием отличается MIMD структура. Виды MIMD: мультипроцессорные, распределенные, с программируемой структурой.

Мультипроцессорные ВС представляют собой множество процессоров и некий единый ресурс – это общая память (как правило). Взаимодействие процессоров с памятью осуществляется через общую память.

Распределенные ВС: Все ресурсы представляются в виде композиции функционально-завершенных элементарных машин.

ВС с программируемой структурой: обладают возможностью автоматически подстраиваться под структуры и классы решаемых задач, т.е. в таких системах могут быть порождены виртуальные и специализированные ВС.

Кластерные ВС – введено компанией DEC. Кластер – группа компонентов, которые связаны между собой и функционирует как единое средство обработки информации. Это синоним ВС, но есть отличия. Кластер может иметь любую архитектуру, кроме SISD.

Общая трактовка: Вычислительный кластер – композиция множества вычислителей, сети связи между ними и ПО, предназначенное для реализации параллельных алгоритмов. Кластерные ВС могут строиться из специально разработанных компонентов, но главным образом, они строятся из серийно произведенных аппаратурно-программных средств. Последнее является принципом проектирования кластерных ВС.

Примеры:

SIMD – ILLIAC IV, CRAY-1

MISD – пуст, нет представителей

MIMD – C.mmp, Cm*, Intel Paragon, CRAY Y-MP

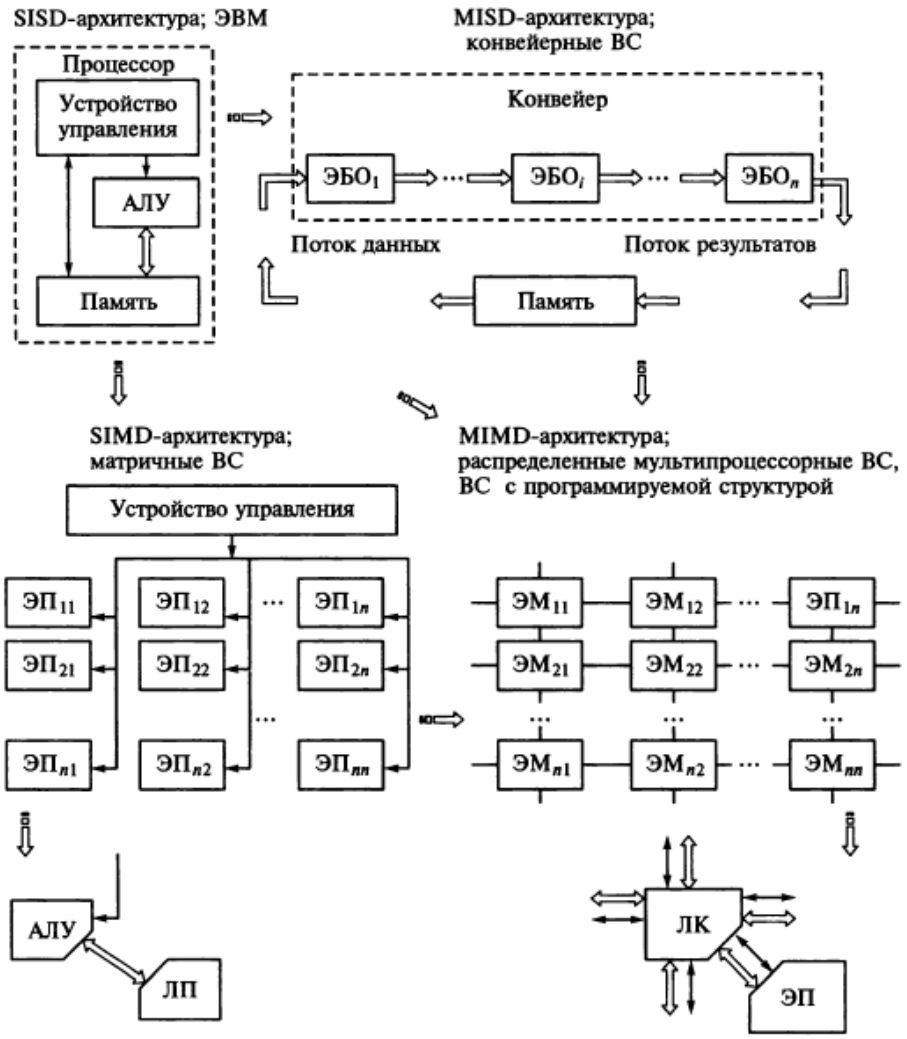


Рис. 3.7. Развитие архитектуры вычислительных средств:

АЛУ — арифметико-логическое устройство; ЭБО — элементарный блок обработки; ЭП — элементарный процессор; ЭМ — элементарная машина; ЛП — локальная память; ЛК — локальный коммутатор; → — поток команд; ⇨ — поток данных; ▫⇨ — направление трансформации архитектуры

6. Конвейерные вычислительные системы.

6.1 Каноническая функциональная структура конвейерного процессора. Назначение, векторные операции. MISD-архитектура. Структура и функционирование.

Конвейерные и инверторные ВС.

Каноническая функциональная структура конвейерного процесса.

ЭБО – элементарный блок обработки.

ЭБО1->ЭБО2->…->ЭБОn->память->ЭБО! (Цикл) (рисунок выше)

В конвейерной ВС основной объем операций по обработке данных выполняется одним или несколькими конвейерными процессорами. Конвейер оперирует с векторами данных, которые являются одномерными массивами или в терминах алгебры строкой или

столбцом. A=($A_1, A_2, \dots A_n$) В конвейере вектор операции реализуется аппаратурно, т.е. рассматриваются операции покомпонентного выполнения операций (+,*,/), либо формирования вектора из чисел, обработанных компонентом данного вектора. Для более сложных операций могут быть введены свои вектора команд. A+αB, (A+α)*B… A,B – вектора данных, α–некоторое скалярное число.

Конвейер в общем случае образуется как цепочка из элементарных блоков обработки информации и памяти. Каждый из блоков обработки (ЭБО) осуществляет частичное преобразование векторов – операндов.

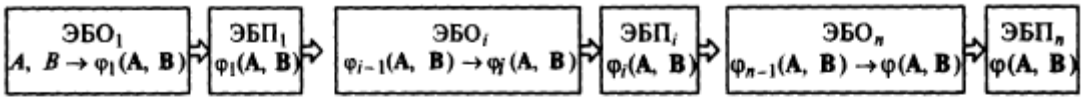
A,B->φ(A,B)->…->$\phi_i(A,B)_i$->… В составе ЭБО имеются элементарные блоки памяти, которые используются для хранения ЭБП; i={1,2,…,n-1} промежуточных результатов могут быть объединены в единое целое либо в оперативную память, либо в вектор регистра.

В простейшем случае, элементарные блоки обработки конвейера могут реализовывать отдельные фазы операций, т.е. выполнение микроопераций. Например, при сложении двух вещественных чисел, представленных с плавающей запятой необходимо выполнение следующих операций:

1. Сравнение порядков, 2. Выравнивание порядков, 3.Сложение мантисс, 4.Нормализация.

Элементы векторов подаются в конвейер в дискретные моменты времени и в соответствии с их расположением в векторах. На каждом шаге в ЭБО1 заносится новая пара элементов (операндов) в качестве операндов A,B. В ЭБОiiє{2,n} заносится информация из предыдущего ЭБОi-1. Процесс вычисления φ(A,B) для пары элементов из векторов A и B разделен на n этапов и все блоки конвейера работают параллельно, но каждый из них реализует свой этап вычислителей и обрабатывает свои элементы в фиксированный момент времени t. Время обработки на конвейере конкретных элементов векторов равно суммарному времени на преобразование во всех ЭБО.

Каноническая структура



Функционирование

$A_1, B_1 \to \varphi_1(\mathbf{A}_1, \mathbf{B}_1)$				
$A_2, B_2 \to \varphi_1(\mathbf{A}_2, \mathbf{B}_2)$	...			
	...			
$A_i, B_i \to \varphi_1(\mathbf{A}_i, \mathbf{B}_i)$	...	$\varphi_{i-1}(\mathbf{A}_1, \mathbf{B}_1) \to \varphi_i(\mathbf{A}_1, \mathbf{B}_1)$		
$A_{i+1}, B_{i+1} \to \varphi_1(\mathbf{A}_{i+1}, \mathbf{B}_{i+1})$	...	$\varphi_{i-1}(\mathbf{A}_2, \mathbf{B}_2) \to \varphi_i(\mathbf{A}_2, \mathbf{B}_2)$	...	
	...		...	
$A_n, B_n \to \varphi_1(\mathbf{A}_n, \mathbf{B}_n)$	...	$\varphi_{i-1}(\mathbf{A}_{n-i+1}, \mathbf{B}_{n-i+1}) \to$ $\to \varphi_i(\mathbf{A}_{n-i+1}, \mathbf{B}_{n-i+1})$	...	$\varphi_{n-1}(\mathbf{A}_1, \mathbf{B}_1) \to \varphi(\mathbf{A}_1, \mathbf{B}_1)$
$A_{n+1}, B_{n+1} \to \varphi_1(A_{n+1}, B_{n+1})$	...	$\varphi_{i-1}(A_{n-i+2}, B_{n-i+2}) \to$ $\to \varphi_i(A_{n-i+2}, B_{n-i+2})$	...	$\varphi_{n-1}(\mathbf{A}_2, \mathbf{B}_2) \to \varphi(\mathbf{A}_2, \mathbf{B}_2)$
	...		...	

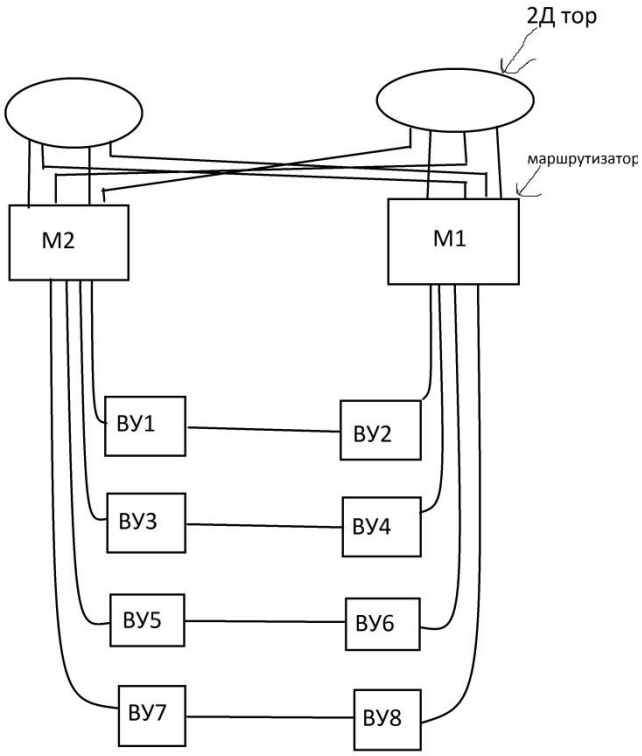
Рис. 4.1. Конвейерный процессор:
ЭБО — элементарный блок обработки информации; ЭБП — элементарный блок памяти;
A, B — векторы-операнды; $\varphi_i(\mathbf{A}, \mathbf{B})$ — частичное преобразование векторов **A, B**

Конвейерный процессор

У каждого процессора производительность 12.26 GFlops при работе с 64-х разрядными словами, также имеется возможность обработки 32-х разрядных слов. Такой тип относится к классу мультиплексорных процессоров, т.е. он является конвейером, но в его состав входит множество простых конвейеров. По сути, архитектура близка к системе Star 100. Каждый элементарный процессор состоит из 4 секций обработки информации и 4 блоков кэш-памяти. Взаимодействие кэш и секций осуществляется через коммутатор. Каждая секция обработки информации включает 1 скалярный блок с собственной кэш памятью и 2 векторных конвейера. Все компоненты даже в пределах секций обработки информации способны работать параллельно.

Рассмотрим коммуникационную среду данной системы. Структура представляет из себя проецированный 2Д тор. Вершины – композиция из выч узлов (вычислительная вершина).

Функциональная структура выч машины



М1 М2 – связь с четными и нечетными вершинами

Для характеристики латентности используется термин Диаметр – макс расстояние определенное на множестве кратчайших путей между всевозможными парами вершин

М – коммутатор, явл логической схемой задержки в котором малы по сравнению с пересылками в ВУ и поэтому диаметр структуры выч вершины = 2

Одна из возможных конфигураций системы – четырехмерный куб 16 вершин

В каждой вершине 8 узлов

Каждый узел – 4 элементарных процессора. Во всей системе 512 элем процессора

Вывод: опыт работы конвейерных ВС привел к необходимости создания распределенных систем с программируемой структурой, которая была сформулирована в 60-70 гг в СССР

6.2 Конвейерные системы «память-память». STAR-100, семейство Cyber.

Фирма CDC (Control Data Corporation основана в 1957 г. Сеймором Креем и Уильямсом Норрисом) начиная с 1973 г. выпустила ряд конвейерных ВС, архитектура которых относится к типу «память-память» (рис. 4.2). В таких ВС элементы-операнды векторов **A** и **B**, необходимые для выполнения векторных команд, выбираются непосредственно из оперативной памяти и результат $\varphi(\mathbf{A}, \mathbf{B})$ записывается в ту же память. В качестве преобразования $\varphi(\mathbf{A}, \mathbf{B})$ может быть результат одной из арифметических операций над элементами векторов **A** и **B**.

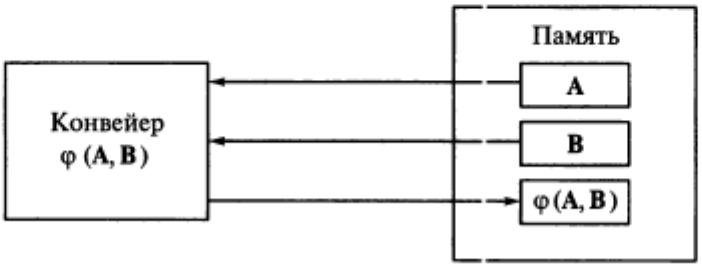


Рис. 4.2. Архитектура ВС типа «память-память»

STAR-100.

Разработка конвейерной ВС STAR-100 (STAR - STring ARray computer - векторный компьютер) осуществлялась фирмой CDC с 1965 по 1973 г. Быстродействие ВС 10^8 опер./с.

Состоял из 2-х подсистем

1 обрабатывала данные

2 обеспечивала работу ОС

Ядром первой подсистемы является процессор, образуемый из нескольких конвееров. В типовых конфигурациях было 3 конвейера: К1, К2, К3

Конвейеры были специализированными: К1 и К2 служили для выполнения векторных операций. К3 для выполнения или реализации программ над скалярными операндами.

К1 и К2 - конвейеры, которые служили для выполнения операций с плав запятой над векторами данными. К3 для обработки обычных операндов в неорганизованных векторах.

К1 и К2 определяли уровень быстродействия системы Star-100 в целом.

Конвейеры Star-100 имел программированную структуру, следовательно, в них могли быть выполнены различные арифметические операции, но до начала новой операции конвейер следовало перенастроить.

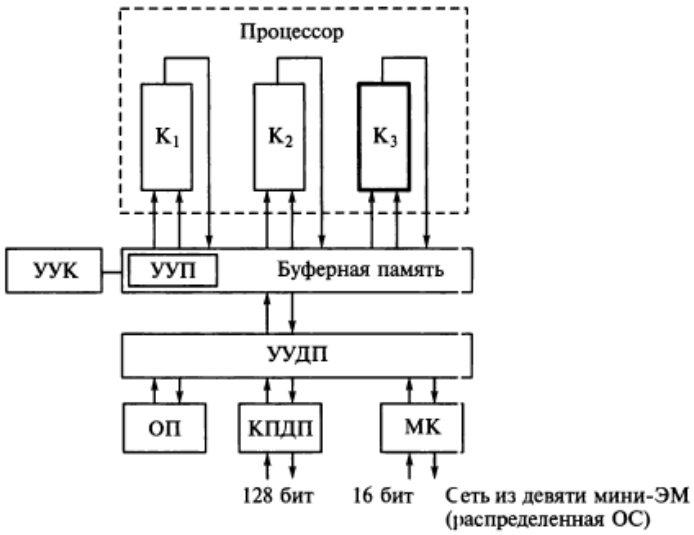


Рис. 4.3. Функциональная структура системы STAR-100:

K_1, K_2, K_3 — конвейеры; УУК — устройство управления командами; УУП — устройство управления потоками; УУДП — устройство управления доступом к памяти; ОП — оперативная память; КПДП — канал прямого доступа в память; МК — мультиплексный канал

Во всех трех конвейерах была заложена возможность реализации сложения, а умножения и деления – только в К1 и К2. Каждый конвейер мог включать в себя до 30 блоков информации. Все блоки работали параллельно. Любой конвейер воспринимал 64-разряд код, либо как 64 битный (разряд) операнд, либо как 2 32-х битных операнда.

Время выполнения операций над одной парой операндов в любом из конвейеров не превышало 40 нс. (10-9с) Следов-но, данные могли поступать в процессор со скоростью 100 млн оп/с.

Система машинных команд Star-100 состояла из 230 команд: 65 работа с векторами данных, 130 работа со скаляр.

Средства управления подсистемой обработки данных представлены композицией из устройства управления командами УУП и УУДП. УУК имело буфер опережаемого просмотра команд емкостью 4 512-ти разряд слова со стековым механизмом работы. УУП использовалось для управления потоками операнд и команд между конвейерами УУК и УУДП. В ОП хранились данные и программа. ОП реализована на магнитных сердечниках и имела емкость до 8 МБ.

В машине были реализованы 4 вирт канала обращения к памяти, которые реализовывались УУДП. 2 канала использовались для чтения операндов для конвейеров К1 и К2, один канал для записи результатов и один для вывода информации.

Буферная память была отведена вследствие того, чтобы быстродействие ОП было существенно ниже быстродействия процессора. Она представляет собой совокупность регистров со временем обращения цикла 110 нс. ОС выч машины относилась к классу распределенных, ее функции включали управление внешними запоминающими устройствами и устройствами ввода-вывода информации реализовывались специальной выч сетью из 7 минимашин.

Система включала в себя компилятор языков APL-100, Star, Sobol, Foltran.

Машина CYBER

По сути это модернизированный вариант системы STAR-100. Имелось две конфигурации: Cyber 203, Cyber 205.

Производительность Cyber-203 100 млн оп/сек имело ту же систему команд и ПО (полностью совместимое) . Машина является также конвейерной, но вместо КЗ стоял обычный скалярный процессор, который обеспечивал 6х увеличение быстродействия при обработке скалярной инфы. Емкость ОЗУ – 16 МБ.

Cyber-205:обладала более современной архитектурой, чем 203. В ней допускалось варьирование кол-ва конвейеров от 1 до 4. Пиковая производительность 200 млн оп/сек. Емкость ОЗУ -32 мб.

Все конвейеры Cyber-205 могли работать лишь в унисон, т.е. выполнять одну и ту же векторную операцию. След-но, архитектура Cyber-205 представляла SIMD-архитектуру. В составе аппаратурно-векторных операций машина Cyber-205 имела триады в виде A+αB, где A и B - вектора, α – скаляр. Выполнялись триады с той же скоростью, что и сложение, умножение векторов. На микроуровне схема Cyber-205 представляла систему MISD, а на макроуровне SIMD

6.3 Конвейерные системы «регистр-регистр». CRAY-1. Мультиконвейерные системы семейства CRAY.

Рис. 4.4. Архитектура ВС типа «регистр-регистр»

Каждый вектор-регистр способен хранить вектор-операнд, при этом при реализации вектор-команд векторы-операнды извлекаются покомпонентно из вектор-регистров. Вектор-результат запоминается также в одном из активных регистров. Векторы-операнды должны быть загружены в вектор-регистры из ОП до начала вектор команды. Вектор-регистры играют роль кэш-памяти. В рамках данной архитектурной концепции (конвейерные Выч системы «регистр-регистр») был выпущен ряд совместных моделей : Cray 1, Cray X-MP, Cray Y-MP, Cray C90, Cray T-90. Cray 1 была однопроцессорной, а остальные многопроцессорными. Т.е. любой из этих процессоров был ориентирован на

выполнение векторных операций и был мультиконвейерным. Модели X-MP, Y-MP, C-90 являются параллельно-векторными ВС. В настоящее время такие системы называют ПВП-системами (Paraller Vector Processors).

Cray 2, 3, 4 были архитектурно несовместимы с вышеизложенным списком, и компания перешла от к конвейерной концепции к концепции выч систем с массовым параллелизмом (МПП системы). Massively Parallel Processing: Cray T3D, Cray T3E, Cray T3F-900.

Cray-1

Произведена в 1976 году. Быстродействие 160 MFLOPS (16*10^7 FLOPS)и 37*10^6 над скалярами. Емкость ОЗУ от 8 до 64 Мб. Длина слова 64 разряда.

Предназначалась для работы над векторными и скалярными данными, состояла из следующих подсистем:

1) Управление программой;
2) Конвейеры
3) Регистры и память
4) Уустройство ввода вывода

Подсистема конвейеров по сути являлась процессором Cray и состоит из 12 конвейеров, которые разделялись на 4 группы:

1) Предназначалась для операций с адресами
2) Над скалярами
3) Для операций с числами с плав запятой
4) Для вектор-операций

Конвейеры состояли из сегментов ЭБО, каждый из которых был ориентирован на выполнение своей микрооперации.

Операционная система Cray (COS) обеспечивала режим пакетной обработки. Особенность архитектуры Cray-1 состояла в том, что система обладала способностью адаптации к решаемой задаче. Это достигалось возможностью настройки цепочек из произвольного числа конвейеров с произвольными их последовательностями. Система одновременно могла выполнять как несколько скалярных, так и несколько векторных операций

Параллельно-векторные системы Cray.

Это системы вида PVP по сути являющиеся коллективами, образованными из конвейерных процессоров. По началу это были процессоры Cray X-MP и Y-MP. Система X-MP – кластер из конвейерных процессоров, относится к классу MIMD; в состав входят 2, 4 процессора. Производительность каждого 235 MFlops. Область применения – разработка авиа-космических объектов.

Y-MP - модификация X-MP. В ней допускалось наличие 1-8 процессоров. Макс быстродействие 2,656 GFlops, каждого процессора 333MFlops.

CrayC-90 T-90

C-90 - быстродействие 16 GFlops. Кол-во процессоров:2,4,8,16. Емкость ОЗУ от 512 Мб до 8Гб. Относится к классу архитектуры MIMD.

Функциональная структура представляла собой композицию конвейеров, регистров и сетей связи. Конвейеры и регистры предназначались для обработки скаляров, векторов и операндов.

Конвейеры так же, как и у Cray-1, распределялись на 4 группы. Тип системы «регистр»-«регистр».

ОЗУ общедоступна. В системе реализовалась многопроцессорная обработка: режим 1 - выполнение нескольких независимых программы на разных процессорах, 2 - обработка данных одной программы на разных процессорах.

T-90 - Модификация C-90. Могла состоять из 4,16, 32 процессоров. Быстродействие 64 GFLOPS. Емкость ОЗУ от 512 Мб до 8 Гб. Система имела архитектуру MIMD, и была предусмотрена возможность макросистем из нескольких T-90.

6.4 Конвейерные MIMD-системы. Система Cray-T3D.

В конце 80-х годов XX в. ряд компаний (Thinking Machines, Kendal Square, NCube, MasPar и Mieko) успешно проводили исследования и разработки новых архитектур супер ВС (Massively Parallel Processing Systems), в которых высокая эффективность достигалась за счет применения большого числа элементарных (простых) процессоров. Системы с массовым параллелизмом (MPP-системы) стали альтернативой для векторно-параллельнх ВС (PVP-систем). Было разработано семейство массово-параллельных ВС, включающее возможные конфигурации моделей: Cray T3D , Cray T3E, Cray XT3 и Cray XT4.

Cray T3D.

Вычислительная система Cray T3D - первая MPP-система корпорации Cray Research, ее разработка была завершена в 1993 г. Это позволило фирме Cray Research Inc. быстро захватить лидерство на рынке MPP-систем. Количество элементарных процессоров в конфигурациях системы Cray T3D достигало 32.. .2048, а диапазоны производительности и емкости памяти были соответственно равны 5...300 GFLOPS и 512 Мбайт...128 Гбайт.

Система в максимальной конфигурации никогда не выпускалась; обычная конфигурация Cray T3D 64-процессорная, она обеспечивала быстродействие, равное 10 GFLOPS. Архитектура системы Cray T3D MIMD, а сама ВС принадлежит к виду распределенных. В системе достаточно полно воплощены принципы модели коллектива вычислителей. Последнее позволило, в частности, достичь в ВС Cray T3D высокой надежности и живучести, а также масштабируемости (варьируемости числа процессоров в пределах от 32 до 2048 с шагом 32). Следовательно, архитектура ВС Cray T3D приспособлена к формированию конфигураций с заданной производительностью и/или стоимостью. Система Cray T3D работает под управлением хост-системы (Host System - управляющая ВС). Одной из функций хост-системы является производительная подготовка программ (включающая компиляцию) и ввод-вывод данных для Cray T3D.

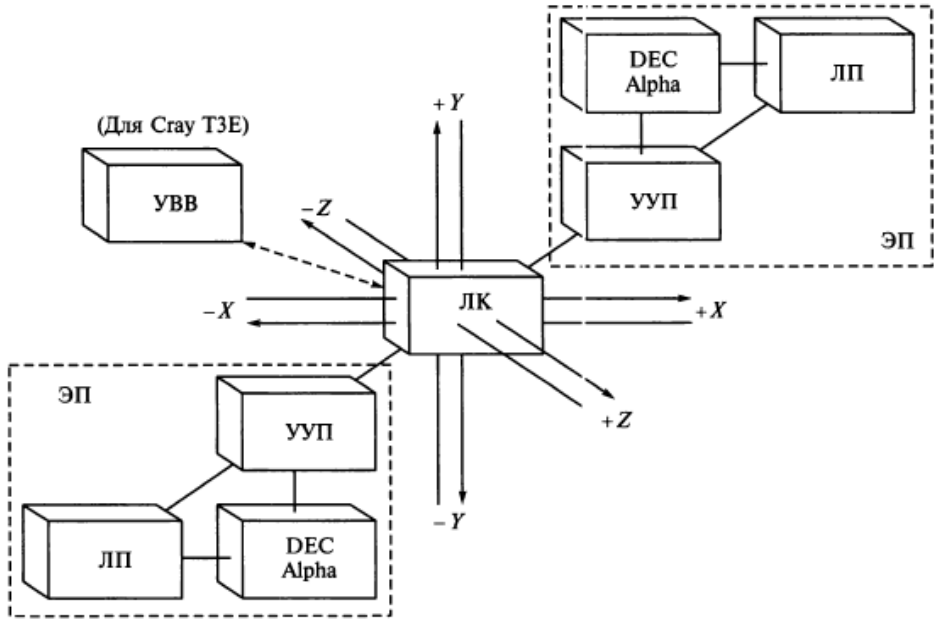


Рис. 4.6. Вычислительный узел системы Cray T3D (Cray T3E):
УВВ — устройство ввода-вывода; ЛП — локальная память; УУП — устройство управления памятью; ЭП — элементарный процессор; ЛК — локальный коммутатор

Вычислительный узел Cray T3D. Все вычислительные узлы (ВУ), составляющие ВС Cray T3D, однородные. Каждый узел (Processing Element Node) системы включает в себя два одинаковых ЭП и ЛК. Элементарный процессор (Processing Element процессорный элемент) представляется композицией из микропроцессора, локальной памяти (ЛП) и устройства управления памятью (УУП).

Локальный коммутатор обеспечивает непосредственную связь ВУ с соседними узлами и представляет собой шестиполюсник. В состав ЛК входят: сетевой маршрутизатор, сетевой интерфейс и контроллер для пересылки блоков данных. Сетевой маршрутизатор (Network Router) ВУ основной элемент управления коммуникационной сетью Cray T3D. Он способен работать с тремя парами двунаправленньх межузловых связей (Cotmunication links), что позволяет создавать трехмерные структуры ВС. Маршрутизатор каждого ВУ определяет путь перемещения каждого пакета данных и может осуществлять параллельный транзит данных по всем трем межузловым связям.

Вычислительная структура: 3D тор.

Преимущества : быстрая связь граничных узлов, небольшая задержка, повышенная живучесть структуры.

Адресация выч узлов разделена на физическую, логическую и виртуальную. Каждому выч узлу присваивается физический адрес и мог быть присвоен логический адрес, определяющий местоположение в логической конфигурации системы.

Виртуальная адресация введена для того, чтобы пользователю предоставлять дополнительный сервис, т.е. чтобы при программировании не учитывать физический и логический адреса. Любой из адресов представляется трехмерным вектором, который определяет расположение узла в 3D коммуникационной сети.

7. Матричные вычислительные системы

7.1 Каноническая функциональная структура матричного процессора. Назначение. SIMD-архитектура. Структура и эволюционирование матричного процессора. SOLOMON.

Матричные выч системы- данные системы по архитектуре относятся к SIMD с массовым параллелизмом. Данный класс предназначен для решения сложных задач, в которых преобладают операции над векторами и над матрицами преобразований.

Каноническая функциональная структура матричного процессора

Матричные (или векторные) процессоры (Array Processors) представляют из себя матрицу ЭП, которые взаимодействуют через сеть связи: работает через единое устройство управления. В состав ЭП входят АЛУ, локальная память, локальный коммутатор. УУ направляет текущую команду во все процессоры одновременно. ЭП исполняют команду параллельно, но каждый над своими данными. Данная структура была технико и канонически обоснована в 1960 гг, когда стоимость УУ составляла значительную сумму по сравнению с АЛУ и памятью, тогда электроника была дорогой. Но к началу 21 века эта структура оказалась востребована, т.к. стали создаваться кристаллы с множеством процессоров. Матричные процессоры нужны для решения сложной задачи, представленной параллельно. Режим мультипрограммирования не предусмотрен, но возможен, если УУ использовать в режиме распределения времени и пространства. Режим разделения времени используется УУ и его время делится между различными параллельными программами, находящимися в нем.Разделение пространства используется в матрице элементарных процессоров, каждой задаче выделяется место из элементарных процессоров. Матричный процессор в отличие от конвейерного не имеет принципиальных ограничений в увеличении производительности.

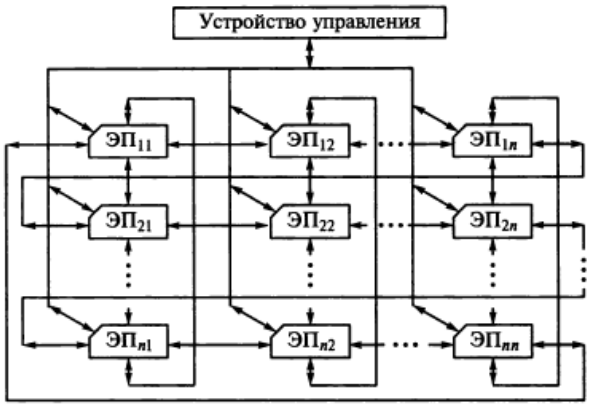


Рис. 5.1. Матричный процессор:
ЭП — элементарный процессор

Первая матричная ВС SOLOMON (Simultaneous Operation Linked Ordinal MOdular Network - вычислительная сеть синхронно функционирующих упорядоченных модулей) была разработана в Иллинойском университете США. Планировалось, что она будет иметь матрицу из 32 х 32 элементарных процессоров, способную выполнять операции над словами с переменной разрядностью от 1 до 128 разрядов. Каждый ЭП должен был иметь в своем составе АЛУ с последовательной поразрядной обработкой и память емкостью 16 К бит. Все ЭП в любой момент времени могли выполнять только одну и ту же операцию над числами, хранящимися в их ячейках памяти (с одними и теми же адресами). При этом каждый ЭП мог находиться либо в активном состоянии и выполнять команды, поступающие из устройства управления, либо в пассивном состоянии и не реагировать на эти команды. Устройством управления в системе SOLOMON могла служить серийно выпускаемая ЭВМ. Эта машина должна была иметь память для хранения программ и осуществлять связь с внешними устройствами.

7.2 Система ILLIAC-IV. Функциональная структура. Архитектурные возможности квадранта и элементарного процессора.

ВС ILLIAC – IV

В марте 1972 собрана первая модификация компанией Burroughs Corp, в октябре установлена в центре NASA. На практике 2*108оп/сек над 64 разрядными словами, состояла из 64 элементарных процессоров. Полезное время работы 80-85 %, т.е. сколько времени требовалось на решение задач, 20-15% на обслуживание, ремонт. Система включена в состав функц сети ARPA (AdvancedResearchProjectAgency), система эксплуатировалась до 1981 г. Публикации по SOLOMON появились на 6 месяцев позднее Новосибирских разработок, а ILLIAC на 6 лет позднее, чем МИНСК-22, хоть он не давал такой производительности, он обладал MIMD-архитектурой.

Функциональная структура ILLIAC-IV

Система должна была состоять из 4 квадрантов под систему ввода-вывода, систему управления, дисковой памяти и архивной памяти. Макс вариант, но в жизни был реализован 1 квадрант – матричный процессор, состоящий из матрицы 8x8=64 ЭП.

Квадрант - матричный процессор, включавший в себя устройство управления и 64 ЭП. Устройство управления представляло собой специализированную ЭВМ, которая использовалась для выполнения операций над скалярами и формировала поток команд на матрицу ЭП. Элементарная матрица из 64 ЭП предназначалась для реализации операций над векторами данных.

Рис. 5.2. Функциональная структура системы ILLIAC IV:
К — квадрант; ДП — дисковая память; АП — архивная память

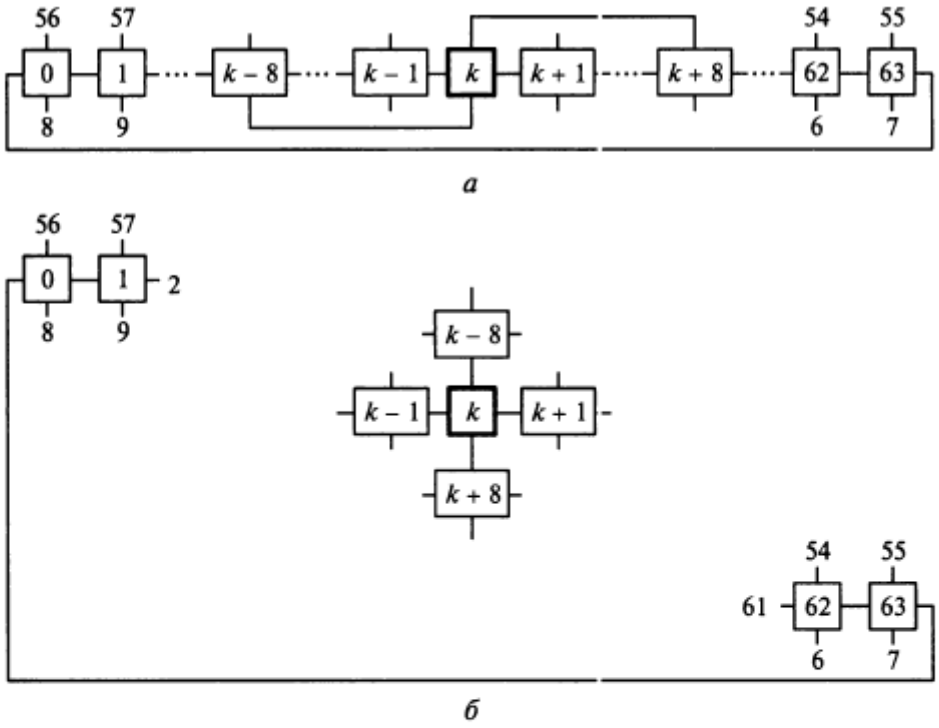


Рис. 5.3. Варианты изображения структуры квадранта ILLIAC IV

цессоры матрицы регулярным образом были связаны друг с другом. Структура квадранта системы ILLIAC IV представлялась двумерной решеткой, в которой граничные ЭП были связаны по канонической схеме (см. рис. 5.1). Позднее подобные структуры стали называться D_n-графами (термин предложен в Отделе вычислительных систем СО АН СССР [5]), и еще позже, в 90-х годах XX в., — циркулянтными графами. Структура квадранта ВС ILLIAC IV представляла собой D_2-граф вида {64; 1, 8} (см. разд. 3.1.2). Следовательно, в этой структуре, например, ЭП с номером 0 был связан с ЭП, имевшими номера 1, 8, 56, 63 (рис. 5.3).

В системе предусматривалось выполнение операций с плавающей запятой над 64 или 32 разрядными словами и с фиксированной запятой над 48, 24 и 8 разрядными числами. Если использовалось 64 разряда, то в векторе данных было 64 компонента. Это самая медленная операция.

Если 8-разрядные, то в векторе размещалось 512 компонентов с быстродействием 1010. Память каждого ЭП 2048 64-х разрядных слов. Архивная память – лазерная память с однократной записью, емкостью 1012Мбит

ВС В-6700 является хост-компьютером , управляла выч системой, через которую осуществлялся доступ к ресурсам ВС. Для восстановления системы (машины) существовали специализированные диагностические машины (в случае отказа).

ПО ILLIAC-IV

Создавались как средство решения сложных задач, представленных параллельными программами

Каждая такая прога состояла из 3 частей.

1) часть предпроцессорная – последовательная, выполняла 10-2чные преобразования и инициировала работу ВС

2) ядро проги – параллельная часть для описания требовалось 5-10% от общего кол-ва команд в проге. Реализация ядра занимала 80-95% общего времени выполнения проги

3) постпроцессорная обеспечивает 2-10чные преобразования, вывод результата на внешние устройство, в том числе и на графопостроитель, с которого была возможность осуществить переход в ядро и это последовательная часть

ОС ILLIAC работала под управлением управляющей программы B-6700

ОС состояла из набора асинхронных программ. Имелось 2 режима работы:

1) Обеспечивал контроль и диагностику, как квадранта так и подсистемы ввода-вывода

2) Обеспечивал выполнение пользовательских параллельных программ

Система программирования включала языки высокого уровня, в частности Tranquil – язык алгольного вида, освобождал пользователя от знания машины ILLIAC-IV.

Реализация транслятора для такого языка (когда маскировалась архитектура ВС) была трудоемкой и требовала огромного объема; от этого языка отказались.

Glynpir – язык высокого уровня, алгольный тип, требовал знания архитектуры системы, от него также отказались

Параллельный FORTRAN работал как обычный FORTRAN были добавлены схемы обмена между ветвями параллельной проги.

Сама машина ILLIAC использовалась как уникальное средство по обработки информации.

7.3 Система DAP. Особенности архитектуры, структура.

Разработку матричной ВС DAP (Distributed Array Processor распределенный матричный процессор) осуществляла английская фирма ICL (International Computers Ltd.). Работы были начаты в 1972 г., опытные образцы были построены в 1976 г. и 1977 г.

Система DAP по своей архитектуре относилась к SIMD-типу, это была ВС с массовым параллелизмом. Планировалось, что ВС будет состоять из 50 000 параллельно работающих ЭП, управляемых одним потоком команд. Каждый ЭП будет представлять собой монолитную большую интегральную схему, подсистема ввода-вывода информации будет выполнена также на БИС. Предполагалось аппаратно реализовать многие функции программного обеспечения. Функциональная структура ВС DAP - это композиция ведущей

ВС и собственно DAP. Ведущая ВС (Host Computer) предназначалась для реализации функций операционной системы (включая подготовку данных и команд для DAP, распределение данных по ЭП).

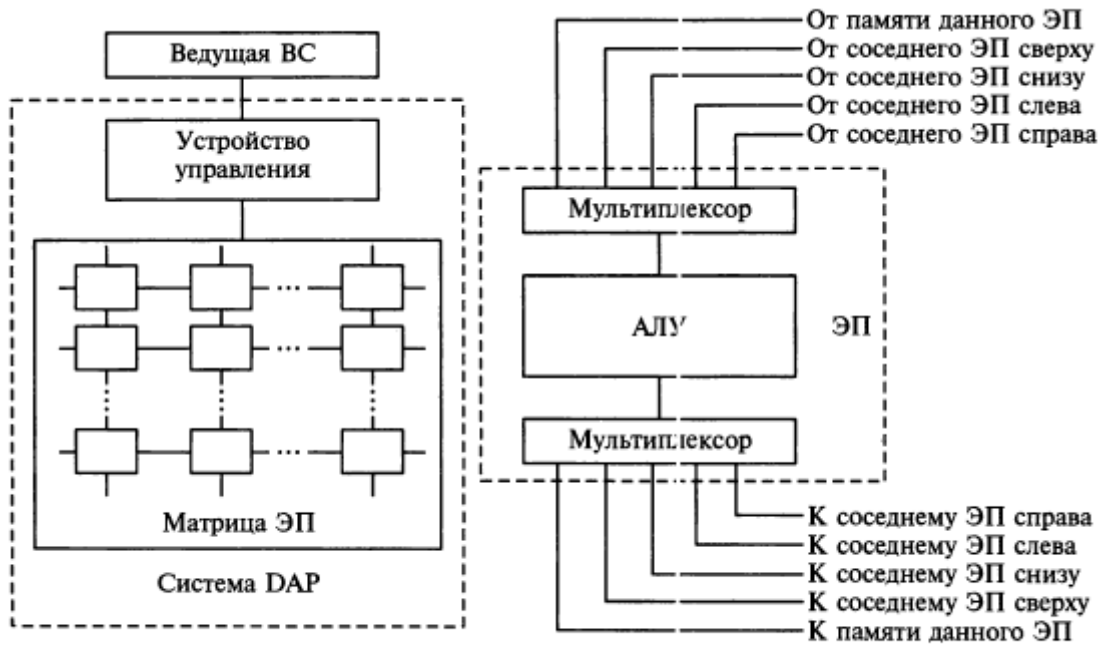


Рис. 5.4. Функциональная структура ВС DAP:
ЭП — элементарный процессор; АЛУ — арифметико-логическое устройство

Устройство управления формировало поток команд на матрицу ЭП, в частности оно направляло команды, адреса и другую информацию, необходимую элементарным процессорам для выполнения «матричных» операции. Между ЭП существовала сеть связей. Она обеспечивала через мультиплексоры связь каждого ЭП с регистрами АЛУ четырех ближайших соседей, расположенных сверху, снизу, слева и справа от него. Следовательно, сеть связей обеспечивала архитектурную гибкость ВС DAP.

7.4 Семейство Connection Machine. Функциональная структура, элементарные процессоры, модель виртуальной машины, ПО, модели семейства СМ.

Эволюция архитектуры матричных ВС и достижения в технологии БИС привели к созданию ВС с массовым параллелизмом, архитектура которых не может быть вписана в какой-либо один из канонов. Архитектура данных систем в зависимости от «глубины» просмотра может быть отнесена к классам MIMD и SIMD одновременно. Например, система в целом может иметь архитектуру MIMD, а ее основные процессорные компоненты SIMD; сети связей между элементами обработки информации на различных иерархических уровнях могут быть также различными. К таким ВС относятся модели семейства Connection Machine: СМ-1, СМ-2 и СМ-5.

Отметим архитектурные особенности систем семейства СМ:

• превалирующий класс архитектуры SIMD (MIMD, любой из моделей в целом);

• массовый параллелизм (MPP Massively Parallel Processing);

• максимальное число ЭП —65536 (или 2 1 '');

• быстродействие до 1 TFLOPS;

• однородность и программируемость структуры сети межпроцессорных связей;

• масштабируемость ВС

Вычислительная система СМ-1 первая модель семейства Connection Machine была спроектирована в Thinking Machines Corp. в течение 1983 г. и первой половины 1984 г.

Функциональная структура системы СМ-1. Модель СМ-1 семейства Connection Machine имеет достаточно развитую функциональную структуру, характеризуется иерархией средств управления процессами обработки информации. В состав ВС СМ-1 входят: параллельное процессорное устройство (Parallel Processor Unit); четыре сервисных процессора с интерфейсом шин (ИШ); коммутатор (Nexus).

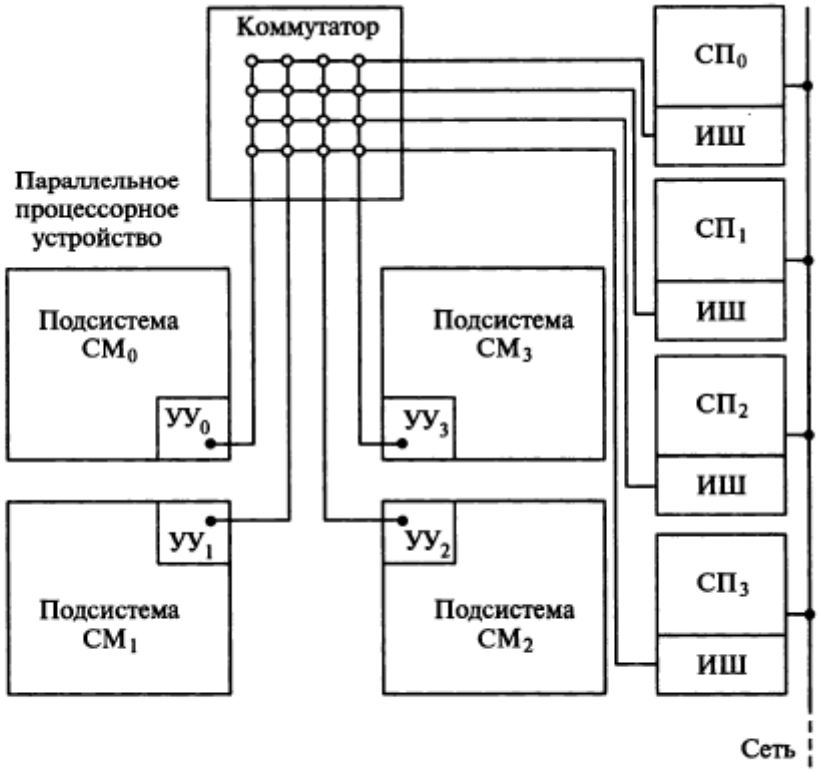


Рис. 5.5. Функциональная структура ВС СМ-1:
СП — сервисный процессор; ИШ — интерфейс шин; УУ — устройство управления

Основу любой конфигурации СМ-1 в целом составляет параллельное процессорное устройство с архитектурой MIMD, которое может иметь в своем составе от одной до четырех подсистем: СМ 0-СМ 3. Архитектура подсистем СМ 0-СМ 3 относится к классу SIMD. Следовательно, в пределах каждой из подсистем данные распределяются по

процессорам и одна и та же программа управляет работой множества процессоров (но каждого над своим подмножеством данных).

Устройство управления (Sequencer) ВС СМ-1 специально спроектированный микрокомпьютер для реализации функций виртуальной машины (архитектура, которой существенно удобнее для пользователя, чем у реальной физической ВС). Это устройство содержит память нанокоманд емкостью 16 К 96-разрядны слов. На входы четырех устройств управления поступает поток информации «высокого уровня», а именно операций виртуальной машины и аргументов. Этот поток поступает из коммутатора по синхронному параллельному (32-разрядному) каналу данных. На выходе УУ имеет место поток нанокоманд, которые и управляют работой элементарных процессоров и памяти.

Все элементарные процессоры ВС СМ сгруппированы в вычислительные узлы (или вершины) по 16 ЭП. Каждый узел конструктивно оформлен как объединение процессорного кристалла и кристалла памяти. Такой узел в целом называют просто процессорным кристаллом. В каждой из подсистем СМо—СМ3 имеется 1024 узла: Взаимодействие между узлами осуществляется через сеть связей, структура которой представляет собой 10-мерный гиперкуб. Сервисные процессоры (Front-ends), по сути, составляют аппаратурно-программную среду для разработки системного ПО. Они выполняют также функции ведущих (Host) процессоров и обеспечивают взаимодействие с сетью ЭВМ (Network).

Интерфейс шин (Bus Interfase) поддерживает 32-разрядный параллельный асинхронный канал между сервисными процессорами и коммутатором. Коммутатор (Nexus) предназначается для организации взаимодействия между сервисными процессорами и устройствами управления, он имеет размер 4 х 4 (4 х 4 Cross-point Switch). Коммутатор реализует механизм разделения, который позволяет в пределах ВС СМ-1 конфигурировать до четырех подсистем, работающих под управлением своего сервисного процессора.

Элементарные процессоры и вычислительные узлы ВС СМ-1.

Элементарный процессор - основной функциональный элемент системы СМ-1. Он имеет архитектуру SISD и является одноразрядным последовательным средством обработки информации. В состав каждого ЭП входят:

• одноразрядное АЛУ;

• битно-адресуемая локальная память (ЛП) емкостью 4 К бит;

• восемь одноразрядных регистров признаков (РП) или флагов;

• интерфейс маршрутизатора (ИМ);

• двумерный интерфейс сети межпроцесссрных связей (ИСМС).

Элементарный процессор ВС СМ-1 не является конструктивно оформленным элементом. В качестве конструктивной (да и функциональной) единицы выступает вершина или вычислительный узел.

При реализации узла используются два типа кристаллов. Первый – это оригинальный, специально спроектированный (заказной) кристалл. Этот кристалл называют процессорным, он содержит АЛУ, регистры признаков и коммуникационный интерфейс для 16 ЭП (маршрутизатор и средства межпроцессорной сети связей). Второй кристалл - коммерческая статическая память с произвольным доступом и с защитой по четности.

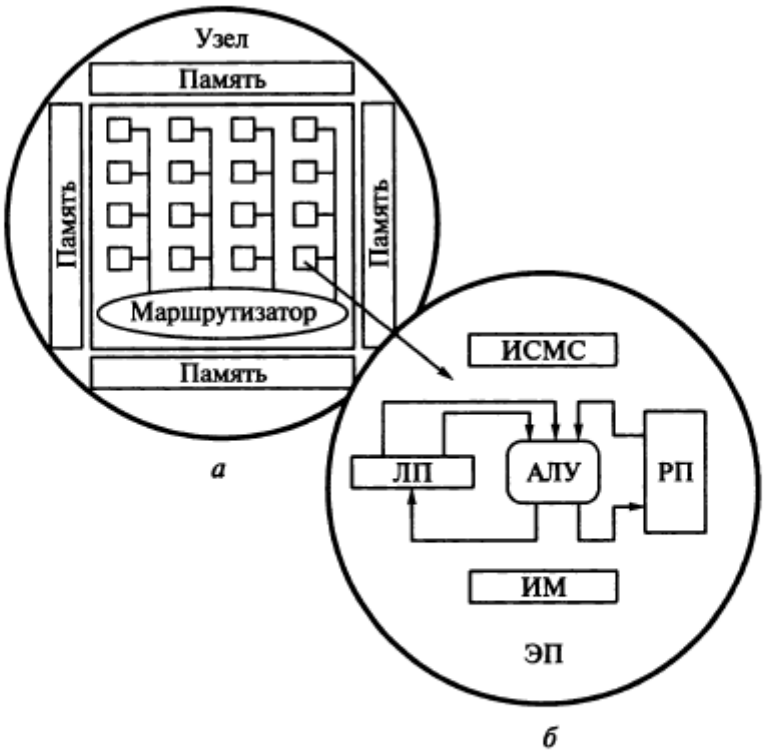


Рис. 5.6. Функциональные структуры ЭП и узла ВС СМ-1:

а — ВУ; *б* — ЭП; ИСМС — интерфейс сети межпроцессорных связей; ЛП — локальная память; АЛУ — арифметико-логическое устройство; РП — регистр признаков; ИМ — интерфейс маршрутизатора; ЭП — элементарный процессор

Виртуальная машина ВС СМ-1.

Пользователю ВС СМ-1 предоставляется удобный сервис - виртуальная машина. Архитектура этой машины весьма близка к архитектуре физической системы Connection Machine и имеет два существенных расширения: ее набор параллельных команд (названный Paris - Парис) существенно расширен и в ней имеется абстракция виртуального процессора. Границами набора команд Paris являются простые логические и арифметические операции и высокоуровневые операции, сортировка и коммуникационные операции. Функции интерфейса Paris между коммуникационным процессором и остальной частью системы СМ-1) сводятся к формированию потока кодов операций и аргументов. Аргументы это, как правило, начальный адрес и количество бит (разрядность операнда) . В качестве аргумента могут служить непосредственно данные или информация о широковещательном приеме. Большая часть набора Paris реализована

в аппаратуре УУ, где осуществляется синтаксический анализ потока ко, нов операции и аргументов и его преобразование в соответствующую последовательность нанокоманд для ЭП. Поскольку Paris является набором команд виртуальной машины, то вполне допустимо использовать те же самые имена и для языка ассемблера системы СМ-1.

Виртуальный процессор необходим во многих областях параллельной обработки данных (так как часто требуются специфические процессоры, которые заметно отличаются от физических ЭП данной системы). Программное обеспечение ВС СМ-1 предоставляет механизм виртуального процессора, он поддерживается Paris и легко понятен пользователю.

Программное обеспечение ВС См-1.

Основу системного ПО СМ-1 составляет операционная система (Operating System), являющаяся штатной операционной средой (либо UNIX, либо LISP) сервисных процессоров с небольшим расширением.

Тем не менее следует заметить, что стандартные языки программирования все же имеют некоторые расширения, поддерживающие параллельные конструкции данных. Однако эти расширения не требуют изучения какого-либо нового стиля программирования.

- Язык CM-FORTRAN системы Connection Machine использует расширения (Arrray Extensions) для работы с векторами, матрицами и массивами.

-Языки *LISP (читается Star LISP) и CM-LISP являются параллельными диалектами обычного языка LISP (LISP LISt Processing language язык обработки списков).

-Язык С* является параллельной версией С.

Connection Machine 2

Модель СМ-2 развитая версия СМ-1. При создании СМ-2 преследовали следующие цели:

• обеспечение совместимости с моделью СМ-1;

• увеличение производительности и емкости памяти;

• повышение общей надежности;

• упрощение производства;

• подключение высокоскоростной системы ввода-вывода (для внешней памяти и дисплеев).

Была совместима с семейством СМ-1 и предшествующими моделями. Обладала большими вычислительными возможностями. В ней аппаратно реализованы операции с плавающей запятой (1986-87 гг). Максимальное количество процессоров (максимальная

конфигурация) - 64 К элемент-процессоров быстродействия 2500 ms и 32 GFLOP (над 32 разрядными числами). Функциональная структура получила развитие: в ней появились подсистемы ввода/вывода и дисковая память большой емкости. Функциональная структура также была масштабируемой, минимальная конфигурация составляла 2 вычислительных узла и соответственно 32 элемент-процессора. Для связи вычислительных узлов конфигураций используется гиперкуб размерностью от 1 до 12. Количество элемент-процессоров (ЭП) было кратно 32. 12D-куб представляется 3D-кубом, в котором каждая вершина является 9D-кубом, а гиперребра имеют размерность 512. В ЭП был добавлен акселлератор для операций с плавающей запятой. Емкость памяти ЭП = 64 Кбит (значительно увеличена).

Connection-Machine-5.

Была реализована в 1991 году. Конфигурация состояла из 16384 ЭП, а максимальная производительность - 1 ТетаFLOPS (10^12 операций с плавающей запятой в секунду). Емкость памяти - 512 Кб. Архитектура ВС была излишня и представляла композицию из SIMD и MIMD архитектуры. Была масштабируема, варьирование количества выч. узлов от 16 до 16384. Соответственно, структура от 4D-куба до 14D-кубов. Множество вычислительных узлов были распределены на вычислительные и управляющие. Это позволяло составлять большое количество подсистем, в котором был один управляющий и множество вычислительных узлов. Машина имела структуру MIMD, а подсистемы - SIMD.

7.5 Анализ матричных ВС

Матричные ВС, начиная с 60-х годов XX в., относятся к основным концепциям построения сверхмощных средств ВТ. Матричный способ обработки информации в отличие от конвейерного в принципе позволяет осуществлять неограниченное количество вычислительных процессов, следовательно, достичь любого уровня быстродействия вычислительных средств.

Матричные ВС - вариант технической реализации модели коллектива вычислителей. В таких системах в высокой степени воплощены фундаментальные архитектурные принципы.

1. Параллельность выполнения операций в матричных ВС обеспечивается на нескольких функциональных уровнях. На макроуровне параллельность достигается за счет одновременной работы нескольких матричных процессоров (квадрантов в ILLIAC-N и процессорных подсистем в системах семейства Connection Machine). На микроуровне параллельность выражается в возможности одновременной работы большого количества элементарных процессоров (например, 64 ЭП в квадранте)

2. Программируемость структуры в матричных системах изначально проявлялась более сильно, чем в конвейерных. В самом деле, матричная ВС может быть так настроена, что ее

различные квадранты или подсистемы будут одновременно решать различные задачи. Кроме того, в пределах квадранта или подсистемы имеется возможность программировать направление передачи информации от каждого ЭП и, следовательно, настраивать канал связи между любыми ЭП. В матричных ВС заложены средства программного управления состоянием каждого ЭП. Последнее позволяет матрицу или подсистему ЭП разбивать на группы, каждая из которых может реализовать свой режим обработки данных.

3. Однородность состава и структуры ВС видна на всех функциональных уровнях. На макроуровне однородность выражена тем, что все матричные процессоры (или квадранты в ILLIAC-IV, или подсистемы в моделях СМ) и устройства управления, входящие в них, одинаковы. На микроуровне однородность ВС достигнута за счет применения множества идентичных элементарных процессоров. Сети межпроцессорных связей в матричных ВС однородные это и двумерные решетки, и гиперкубы. Однородность проявляется и в конструкции матричных ВС, они формируются из конструктивно однотипны элементарных процессоров или процессорных кристаллов-узлов.

Таким образом, матричные вычислительные системы с канонической архитектурой относятся к важнейшим вехам компьютерной истории. Матричные ВС с момента своего зарождение обеспечивали уровень производительности, адекватный потребностям в высокопроизводительных вычислениях и технико-экономическим возможностям общества.

8. Мультипроцессорные вычислительные системы

8.1 Каноническая функциональная структура мультипроцессора. MIMD-архитектура. Функционирование мультипроцессора.

Мультипроцессор ВС.

Ключевая особенность - наличие единого ресурса; много процессоров и, как правило, общая память (оперативная, внешняя, коммуникатор общей шины, ЭВМ-посредник). В качестве каноничной функциональной структуры мультипроцессора можно считать следующую функциональную структуру:

Рис. 6.1. Каноническая функциональная структура мультипроцессора:
ЭП — элементарный процессор; МП — модуль памяти; КЭШ — кэш-память

Композиция из множества ЭП, модулей памяти и коммутаторов, обеспечивающих взаимодействие 2х подмножеств: ЭП и модулей памяти. Любая область любого модуля памяти доступна любому ЭП. Следовательно, множество мультипроцессоров (МП) образуют общедоступную память.

Коммутатор может быть как сосредоточенным, так и распределенным. Распределенный представляется композицией локальных коммутаторов, каждый из которых находится во взаимноодназначном взаимодействии в соответствии либо с МП, либо с ЭП. Стоит отметить, что коммутатор в данном случае не является общим ресурсом, а таковым ресурсом является общая память. Обмен информацией между ЭП осуществляется не через коммутатор, а через общую память.Например, ЭП 1 заносит информацию в МП 2, а ЭП N извлекает ее из МП N-1. Соответственно. в функциональной структуре может использоваться шина либо система шин. В теоретических исследованиях по изучению парал. алгоритмов используется идеализированный параллельный микропроцессор (Parallel Random Access Machine). В рамках данной модели предполагается, что время обращения любого ЭП к любому МП есть постоянная величина. На практике это не так. Поэтому необходимость масштабирования мультипроцессора однозначно определяет наличие иерархии памяти, т.е. наряду с оперативной памятью вводится сверхбыстрая память – кэш, которая работает с большей скоростью, чем оперативная.

В кэш заносятся наиболее часто используемые данные, и поэтому сокращается количество обращений к оперативной памяти. Кэш-память может быть сосредоточенной или распределенной. Вычислительные системы на основе каноничной функциональной структуры мультипроцессора относятся к ВС с общей и разделенной памятью (True Shared Memory).

8.2 ВС C.mmp. Функциональная структура, анализ надежности.

При создании преследовали следующие цели:

1) достижение высокой производительности (большой полосы пропускания канала «процессор память»);

2) проведение экспериментальных исследований по эффективности параллельной обработки данных;

3) экспериментальное изучение и обеспечение надежности;

4) достижение приемлемых технико-экономических показателей;

5) воплощение принципа максимального использования аппаратурно-программных средств мини-ЭВМ.

Последний принцип позволил:

• свести разработку системы к работам по созданию лишь системных компонентов, тем самым не расходовать материальных ресурсов на проектирование и изготовление процессоров, памяти и устройств ввода-вывода информации;

• использовать ПО (в частности, контрольно-диагностические программы) серийной аппаратуры;

• достичь большей надежности в работе ВС как совокупности взаимосвязанных модулей обработки и хранения информации (благодаря их массовому производству).

Структура:

Систем состояла из 16 ЭП, общей памяти и матричного коммутатора. В состав ЭП входили незначительно модифицированный процессор, локальная (или местная, или индивидуальная) память, блок отображения адреса и контроллер межпроцессорного интерфейса (который обеспечивал подключение процессора к межпроцессорной шине). Кроме указанных компонентов в состав ЭП могли входить память на магнитных дисках, страничная память на дисках, внешние устройства и др. Матричный коммутатор 16 х 16 позволял установить связь между любым процессором ЭП и любым портом МП.

Рис. 6.2. Функциональная структура системы C.mmp:
ЭП — элементарный процессор; МП — модуль памяти

Анализ надежности.

Реальная ВС C.mmp содержала несколько подсистем одинаковых модулей (процессоров, модулей локальной памяти для каждого процессора, модулей памяти общего доступа для каждого из портов, блоков отображения, контроллеров межмашинного интерфейса и др.) и единственный коммутатор. Надежность ВС C.mmp существенно определял способ организации коммутатора. При этом разработчиками системы использовались две модели коммутатора. В простейшем случае (сосредоточенныій коммутатор) коммутатор рассматривался как единый элемент, выход которого из строя вызывал отказ всей системы. Вторая модель (распределенный коммутатор) отражала потенциальные возможности структуры коммутатора (далеко не все отказы коммутатора приводили к отказу системы).

Анализ надежности ВС C.mmp показал, что сосредоточенный коммутатор является критическим источником отказов в мини-ВС.

8.3 ВС В 6700, В 7700

В этих ВС нашли воплощение новые архитектурные и структурные решения, которые радикально отличались от концептуальных решений ЭВМ Дж. фон Неймана. Так, например, даже в ВС В 5000 было реализовано следующее:

- «ручная» реконфигурируемость состава (в ВС могло быть один или два центральных процессора и до восьми модулей оперативной памяти);
- механизм виртуальной памяти;
- аппаратурная реализация функций, выполнявшихся ранее программно;
- операционная система главная управляющая программа
- языки высокого уровня ALGOL 60 и COBOL.

В 6700.

Вычислительная система В 6700 (1971 г.) — это композиция ЭП, модулей памяти, коммутатора и периферийного оборудования (процессоров передачи данных, каналов, контроллеров и др.). Подмножество ЭП составляли 1-3 центральных процессоров и 1-3 процессоров ввода-вывода. Центральный процессор (ЦП) системы В 6700 обладал быстродействием порядка 1 млн опер./с (над 48-разрядными числами). Оперативная

память состояла из 1-64 модулей (МП), обладала емкостью до 6 Мбайт.

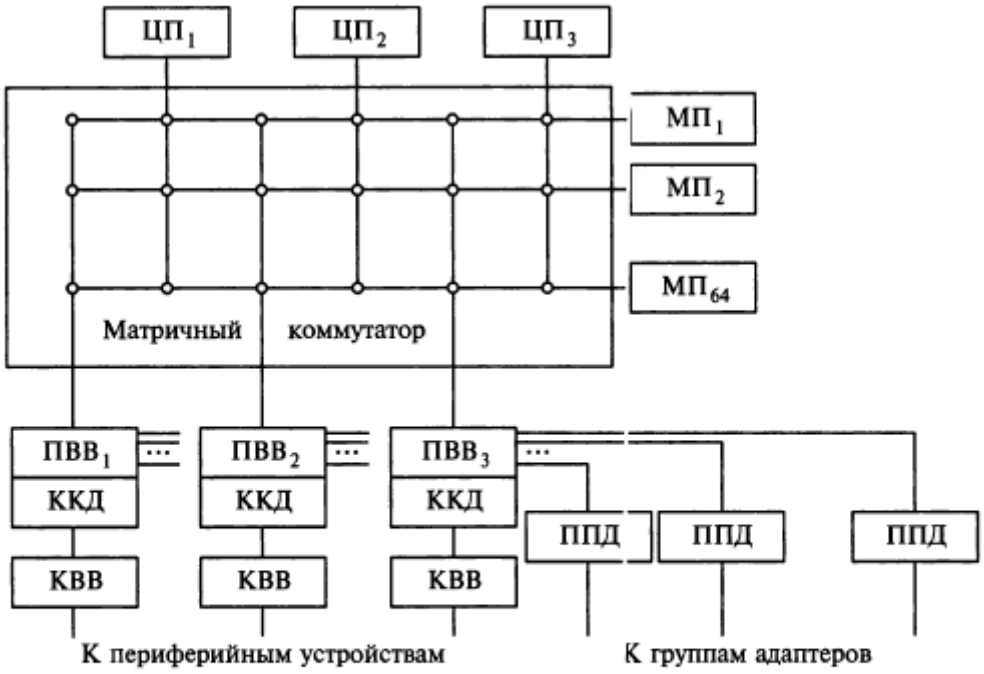


Рис. 6.3. Функциональная структура ВС В 6700:

ЦП — центральный процессор; ПВВ — процессор ввода-вывода; ККД — канал коммутации данных; КВВ — контроллер ввода-вывода; ППД — процессор передачи данных; МП — модуль памяти

Процессоры ввода-вывода (ПВВ) предназначались для подключения периферийного оборудования, в каждом из процессоров имелось 4-12 каналов коммутации данных (ККД). Каждый ПВВ был соединен с 1-4 процессорами передачи данных (ППД), каждый из которых был связан с 1-16 группами адаптеров.

В 7700.

Вычислительная система В 7700 (1973 г.) в отличие от В 6700 могла иметь в своем составе 1-7 ЦП и от 1-7 процессоров ввода-вывода информации, при этом общее количество процессоров не превышало восьми. Быстродействие ЦП составляло 4...5 млн опер./с. Процессор ввода-вывода обслуживал 32 канала и четыре процессора передачи данных. К каждому каналу подключалось периферийное оборудование; максимальное количество адресуемых внешних устройств составляло 255. Процессор передачи данных был рассчитан на подключение до 256 линий связи. Оперативная память ВС В 7700 состояла из восьми модулей, была 8-входовой и имела емкость 12 288 К байт. Слово состояло из 52 разрядов, из которых 48 были информационными, 3 управляющими и 1 для контроля по четности. Все разряды слова были доступны процессорам, как центральным, так и ввода-вывода и передачи данных.

Программное обеспечение системы В 7700 представляло собой модификацию ПО В 6700. Все системные программы (кроме ОС) и все прикладные программы могли работать как на В 6700, так и на В 7700 без каких-либо изменений.

8.4 ВС «Эльбрус».

В рамках работ по проекту «Эльбрус» преследовалась цель создать семейство высокопроизводительных и надежных ВС. Для моделей семейства «Эльбрус» характерны:

- MIMD-архитектура;
- распределенное управление;

• однородность, модульность и масштабируемость структуры;
• надежность и самоконтроль;
• аппаратурная поддержка функций ОС и средств языка высокого уровня;
• разрядность слов 32, 64, 128;
• многоуровневая память;
• спецпроцессоры приема-передачи данных;
• производительность до 125 MFLOPS.

Функциональная структура ВС Эльбрус.
Любая из моделей этого семейства относится к классу распределенных ВС и позволяет подобрать адекватную конфигурацию для области применения.

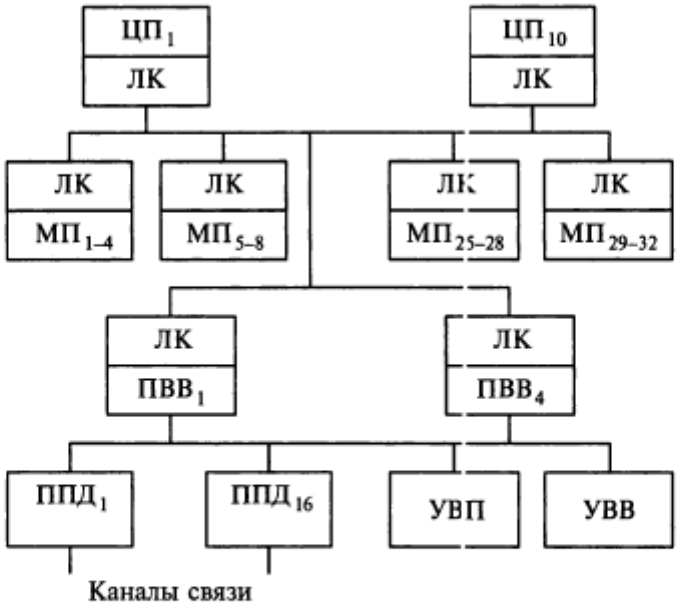


Рис. 6.4. Функциональная структура системы «Эльбрус»:

ЦП — центральный процессор; ЛК — локальный коммутатор; МП — модуль памяти; ПВВ — процессор ввода-вывода; ППД — процессор передачи данных; УВП — устройство внешней памяти; УВВ — устройство ввода-вывода

В состав входили до 10 ЦП и от 4 до 32 модулей памяти, от 1 до 4 процессоров ввода/вывода информации и от 1-16 процессоров передачи данных. УВВ – устройство ввода/вывода, УВП – устройство внешней памяти.
Взаимодействие между ЦП, МП и ПВВ осуществлялось через распределенный коммутатор, представленный композицией из множества локальных коммутаторов (ЛК) ППД, обеспечивая дистанционный доступ к системе. Всевозможные внешние устройства могли быть подключены непосредственно как к ПВВ, так и к ПДД.

Архитектурные особенности машины.
Каждый из элементов: ЦП, ПВВ, МП и ЛК имели 100%ный аппаратный контроль, т.е. даже при появлении одиночной ошибки возникал сигнал неисправности, который воспринимался ОС. ОС исключала отказавший элемент из рабочей конфигурации и переводила его в резервную конфигурацию. Для устранения неисправностей в элементе использовался комплекс контрольно-диагностических тестов и специальная аппаратура. После восстановления элемент снова включался в состав рабочей конфигурации, так проявлялась программируемость структуры. Кроме того, представлялась возможность построения сверхсложных конфигураций, когда для элементов программировались

резервные и подобные элементы. В такой структуре функции отказавшего элемента за 1 мл сек могут быть переданы резервному элементу.

Особенность системы Эльбрус.
Образец машины построен в 1976 году, промышленное производство – 1980 г. Производительность от 5 до 15 MegaFLOPS, емкость памяти до 4 МБ. ЦП мог оперировать с числами 32, 64, 128 разрядов с плавающей запятой. В состав ПО входила распределенная ОС, система программирования, комплекс стандартных серверных программ, программа телеобработки и комплекс программ технического обслуживания, куда входили контрольно-диагностические системы. ОС могла работать в режиме пакетной обработки, дистанционной обработки. ОС осуществляла динамическое распределение всех ресурсов, управляла работой процессоров и обеспечивала их синхронизацию, также ОС обеспечивала автоматическую реконфигурацию, восстановление файлов и перезапуск задач системы. Ситема программирования была определена всеми доступными языками.

Эльбрус-2.
Система с максимальной конфигурацией. Производительность – 125 MegaFLOPS, емкость памяти – 100 МБ, пропускная способность распределенного коммутатора – 2 Гбит/с. Часто встречаемы программные конструкции языков высокого уровня имели аппаратную поддержку. Диспетчеры процессоров ввода/вывода также были реализованы аппаратно. В машине допускалось формирование конфигурации Эльбрус-2 с использованием вместо ЦП специально разработанного процессора, полностью совместимого с БЭСМ-6 (чья производительность в 6 раз выше, чем у Эльбрус-2).

Другие модели Эльбрус.
1995 год – создана Эльбрус-3 в единственном экземпляре с 16тью процессорами. Дальнейшее развитие связывалось с разработкой процессора E2K-Elbrus-2000.
В настоящее время Институтом микропроцессорных вычислительных систем РАН выполняются работы в рамках госзаказа по созданию ВС «Эльбрус-3М». Данная мультипроцессорная ВС будет компоноваться из однокристальных высокопроизводительных микропроцессоров и иметь распределенную общую когерентную память.

8.5 Предпосылки совершенствования архитектуры мультипроцессорных ВС.
С развитием микроэлектроники и необходимости удовлетворения потребностей производителем вычислительных систем привело к разработке и применению модифицированной функциональной структуры мультипроцессора.

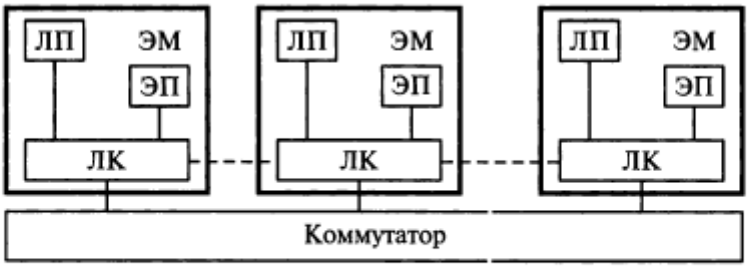


Рис. 6.5. Модифицированная функциональная структура мультипроцессора:
ЛП — локальная память; ЭМ — элементарная машина; ЛК — локальный коммутатор; ЭП — элементарный процессор

Промышленное устройство таких процессоров привело к тому, что соединения между ЭП и МП осуществляются через коммутатор. Необходимость достижения большей масштабируемости и надежности мультипроцессора, чем у каноничного мультипроцессора, заставило разработчиков перейти к элементам обработки, представленных в виде ЭП, ЛП и ЛК, и осуществить их взаимодействие через общий коммутатор. Следующим шагом развития стало использование не сосредоточенного, а распределенного коммутатора. Первым шагом на этом пути было создание коммутатора на основе принципа модульности, а затем уже программирования коммутатора, как композиции из ЛК, которых должно быть столько, сколько элементарных машин в системе. Если это будет так, то следующим шагом развития будет расширение функций ЛК в пределах одной машины и организация связей, представленных штрихом (------) в схеме. Но это уже архитектура, относящаяся к полностью распределенной. Соответственно, дальнейшее развитие такой структуры в многомерную и использование ЛК в многопользовательском режиме.
ВС, основанные на модифицированном мультипроцессоре, называют мультипроцессорными системами с разделяемой виртуальной памятью (Virtual Shared Memory). Современные процессоры – это по сути MPP системы.

8.6 Система Cm*. Архитектура, средства обеспечения надежности, система самодиагностики, анализ.
Вычислительная система Cm* была разработана Университетом Карнеги—Меллона и относилась к микроВС. Целью разработки являлось создание ВС из сравнительно большого числа (до 100) микропроцессоров (построение микроВС) и исследование аппаратурно-программных решений в области архитектуры средств на базе БИС.
Система Cm* (в сравнение с C.mmp)приобрела заметные архитектурные усовершенствования, в ней полнее были реализованы принципы модели коллектива вычислителей. Архитектура ВС Cm* стала более близкой к архитектуре ВС с программируемой структурой. В самом деле, в микроВС Cm * пара «элементарный процессор - локальная память» выполняла функции вычислительного модуля, а вычислительный модуль в совокупности либо с контроллером К отображения адресов, либо с локальным коммутатором. ЛК реализовывали функции ЭМ. В первом случае ЭМ выглядела как двухполюсник, а во втором как многополюсник (не менее двух полюсов). В состав ЭМ могли включаться внешние устройства (со своими контроллерами). Взаимодействие между вычислительными модулями осуществлялось через «распределенный коммутатор» сеть ;вязи, образуемую подсоединением контроллеров отображения адресов к межмодульным шинам или (и) отождествлением полюсов локальных коммутаторов различных элементарных машин. Контроллер - это специальный процессор, который выполнял все функции, связанные с передачей сообщений.

Локальный коммутатор имел простую структуру, обеспечивающую параллельную передачу слова между процессом нами.

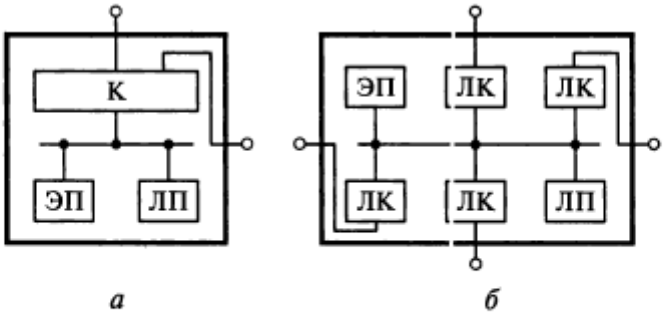


Рис. 6.6. Элементарная машина системы Cm*:

а — на базе контроллера; *б* — на базе локального коммутатора; К — контроллер; ЭП — элементарный процессор; ЛК — локальный коммутатор; ЛП — локальная память

Рис. 6.7. Каноническая структура системы Cm*:

ЭМ — элементарная машина

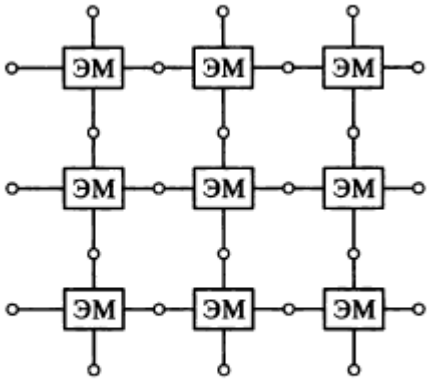


Рис. 6.8. Фрагмент двумерной структуры системы Cm*:

ЭМ — элементарная машина

Характерные особенности Cm*:

1. Реконфигурируемость - способность микроВС к априорной адаптации своего состава и структуры сети межмашинных связен к конкретной области применения.
2. Масштабируемость (наращиваемость) - способность микроВС к развитию в целях увеличения объема памяти, производительности или полос пропускания каналов связей.
3. Общедоступность и распределенность памяти. Память системы Cm* состояла из общей памяти локальной элементарных машин. Вся память системы была потенциально доступна для всех процессоров.
4. «Локальность» программы свойство, положенное в основу архитектуры микроВС Cm*. Эффективное функционирование системы достигалось, если большая часть программы и данных, с которыми работала каждая ЭМ, хранилась в ее локальной памяти.
5. Эффективность использования ресурсов системы обеспечивалась возможностью организации параллельной обработки исходных данных элементарными машинами и параллельных межмашинньх обменов информацией.
6. Экономичность ВС, т. е. удовлетворительные значения отношения “стоимость/производительность” результат применения большого числа недорогих серийных микропроцессоров.
7. Надежность функционирования микроВС достигалась за счет распределенности ее ресурсов. В системе не было критического ресурса, отказ которого приводил бы к отказу ВС в целом , аппаратурно-программные средства позволяли «удалять» из структуры неисправные компоненты. Контроль по четности, дистанционное диагностирование и

повторение команд обеспечивали обнаружение и исправление не только устойчивых, но и перемежающихся отказов аппаратуры.

О надежности Cm*
Для достижения надежного функционирования микроВС Cm* использовались специальные аппаратурно-программные средства. В микроВС Cm* реализован подход, основаный на введении в контролер отображения адреса специальной аппаратуры - ловушек (Hooks). Ловушка предоставляла микропроцессору (названному hooks-процессором) возможность тщательного исследования и изменения внутреннего состояния контроллера. Она позволяла загружать в управляющую память процессора микропрограмму, считывать значения сигналов на шинах и управлять генератором тактовых импульсов процессора.
Память была защищена контролем четности.

Контроль и диагностирование свободных от работы вычислительных модулей в микроВС Cm* выполнялся системой самодиагностики, представляющей собой последовательность из четырех диагностических программ:
1. Программа для диагностики памяти
2. Программа для диагностики системы команд
3. Программа для диагностики системы прерываний
4. Программа для диагностики системной аппаратуры

Анализ.
Очевидно, что вероятностная модель микроВС Cm * существенно сложнее модели для мини-ВС C.mmp, поэтому разработчиками были подвергнуты анализу лишь конкретные конфигурация системы. Количественный и качественный анализ убедил разработчиков ВС из Университета Карнеги Меллона в том, что с позиций надежности и живучести архитектура системы Cm* более перспективна, чем архитектура системы C.mmp (и конечно, систем семейства Burroughs).Главным недостатком микроВС Cm* являлось использование нераспределенного диспетчера, его отказ приводил к невозможности реализации функций ОС и, следовательно, к отказу системы как единого аппаратурно-программного ансамбля.

8.7 Кластерные ВС. Понятие о вычислительном кластере, архитектура кластерных ВС, ПО и области применения.
Одно из самых современных направлений в области создания вычислительных систем – это кластеризация. По производительности и коэффициенту готовности кластеризация представляет собой альтернативу симметричным мультипроцессорным системам. Кластер – это группа взаимно соединенных вычислительных систем (узлов), работающих совместно, составляя единый вычислительный ресурс и создавая иллюзию наличия единственной ВМ. В качестве узла кластера может выступать как однопроцессорная ВМ, так и ВС типа SMP или MPP. Каждый узел в состоянии функционировать самостоятельно и отдельно от кластера. Архитектура кластерных вычислений сводится к объединению нескольких узлов высокоскоростной сетью. Наряду с термином «кластерные вычисления» часто применяются такие названия, как: кластер рабочих станций (workstation cluster), гипервычисления (hypercomputing), параллельные вычисления на базе сети (network-based concurrent computing).
Перед кластерами ставятся две задачи:
- достичь большой вычислительной мощности;

- обеспечить повышенную надежность ВС.

Первый коммерческий кластер создан корпорацией DEC в начале 80-х годов прошлого века.

В качестве узлов кластеров могут использоваться как одинаковые ВС (гомогенные кластеры), так и разные (гетерогенные кластеры). По своей архитектуре кластерная ВС является слабо связанной системой.

Преимущества, достигаемые с помощью кластеризации:

- Абсолютная масштабируемость. Возможно создание больших кластеров, превосходящих по вычислительной мощности даже самые производительные одиночные ВМ. Кластер в состоянии содержать десятки узлов, каждый из которых представляет собой мультиплексор.
- Наращиваемая масштабируемость. Кластер строится так, что его можно наращивать, добавляя новые узлы небольшими порциями.
- Высокий коэффициент готовности. Поскольку каждый узел кластера – самостоятельная ВМ или ВС, отказ одного из узлов не приводит к потере работоспособности кластера. Во многих системах отказоустойчивость автоматически поддерживается программным обеспечением.
- Превосходное соотношение цена/производительность. Кластер любой производительности можно создать, соединяя стандартные ВМ, при этом его стоимость будет ниже, чем у одиночной ВМ с эквивалентной вычислительной мощностью.

На уровне аппаратного обеспечения кластер – это просто совокупность независимых вычислительных систем, объединенных сетью. При соединении машин в кластер почти всегда поддерживаются прямые межмашинные связи.

Существуют различные способы классификации кластеров. Простейший вариант основан на том, являются ли диски в кластере разделяемыми всеми узлами или нет. На рисунке показан кластер из двух узлов, совместная работа которых координируется за счет высокоскоростной линии, по которой происходит обмен сообщениями. Такой линией может быть локальная сеть, используемая также и не входящими в кластер компьютерами, либо выделенная линия. В последнем случае один или несколько узлов кластера будут иметь вы ход на локальную или глобальную сеть, благодаря чему обеспечивается связь между серверным кластером и удаленными клиентскими системами.

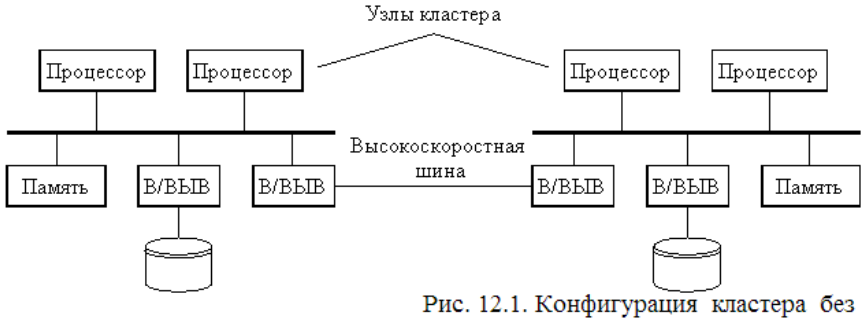


Рис. 12.1. Конфигурация кластера без совместно используемых дисков

Методы кластеризации Таблица 12.1.

Метод кластеризации	Описание
Пассивное резервирование	Вторичный сервер при отказе первичного берет управление на себя
Резервирование с активным вторичным сервером	Вторичный сервер, как и первичный, используется при решении задач
Самостоятельные серверы	Самостоятельные серверы имеют собственные диски, а данные постоянно копируются с первичного сервера на вторичный
Серверы с подключением ко всем дискам	Серверы подключены к одним и тем же дискам, но каждый сервер владеет своей их частью. Если один из серверов отказывает, то управление его дисками берет на себя другой сервер.
Серверы с совместно используемыми дисками	Множество серверов работают в режиме коллективного доступа к дискам.

Огромный потенциал масштабирования, свойственный кластерной архитектуре, делает ее очень перспективным направлением в области создания высокопроизводительных вычислительных систем. Масштабирование возможно как за счет увеличения числа узлов, так и путем применения в качестве узлов не одиночных ВМ, а также хорошо масштабируемых вычислительных систем, обычно SMP-типа.

8.8 Анализ мультипроцессорных ВС.

Анализ архитектуры мультипроцессорных ВС позволяет сделать нижеследующие выводы.

1. Основная тенденция в области архитектуры мультипроцессорных ВС повышение степени полноты воплощения принципов модели коллектива вычислителей (параллелизма, программируемости структуры и конструктивной однородности).
2. Архитектурные возможности ЭП неуклонно наращиваются, их структура претерпела трансформацию от простейших конфигураций без памяти до элементарных машин композиции из мощных микропроцессоров, оперативной памяти и внешних запоминающих устройств (и даже устройств ввода-вывода информации).
3. Мультипроцессорные ВС, начавшие свою историю как композиции из нескольких процессоров, превратились в системы с массовым параллелизмом.
4. Современные мультипроцессорные ВС -- это распределенные средства обработки информации, они имеют множество процессоров и распределенную память. Более того, в них и коммутатор (или другой ресурс), через который осуществляется взаимодействие процессоров, может быть распределенным. Программное обеспечение таких ВС также является распределенным.
5. Высокопроизводительные ВС рассматриваемого класса представляют собой суперсистемы: это множество мощных микропроцессоров-конвейеров или матричных процессоров или даже объединение мультипроцессоров.

9. Вычислительные системы с программируемой структурой

9.1 Понятие о вычислительных системах с программируемой структурой. Сосредоточенные и распределенные ВС.

Вычислительные системы с программируемой структурой - это распределенные средства обработки информации. В таких ВС нет единого функционально и конструктивно реализованного устройства: все компоненты (устройство управления, процессор и память) являются распределенными. Тип архитектуры ВС MIMD; в системах заложена возможность программной перенастройки архитектуры MIMD в архитектуры MISD или SIMD.
Основной функционально-структурной единицей вычислительных ресурсов в системах рассматриваемого класса является элементарная машина (ЭМ). Допускается конфигурирование ВС с произвольным числом ЭМ. Следовательно, ВС с программируемой структурой относятся к масштабируемым средствам обработки информации и допускают формирование конфигураций с массовым параллелизмом.

При построении ВС с программируемой структурой доминирующими являются следующие три принципа:
1) массовый параллелизм (параллельность выполнения большого числа операций);
2) программируемость (автоматическая перестраиваемость или реконфигурируемость) структуры;
3) конструктивная однородность.

Принцип программируемости структуры требует, чтобы в ВС была реализована возможность хранения программного описания функциональной структуры и программной ее модификации (перенастройки) с целью достигнуть адекватности структурам и параметрам решаемых задач. Под ВС с программируемой структурой понимается совокупность
элементарных машин, функциональное взаимодействие между которыми осуществляется через программно настраиваемую сеть связи.

В классе систем с программируемой структурой выделяют (пространственно) сосредоточенные и распределенные ВС. Характерной особенностью сосредоточенных ВС является компактное пространственное размещение средств обработки и хранения информации, при котором нет необходимости использовать телекоммуникационные сети. В сосредоточенных ВС нет линий связи, вносящих существенную задержку в работу ВС, нет жестких ограничений на топологию сети связи, на возможность параллельной передачи информации между функциональными модулями (процессорами, модулями памяти, ЭМ и др.).
Распределенная ВС - объединение пространственно удаленных друг от друга сосредоточенных ВС, основанное на принципах:
• параллельности функционирования сосредоточенных ВС (т. е. способности нескольких или всех сосредоточенных систем совместно и одновременно решать одну сложную задачу, представленную параллельной программой);
• программируемости структуры (т. е. возможности автоматически настраивать сеть связи между сосредоточенными ВС);
• гомогенности состава (т. е. программной совместимости различных сосредоточенных ВС и однотипности элементарных машин в каждой из них).

9.2 Архитектурные особенности ВС с программируемой структурой.

Как известно, структура ВС описывается графом G, множеству вершин которого сопоставлены ЭМ (или системные устройства, или локальные коммутаторы), а множеству ребер - линии межмашинных связей.
Требования, предъявляемые к структуре ВС:
1. Простота вложения параллельного алгоритма решения сложной задачи в структуру ВС. Структура ВС должна быть адекватна достаточно широкому классу решаемых задач; настройка проблемно-ориентированных виртуальных конфигураций и реализация основных схем обмена информацией между ЭМ не должны быть связаны со значительными
накладными расходами.
2. Удобство адресации элементарных машин и «переноса» подсистем в пределах ВС. Вычислительная система должна предоставлять возможность пользователям создавать параллельные программы с виртуальными адресами ЭМ.
3. Осуществимость принципа близкодействия и минимума задержек при межмашинных передачах информации в ВС.
4. Масштабируемость и большемасштабность структуры ВС.
5. Коммутируемость структуры ВС. ВС должна быть приспособлена к реализации групповых межмашинных обменов информацией. Следовательно, структура ВС должна обладать способностью осуществлять заданное число одновременных непересекающихся взаимодействии между элементарными машинами.
6. Живучесть структуры ВС. Важным требованием к ВС в целом является обеспечение работоспособности при отказе ее компонентов или даже подсистем.
7. Технологичность структур ВС. Структура сети межмашинных связей ВС не должна предъявлять особых требований к элементной базе.

Структурные характеристики.
Структура ВС представляет собой граф G (как правило, однородный для масштабируемых и большемасштабных ВС). Следовательно, структурные задержки пои передачах информации между машинами ВС определяются расстоянием (в смысле теории графов) между вершинами структуры, сопоставленными взаимодействующим машинам.
Характеристики структуры:

Для оценки структурных задержек в ВС используются диаметр d и средний диаметр $\bar{d}$ структуры. *Диаметр* — максимальное расстояние, определенное на множестве кратчайших путей между двумя вершинами структуры ВС:

$$d = \max_{i,j}\{d_{ij}\}, \tag{7.1}$$

средний диаметр

$$\bar{d} = (N-1)^{-1}\sum_{l=1}^{d} l n_l, \tag{7.2}$$

где d_{ij} — расстояние, т. е. минимальное число ребер, образующих путь из вершины i в вершину j; $i, j \in \{0, 1, \ldots, N-1\}$; n_l — число вершин, находя-

щихся на расстоянии l от любой выделенной вершины (однородного) графа G.

Показателем, оценивающим *структурную коммутируемость* ВС, является вектор-функция

$$\mathcal{K}(G, s, s') = \{\mathcal{K}_h(G, s, s')\}, \quad h \in \{1, 2, \ldots, [N/2]\}, \tag{7.3}$$

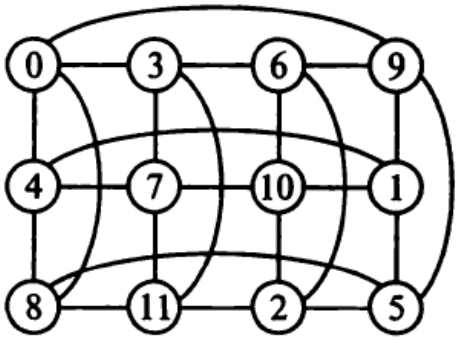
в которой координата $\mathcal{K}_h(G, s, s')$ есть вероятность реализации в системе при заданных структуре G и коэффициентах готовности s и s' соответственно одной ЭМ и линии связи h (см. § 2.8.4) одновременных непересекающихся межмашинных взаимодействий (обменов информацией между ЭМ); $[N/2]$ — целая часть числа $N/2$.

Структурная живучесть ВС оценивается вектор-функцией

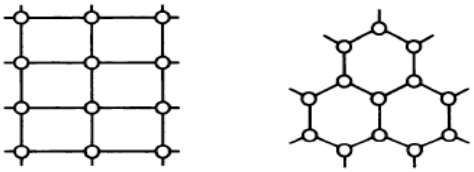
$$\mathcal{L}(G, s, s') = \{\mathcal{L}_r(G, s, s')\}, \quad r \in E_2^N = \{2, 3, \ldots, N\}. \tag{7.4}$$

Здесь $\mathcal{L}_r(G, s, s')$ — вероятность существования подсистемы ранга r (т. е. подмножества из r работоспособных ЭМ, связность которых устанавливается через работоспособные линии связи) при заданных структуре G, коэффициентах готовности s и s' ЭМ и линии связи соответственно.

Перспективные структуры ВС.
К таким относятся Dn-графы (циркулянтная структура).
{N;1, w2,…..,wn}
N – количество вершин или порядок графа. Вершины помечены числами i по модулю N, т.е. {0,1,2….,N-1}. Вершина i соединена ребрами i±w1, i±w2,……,i±wn (mod N), где w1…wn – множество образующихили целые числа, такие, что 0<w1<w2<….<wn<(N+1)/2, и для чисел N, и для всех w1….wn наибольшим общим делителем является 1.
n – размерность графа, 2n – степеь вершин в графе, N – порядок графа.



Целые числа i є [0….N-1] и отмечающие вершины Dn-графа называют адресами.
Адресация вершин в таких структурах называется диофантовой. В циркулянтных структурах при полном переносе какой-либо подструктуры сохраняются все ее свойства и адресация вершин. Полный перенос подразумевает смещение всех вершин структуры на одно и то же расстояние в одном из направлений.
В качестве структур ВС допускающих масштабирование без коренной коммутации уже имеющихся межмашинных связей используется Л(N,v,g)-граф. В такие графы вкладываются Dn-графы. По сути Л(N,v,g) – это неориентированный однородный граф с числом и степенями вершин соответственно N и v и обхватом g (длина кратчайшего цикла в графе называется обхватом).
В такие графы при N>=3 входит не менее v кратчайших простых графов длиной g.



а *б*

Рис. 7.3. Фрагменты $Л(N, v, g)$-графов:
$а$ — $v = 4$, $g = 4$; $б$ — $v = 3$, $g = 6$

Оптимальность структуры определяется не показателями ВС, которые выше были перечислены. Существуют алгоритмы для конкретных классов графов; для практических целей – создаются и пополняются каталоги оптимальных структур.

Режимы функционирования ВС.

1. Режим решения одной сложной задачи.

Все ресурсы системы выделены для выполнения одной параллельной программы. При решении таких задач, с точки зрения архитектуры, возникают следующие проблемы:

- Необходимо распараллелить программу
- Организовать отказо-устойчивое выполнение программы или задачи
- Эффективно вложить ветви параллельной программы в структуру системы (нужно ветви параллельной программы расположить в системе так, чтобы взаимодействие между ними происходило как можно быстрее)

2. Относится к мультипрограммным режимам, в которых все ресурсы делятся между несколькими выполняемыми одновременно параллельными программами.

Мультирежимы:

1. Режим мультирешения задач. Известно количество задач и все их параметры.
2. Режим обслуживания потока задач. В случае потока задачи поступают в случайные моменты времени и их параметры заранее неизвестны.

Проблемы: все задачи разные и могут не поместить в систему.

Способы обработки информации:

Различают распределенный, матричный и конвейерный способы обработки информации.

- При распределенной обработке параллельные программы и данные рассредотачиваются по элементарным машинам ВС.
- В случае матричной обработки данных программа вычислений содержится в одной (управляющей) ЭМ, а данные однородно распределяются по всем машинам ВС (или подсистемы). Процесс решения задачи состоит из чередующихся процедур: рассылки команд из управляющей ЭМ остальным машинам и исполнения этих команд всеми машинами, но каждой над своими операндами.
- Матричный способ в сравнении с распределенным дает экономию в использовании (распределенной по ЭМ) памяти ВС. Однако данному способу присущ недостаток, заключающийся в неоднородном использовании машин и, в частности, в неоднородной нагрузке на их память. Этого недостатка лишен обобщенный матричный способ обработки информации. При этом способе программа не целиком помещается в одной ЭМ, а предварительно сегментируется (не распараллеливается, а сегментируется!) и затем посегментно размещается в памяти машин.
- При конвейерном способе обработки данных структура ВС предварительно настраивается так, что машины образуют конвейер (или «линейку», или «кольцо»). Затем осуществляются сегментирование программы и размещение в машинах ВС

последовательности полученных сегментов в соответствии со структурой конвейера.
Размещение данных может быть сосредоточенным (например, на внешней памяти одной ЭМ) или распределенным
(по памятям всех машин конвейера). В процессе решения задачи данные проходят через последовательность машин, составляющих конвейер.

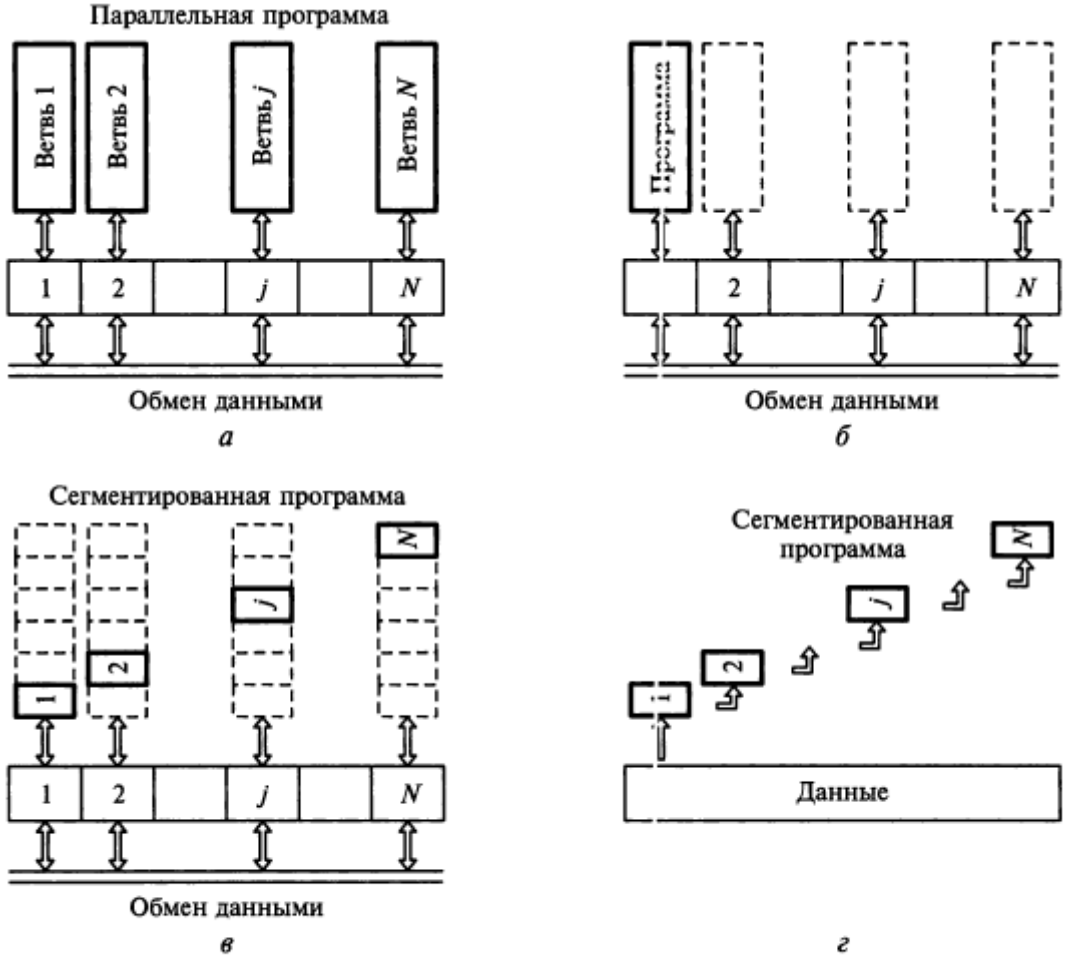


Рис. 7.7. Способы обработки информации:
а — распределенный; *б* — матричный; *в* — обобщенный матричный; *г* — конвейерный;
⇨ — направление потоков данных

Архитектурные аспекты.
Архитектура ВС должна удовлетворять главнейшему требованию - реализации в системе параллельных вычислений во всех режимах. В основе организации параллельных вычислений в ВС лежит представление вычислении в виде совокупности совместно протекающих асинхронных взаимодействующих процессов. Совместность процессов означает не только обычную для мультипрограммных систем (в частности, систем разделения времени) одновременность реализации алгоритмически независимых процессов (разделение ресурсов ВС), но и существование связи между отдельными процессами, которая обусловлена тем, что они представляют собой части одного сложного алгоритма (объединение ресурсов ВС).
Временное и пространственное распределение аппаратурно-программных ресурсов системы, программируемость структуры.

9.3 Вычислительная система «Минск-222».
Работы велись с 1965-66 гг. В 1966 – появился первый образец. Руководитель – Э.В. Евреинов.

Основные разработчики: Хорошевский, Сидристый, Лопато, Василевский. Первый образец был установлен в 1966 году в институте математики БССР. Архитектурная система – MIMD и распределенность ресурсов. Полностью реализованы принципы модели коллектива вычислителей: параллелизм, однородность, коллективность структуры. Топология: одномерная, кольцевая. Масштабируемость: от 1 до 16 элементраных машин (ЭВМ 2го поколения).
Функциональная структура:

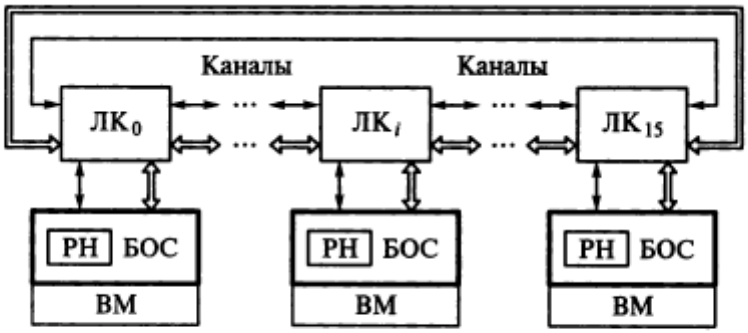


Рис. 7.8. Функциональная структура системы «Минск-222»:
ЛК — локальный коммутатор; БОС — блок операций системы; РН — регистр настройки; ВМ — вычислительный модуль; ⇨ — рабочий канал; → — управляющий канал

ЛК – локальный коммутатор
БОС – блок операции системы
ВМ – вычислительный модуль
РН – регистр настройки

Вычислительный модуль – машины МИНСК-2, МИНСК-22. Отличие между ними – обладала магнитной памятью удвоенной емкости:
МИНСК-22 : 8 килослов = 8192 слова с разрядностью 37, а также с дополнительным набором ввода/вывода.
ЛК представляли из себя вентили, которые открывали/закрывали канал связи, идущий к машине справа.
РН входил в состав блока операции системы, определяет соединительные функции коммутатора и степень участия элементарной машины при системных взаимодействиях. Состоит из трех разрядов: TR, TQ и TΩ. TR – разбивает систему на подсистемы: 1 – связь есть, 0 – связи нет. TR – R0,R1….R15, т.к. машина разбивалась на несколько подсистем, TQ и TΩ - конкретизируют участие машины в выполнении систем команд.

Триггеры TQ и $T\Omega$ конкретизировали степень участия машин в выполнении некоторых системных команд. В частности, триггеры $T\Omega$ использовали для выделения машин, участвовавших в выработке обобщенного признака Ω_k $(k = 1, 2, 3)$, который управлял ходом вычислений. Признак

$$\Omega_k = \underset{i \in E}{\&} \omega_{ki}, \quad k = 1, 2, 3, \tag{7.6}$$

где E — подмножество номеров машин, управлявших ходом вычислений (т. е. отмеченных единицей в разряде $T\Omega$), $E \subseteq \{0, 1, \ldots, N-1\}$; ω_{ki} — признаки, вырабатываемые ЭМ с номером i, причем $\omega_{1i} = 1$, если знак последнего вычислительного результата был меньше нуля, $\omega_{2i} = 1$, если происходило переполнение, $\omega_{3i} = 1$, если последний вычислительный результат арифметико-логического устройства был равен нулю. В конфликтных ситуациях приоритет имела машина с меньшим номером.

Система команд МИНСК -222.
Была расширена командами настройки, обеспечивающими программируемость структуры ,а аткже обеспечивает обмен информацией и командами обобщенного условного и безусловного переходов.
Схема команды. Общая структура состоит из пяти блоков:

0	1…..	7……	13…..	25…..
	6	12	24	36

+ -	КОП	Θ	A1	A2
1	2	3	4	5

1 – знак
2 – код операции
3 – 6-разрядное поле, 2 разряда которых определяли номер блока памяти, а 4 оставшиеся – адрес (индекс) ячейки.

Команды настройки:

1	Ø1	ØØ	A1	A2

Имели три модификации: Н0, Н1, Н2, которые отличались в зависимости от значения 29-го и 34-го бита.
Н0: 29й=0, 34й=0: команда изменяет содержание регистра настройки только своей ЭМ, при этом новые значения регистра настройки задаются в TR=31, TQ=32, TΩ=33.
Н1: 29й=1: изменяет содержимое регистров настроек (РН) тех ЭМ, которые указаны с 13й по 28й разряд, при этом номер машины считается как номер разряда минус 13 (чтобы удаленно управлять другими машинами). Содержание своих РН не меняется.
Н2: 29=0, 34 = 1: изменяет содержимое РН только своей ЭМ и изменяет значение регистра округления в 35-м и признака w3i в 36-м разряде.

Команда обмена.

Передача	1	56	ØØ	l	α
Прием	1	57	ØØ	h	β

l – определяла количество передаваемых слов, h – количество принимаемых слов
α – указатель начала буфера (откуда слова начинают считывать), β – начало буфера, куда записывать.
Команда приема является блокирующей, т.е. не будет завершена, пока не будет принято h слов.

Команды обобщенного безусловного перехода.

ОБП (общий вид)	1	02	a_7Ø	ØØ	A2
ОБПØ, a_7=0	1	02	ØØ	ØØ	A2
ОБП1, a_7=1	1	02	1Ø	ØØ	A2

При выполнении ОБПØ следующая инструкция выполняется на всех машинах после окончания текущей (не вытесняющая), используется для синхронизации выполнения команд.
ОБП1 – вытесняющая и выполняется сразу же. С помощью этих команд может быть передана любая команда из любой машины. Таким образом, удаленно с других машин может быть изменен ход вычислительного процесса.

Команды обобщенного условного перехода.

ОУП	1	65	ØØ	A1	A2

Четыре модификации ОУПk, k=0,1,2,3, которые отличаются значением вектора a13a14a15.
Выполняется только в тех машинах, в которых в регистре настройки содержимое Ω=1.
ОУПØ – переход к следующей команде, синхронизация.
ОУП1,2,3 – происходит передача управления либо по адресу, указанному в A2, либо к следующей команде.
ОУП1 - w1 < Ø
ОУП2 – w2 переполнено
ОУП3 – w3=0
Значение A1 определяет класс операции (A1=0,1,2,3).

ПО ВС МИНСК – 222
Расширенная ПО ЭВМ Минск-222 можно разделить на две части:
1. Система параллельного программирования
2. Пакет прикладных адаптирующихся параллельных программ.

Система параллельного программирования состоит из:
1. Средства автоматизации параллельного программирования, использующее адаптированные языки ALGOL, BASIC.
2. Средства отладки и редактирования параллельных программ

3. Средства анализа параллельных программ.

Пакеты прикладных адаптирующихся параллельных программ ориентированы на решения задач повышенной сложности. Под адаптирующимися параллельными программами понимаются такие программы, которые могли настраиваться на количество машин ВС как на параметры:

1. Задача линейной алгебры и математического программирования
2. Пакеты решения систем дифференционных уравнений
3. Пакеты решения информационно-логических задач
4. Пакеты решения задач статистического моделирования сложных задач.

Использовалась при решении математических задач: численное интегрирование, дифференцирование, решение систем дифференциальных уравнений методом Рунге-Кутта, методом Зейделя, решение задачи Коши и многое другое.

Были использованы все схемы обмена информацией –ТО, ТЦО, ДО, КП и т.д., эффективность и быстродействие были значительно выше, чем у «Минск-22» за счет быстродействия каналов связей и большей емкости оперативной памяти.
Следует обратить особое внимание на парадокс параллелизма (нелинейный рост производительности ВС при повышении количества N ЭМ), что противоречит якобы здравому смыслу. Парадокс параллелизма был впервые обнаружен при работе на ВС <Минск-222».

9.4 Вычислительная система МИНИМАКС.

ВС МИНИМАКС

Функциональная структура мини-ВС МИНИМАКС - композиция из произвольного количества элементарных машин и программно настраиваемой сети связей между ними. Мини-ВС МИНИМАКС обладала свойством масштабируемости: в ней количество ЭМ не было фиксировано и определялось сферой применения.

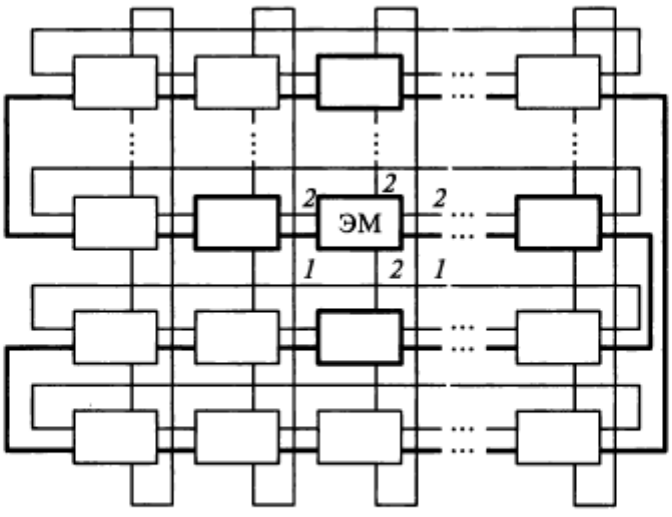


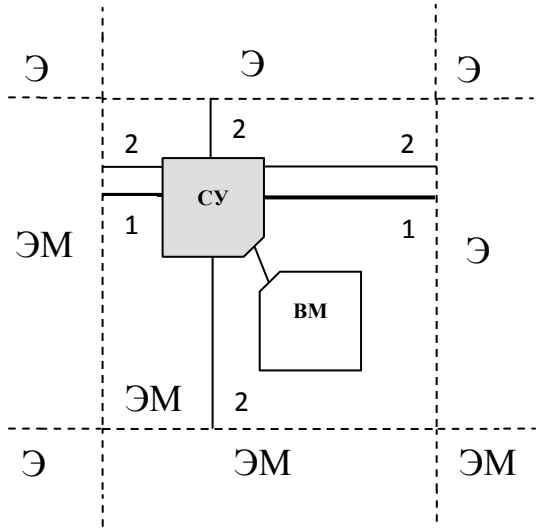
Рис. 7.10. Функциональная структура мини-ВС МИНИМАКС:

ЭМ — элементарная машина; *1* — одномерные управляющие каналы; *2* — двумерные рабочие каналы

Связи:

1-двумерные рабочие каналы(использовались для обмена данными)

2-одномерные управляющие каналы. В качестве двумерных каналов использовались 2д графы(Д2-граф), модификация которого зависела от кол-ства ЭМ в системе, т.е. от требуемой мощности.



СУ - системное устройство
ВМ - вычислительный модуль
Системное устройство было спроектировано как автономное устройство АСВТ-М (агрегатное средство ВТ на микроэлектронной основе). Оно подключалось к ВМ через связи 3.

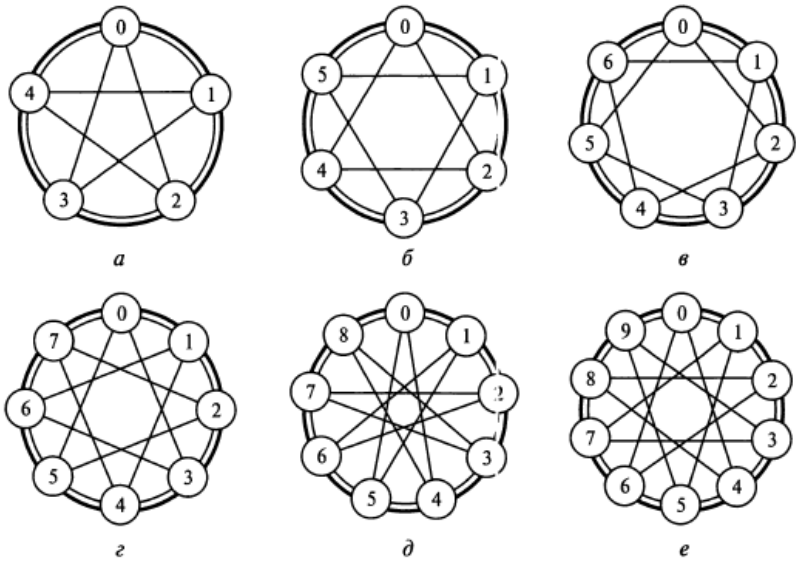
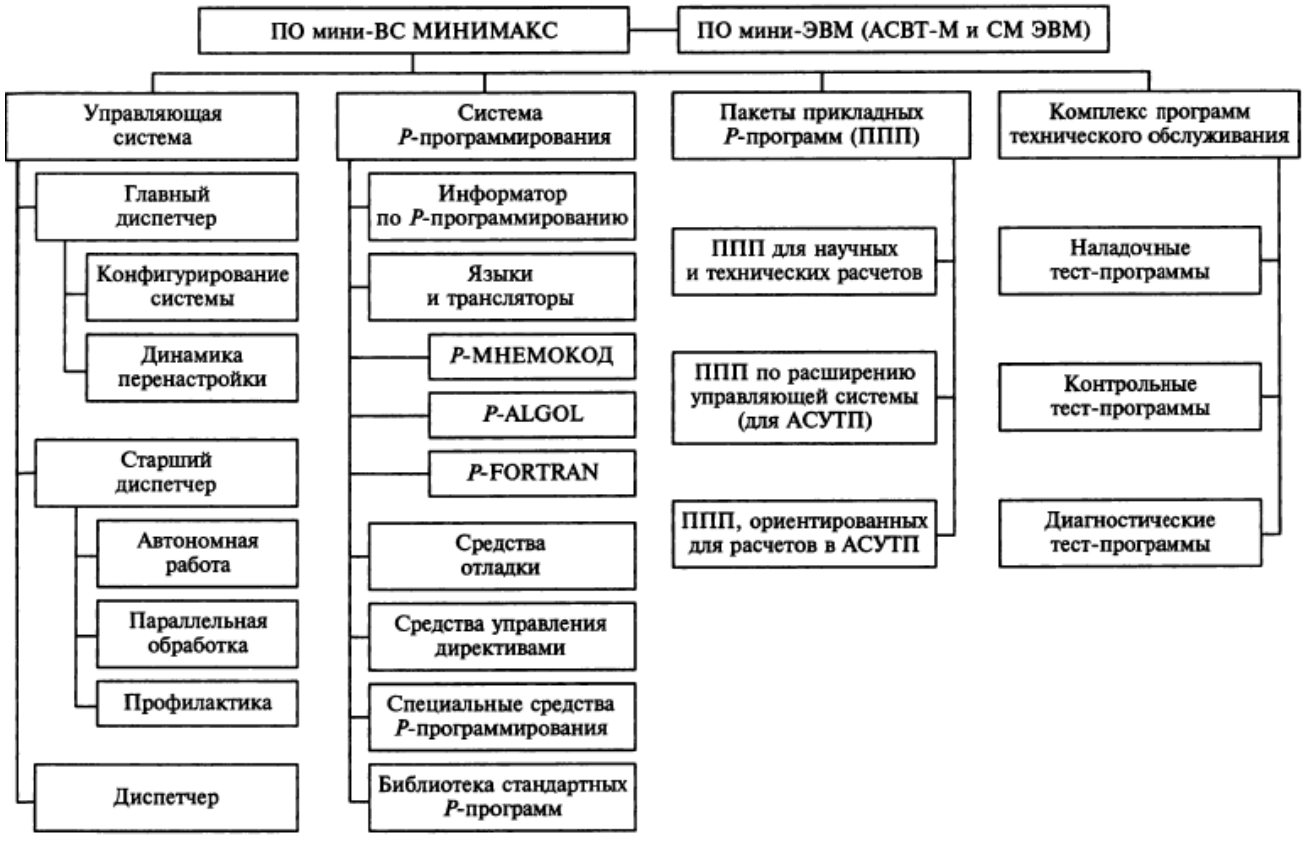


Рис. 7.11. Оптимальные структуры мини-ВС МИНИМАКС, D_2-графы вида:
$а$ — {5; 1,2}; $б$ — {6; 1,2}; $в$ — {7; 1,2}; $г$ — {8; 1,3}; $д$ — {9; 1,4}; $е$ — {10; 1,4}

Системные команды мини-ВС МИНИМАКС.
1.Настройка. Подразделялась на настройку машины и настройку системы. Настройка машины проводилась с помощью одной из двух команд. Первая осуществляла занесение кода, заданного в команде, на регистр настройки, вторая - перепись содержимого регистра Р 1 на регистр настройки. Настройка системы выполнялась при помощи четырех команд, определявших направление настройки

(вправо или влево от настраивающей ЭМ) и ее характер (занесение кода настройки на регистр настройки или Р1).
2. Обмен. Осуществлялся при помощи команд передачи и приема.
3. Обобщенный безусловный переход. Использовался для передачи управления в машинах по адресу, поступавшему из управляющей машины. Для реализации обобщенного безусловного перехода использовались команды передачи адреса и приема адреса.
4. Синхронизация работы машин. Осуществлялась с помощью специальной команды. Машины, выполнявшие эту команду, вырабатывали значения индивидуальных признаков ω=1. Синхронизация достигалась, когда все синхронизируемые машины вырабатывали значения обобщенного признака Ω =1.
5. Обобщенный условный переход. Реализовывался по специальной команде. Эту команду обязаны были выполнить все машины, в которых требовалось осуществить условную передачу управления по значению обобщенного признака Ω (конъюнкции индивидуальных признаков ω).

Программное обеспечение МИНИМАКС.



Применение.
Вычислительные системы МИНИМАКС могли применяться:
• в химической, нефтехимической, нефтеперерабатывающей, металлургической, металлообрабатывающей, приборостроительной, радиотехнической, электронной и других отраслях промышленности;
• на тепловых и атомных электростанциях и в системах энергоснабжения;
• в научно-исследовательских учреждениях, занимающихся исследованием проблем физики, гидро-аэродинамики, химии, биологии, медицины и др.
Системы МИНИМАКС использовались:
• для решения научных, экономических и технических задач;

• для сбора и первичной переработки информации в сложных иерархических системах;
• для управления научными экспериментами;
• для исследования технологических процессов и расчета технико-экономических показателей;
• для управления технологическими процессами;
• для контроля качества промышленной продукции (полупроводниковых приборов, электронных схем и др.);
• в качестве центра коммутации сообщений.

9.5 Вычислительная система СУММА.
Система СУММА была разработана ИМ СО АН СССР(отделом ВС) совместно с производственным объединением «К-варц». Министерство электронной промышленности СССР (г. Калининград).
Техническое проектирование ВС было выполнено в 1974 году опытно-промышленные образец был изготовлен в 1976 году.
Мини-ВС СУММА формировалась из ЭМ-трехполюсников (Л(N,3,g)-графы), количество которых не было фиксировано. Система характеризовалась большой архитектурной гибкостью. Ее можно было легко расширить или сократить в соответствии с предъявляемыми требованиями.

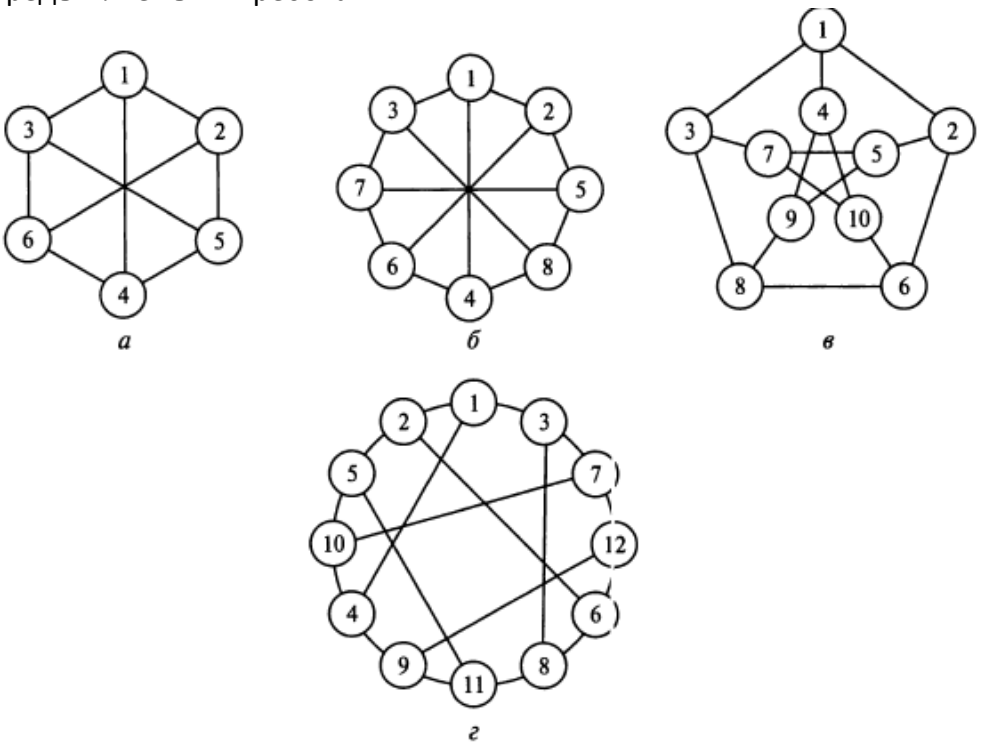


Рис. 7.15. Оптимальные структуры мини-ВС СУММА:
а — $L(6, 3, 4)$; *б* — $L(8, 3, 4)$; *в* — $L(10, 3, 5)$; *г* — $L(12, 3, 5)$

Элементарная машина системы СУММА формировалась как «трехполюсник», или, точнее, композиция из ВМ и СУ, рассчитанного на три межмашинные связи:

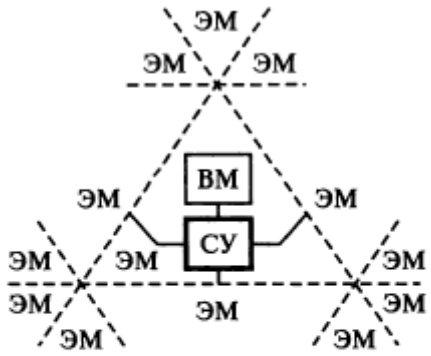


Рис. 7.16. Функциональная структура ЭМ мини-ВС СУММА:

ЭМ — элементарная машина; ВМ — вычислительный модуль; СУ — системное устройство

ВМ – вычислительный модуль, СУ – системное устройство
Вычислительный модуль предназначался для выполнения всех операций, связанных с переработкой информации, в частности для инициирования реализации системных операций. Системное устройство использовалось для реализации системных взаимодействий машин, в частности для программирования структуры мини -ВС. Системное устройство конструктивно было оформлено в виде отдельного модуля. К мини-ЭВМ оно подключалось через общую шину (как и внешнее устройство), а к СУ трех соседних ЭМ: через каналы межмашинной связи.

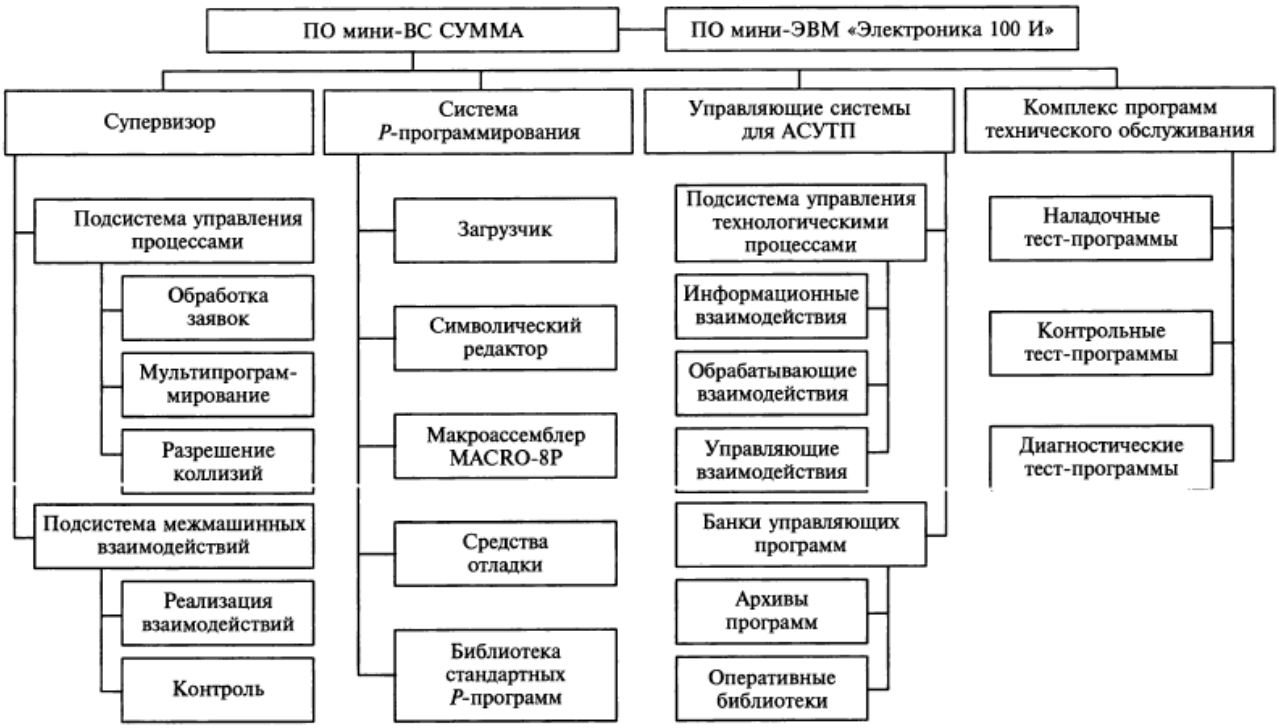
ПО ВС СУММА было ориентировано на управление процессами в реальном масштабе времени. Вместе с тем оно содержало компоненты, позволяющие использовать ВС в режиме общего назначения.

Архитектура системы СУММА:
– MIMD-архитектура;
– распределённость средств управления, обработки и памяти;
– параллелизм, однородность, модульность;
– программируемость структуры;
– масштабируемость;
– живучесть;
– единый канал для управляющей и рабочей информации;
– аппаратурно-программная реализация системных взаимодействий.

Системные команды мини-ВС СУММА были разделены на три группы. Рассмотрим подробно функциональные особенности реализации команд этих групп. Первую группу составляли команды обращения из мини-ЭВМ в собственное СУ. Эти команды являлись командами обращения к внешним устройствам (но с селекторным кодом, присвоенным СУ).
Команды второй и третьей групп выполнялись совместно и позволяли осуществить обмен информацией между любыми ЭМ подсистемы. Процесс передачи, инициированный передающей ЭМ, начинался с «захвата» собственного СУ. Применялось два режима захвата мягкий и жесткий. При мягком режиме процессор устанавливал в СУ заявку на обслуживание и ждал освобождения СУ от текущей работы. При жестком режиме захват СУ происходил независимо от текущего состояния СУ. Передача процессором

необходимой информации в СУ осуществлялась лишь после подтверждения, что захват СУ произошел.

ПО.
Супервизор являлся резидентной программой управления процессами в реальном масштабе времени. Он состоял из подсистем управления процессами и межмашинных взаимодействии.



Применение мини-ВС СУММА было эффективно и при решении широкого класса задач, представленных параллельными программами. Кроме того, она могла быть использована в качестве вычислительного ядра «интегрированных» АСУТП (автоматизированная система управления технологическими процессами).

9.6 Вычислительная система МИКРОС.
Модель реализовалась в следующих модификациях МИКРОС-1 1986, МИКРОС – 2 1992, МИКРОС – Т 1996. В качестве ЭМ МИКРОС 1 и МИКРОС 2 использовались Электроника 60-М и Электроника 60-1, а также спецпроцессоры Электроника МТ-70 и 16-03.
ЭМ состояла из центрального профессора(ЦП), оперативной памяти(ОП), модульного системного устройства(МСУ). В составе конфигураций допускалось до 4х МСУ.

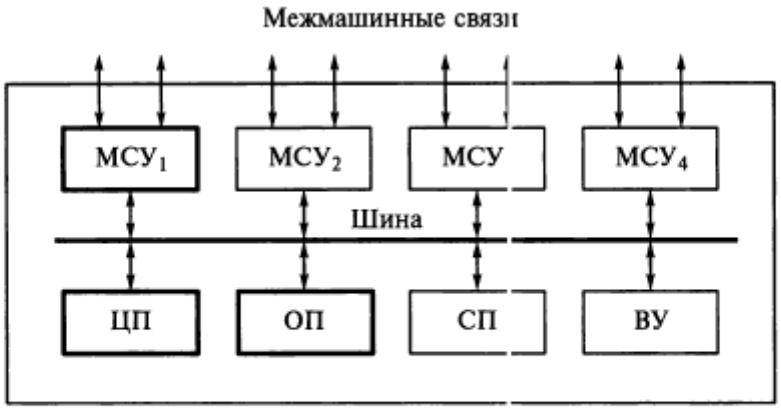


Рис. 7.21. Функциональная структура ЭМ систем МИКРОС-1 и МИКРОС-2:

МСУ — модуль системного устройства; ЦП — центральный процессор; ОП — оперативная память; СП — специальный процессор; ВУ — внешнее устройство

МСУ позволял использовать в качестве каналов различные средства, в частности, экранированные провода (при расстоянии между ЭМ до 30 м), либо радиочастотные кабели (если расстояние между ЭМ не превышало 300 м), либо коммутируемые или выделенные телефонные каналы связи (с использованием аппаратуры передачи данных независимо от расстояния между ЭМ). Заложенная в МСУ схема обеспечения связности машин была равно пригодна для формирования как сосредоточенных, так и пространственно распределённых вычислительных систем.

Для внутреннего взаимодействия использовалась внутренняя системная шина. Структуры использовались различные Dn графы, так и L(N,v,g) графы.

ЭМ МИКРОС Т отличались от МИКРОС 1 и МИКРОС 2

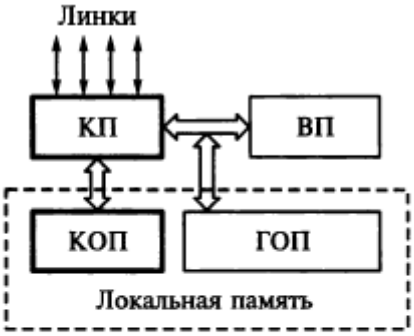


Рис. 7.22. Функциональная структура ЭМ МИКРОС-Т:

КП — коммуникационный процессор; ВП — вычислительный микропроцессор; КОП — коммуникационная память; ГОП — главная оперативная память

Элементарная машина системы МИКРОС-Т
У МИКРОС Т заложена возможность использовать не только Dn графы, а также произвольные, нерегулярные графы.

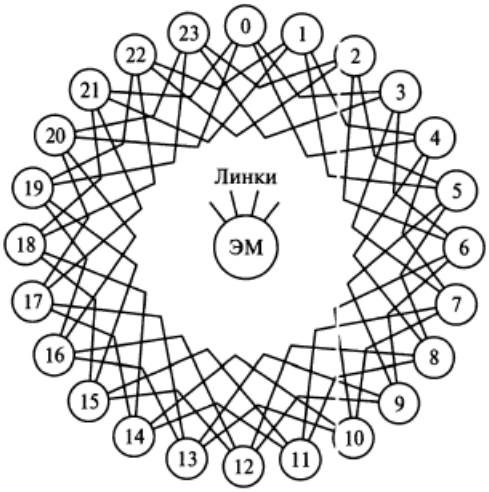


Рис. 7.23. Оптимальная структура 24-машинной ВС семейства МИКРОС в виде D_2-графа {24; 3, 4}:
ЭМ — элементарная машина

МИКРОС Т является транспьютерной ВС. Транспьютер = транзистор+компьютер (компьютер на транзисторах).
В качестве ВП использовался INTEL 860, процессоры ALPHA.
Система МИКРОС-Т базируется на транспьютерных технологиях* (транспьютер – ЭМ-четырехполюсник в интегральном исполнении). Такие технологии позволяют формировать двумерные ВС с массовым параллелизмом. двумерные структуры ВС формируются путем отождествления полюсов-линков (Link-связь).
Простейшая конфигурация ЭМ представляется транспьютером (например, Inmos T 805) с памятью, развитые конфигурации ЭМ могут включать в себя: высокопроизводительные микропроцессоры Intel 860 (компания Intel), PowerPC (альянс компаний IBM, Apple и Motorola), Alpha (компании DEC и Сотраq) и др. Для формирования ЭМ системы МИКРОС-Т
могут быть использованы стандартные решения зарубежных и отечественных фирм-производителей транспьютерных модулей.

ПО МИКРОС Т – расширенное, включает в себя ОС МИКРОС, средства самодиагностики, средства формирования подсистем, маршрутизация, средство загрузки, параллельные программ, средство динамического управления нагрузкой ЭМ, средство поддержки отказоустойчивости и интерпретатор языка управления параллельным вычислениями (ЯУПВ).

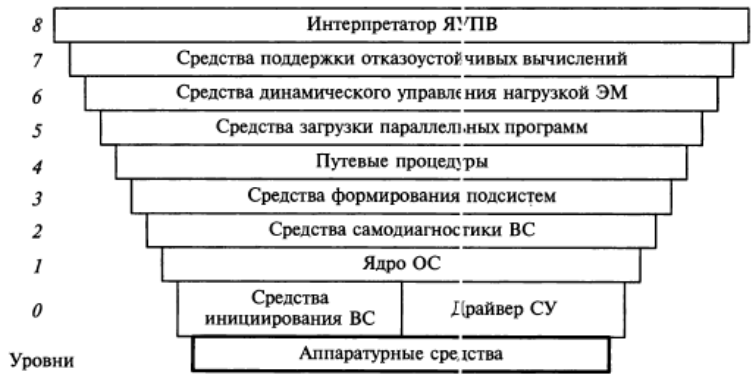


Рис. 7.24. Операционная система МИКРОС-Т

Нижний (нулевой) уровень ОС МИКР0С-Т включает средства инициирования работы ВС, приводящие, в частп:ости, все ЭМ в исходное состояние, и драйвер СУ, используемый для организации обменов данными и командами между соседними ЭМ (т. е. машинами, непосредственно соединенными линком). Следующий уровень 1 (ядро ОС) интерпретирует примитивы ОС, предназначенные для динамического распределения памяти ЭМ, а также для порождения, уничтожения процессов и организации их взаимодействия (синхронизации и обмена данным) в пределах одной машины. В совокупности с драйвером СУ ядро ОС позволяет порождать и уничтожать процессы в соседней машине и выполнять взаимодействия процессов, протекающих в этих машинах.
Средства формирования виртуальных подсистем (уровень 3) используются в мультипрограммных режимах работы системы. Они выделяют машины, входящие в подсистему, посредством созданных в этих машинах «окружений». Окружение содержит информацию о принадлежности машины подсистеме и другие параметры, в частности задающие вид связности машин: «линейку», «кольцо», «дерево», «решетку» и др.
Значения элементов окружения используются путевыми процедурами (уровень 4) при выполнении обменов между машинами подсистемы. Набор путевых процедур предназначается для реализации схем межмашинньх обменов в пределах подсистем и всей системы в целом. Фактически путевые процедуры распространяют функции ядра ОС на всю систему.
После формирования окружений в машины подсистемы загружаются предназначенные им программы и данные. загрузка машин осуществляется с помощью специальных средств уровня 5.
Уровни 0-5 (за исключением программы инициирования) составляют резидентную часть ОС, содержащуюся в каждой ЭМ вычислительной системы.
Средства динамического управления нагрузкой ЭМ (уровень 6) осуществляют перераспределение программ и данных между ЭМ подсистемы по завершении ее формирования или реконфигурации. На уровне 7 поддержки отказоустойчивых вычислений, выполняются операции, связанные с перезапуском параллельных процессов вычислении с заданных точек возврата.
Интерпретатор языка управления параллельными вычислениями (ЯУПВ, уровень 8) по командам с терминала порождает процессы, осуществляющие: генерацию подсистемы необходимого типа («дерево», «линейка», «кольцо» и т. п.) из требуемого числа работоспособных ЭМ; загрузку параллельной программы в сформированную подсистему и инициирование ее выполнения.

Опишем архитектурные свойства ВС семейства МИКРОС. Класс архитектуры любой модели ВС это MIMD; допустима трансформация архитектуры MIMD в архитектуру MISD или SIMD путем
программной перенастроики системы.
Класс ВС - система с программируемой структурой и с распределенным управлением.
Характер пространственного размещения вычислительных ресурсов - сосредоточенный или распределенный.
Основная функционально-структурная единица вычислительных ресурсов - ЭМ .
Функции ЭМ традиционные для ЭВМ функции по переработке информации плюс функции, связанные с управлением ВС в целом как коллектива (ансамбля) машин.
Количество N элементарных машин не фиксировано (масштабируемость).
Тип оперативной памяти распределенная и общедоступная.

9.7 Вычислительные системы семейства МВС.
Модели МВС-100 МВС-1000.
Созданы в НИИ «КВАНТ» Москва в содружестве с институтами РАН. Руководитель работ – В,К, Левин. Семейство МВС является промышленным решением ВС МИКРОС.
МВС–100 выпускались в 1992-96г.
МВС-1000 выпускались в 1997-2000г.

Функциональная структура аналогична МИКРОС Т. Структурная организация:
Появляется понятие структурный модуль – матрица 4х4 связанных ЭМ, граничные линки модуля использовались следующим образом:

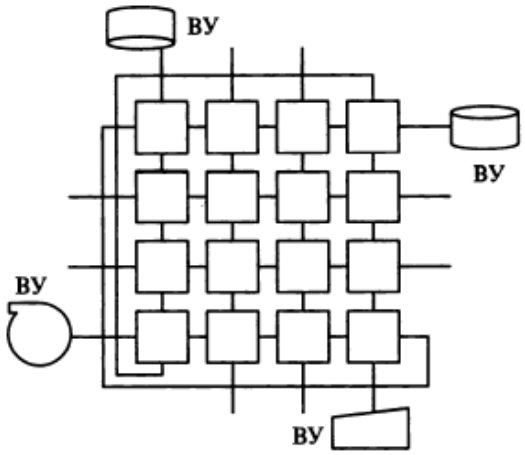


Рис. 7.25. Структурный модуль ВС семейства МВС:
ВУ — внешнее устройство

КП – коммуникационный процессор, ВП – вычислительный процессор.

При формировании ЭМ использовались:
Для МВС-100: ВП: Intel – 860, Power PC
КП: транспьютер INMOC-T425

Для МВС – 1000: ВП: ALPHA 31164
КП: TMS – 320C44 SHARCADSP 21060

Структурный модуль ВС семейства МВС

- линки использовались для организации диагональных связей
- оставшиеся 4 линка угловых машин – для подсоединения хост-компьютеров (host – ведущая машина) и внешних устройств (ВУ) и для связи с ЭМ других модулей
- Оставшиеся восемь линков – для соединений с подобными структурными модулями.

Данная структура позволяет снизить диаметр.

Конструкция и управление вычислительной системой семейства МВС.

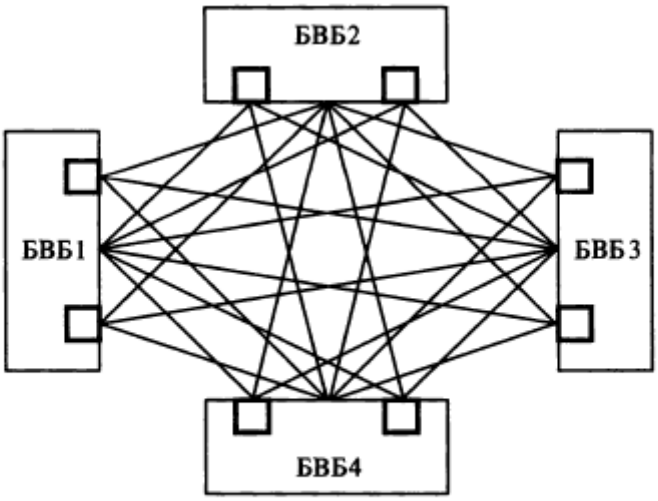


Рис. 7.28. Структура 128-машинной ВС семейства МВС:
БВБ — базовый вычислительный блок; □ — коммуникационный процессор

ПО МВС.
Софт – UNIX-подобные системы.
Операционная система выполняет функции по оптимальному использованию ресурсов ВС (коллектива ЭМ, средств ввода-вывода информации) и обеспечивает доступ пользователей. Операционные системы Digital Unix (Tru64 Unix) для AlphaStation и Linux для ЭВМ PC устанавливаются в хост-компьютерах. Средства программирования представлены языками и компиляторами FORTRAN 77, C и C++.

Приведем список задач, параллельные алгоритмы решения которых эффективно реализуются на ВС семейства МВС :
1)задачи расчета аэродинамики летательных аппаратов, в том числе интерференции при групповом движении;
2) расчет трехмерных нестационарных течений вязкосжимаемого газа;
3) расчет течений с локальными тепловыми неоднородностями в потоке;
4) квантовая статистика поведения вещества при экстремальных условиях;
5) структурообразование биологических макромолекул;
6) моделирование динамики молекулярных и биомолекулярных систем;
7) дифференциальные игры, динамические задачи конфликтов управления;
8) механика деформируемых твердых тел (с учетом процессов разрушения).

9.8 Анализ вычислительных систем с программируемой структурой.
Вычислительные системы с программируемой структурой - гибкий класс средств обработки информации с архитектурой MIMD. Архитектура таких систем не имеет принципиальных ограничений на пути ко все более полному воплощению в реализациях принципов модели коллектива вычислителей как на макроуровне (на уровне системы в целом), так и на
микроуровне (на уровне одной ЭМ). Концепция ВС с программируемой структурой свидетельствует о революционном отходе от классической архитектуры ЭВМ Дж. фон Неймана.
Вычислительные системы с программируемой структурой – это коллектив ЭМ, количество которых и структур; а сети связей между которыми допускают варьирования в широких пределах. Рост производительности таких ВС обеспечивается увеличением количества ЭМ и расширением конфигураций каждой из них.