



Сергей Немнюгин
Ольга Стесик

Параллельное программирование для многопроцессорных вычислительных систем

- Архитектура высокопроизводительных вычислительных систем
- Модели и методы параллельного программирования
- Разработка и отладка параллельных программ



МАСТЕР ПРОГРАММ

Сергей Немнюгин
Ольга Стесик

Параллельное программирование для многопроцессорных вычислительных систем

Санкт-Петербург

«БХВ-Петербург»

2002

УДК 681.3.06
ББК 32.973.26-018.1
Н50

Немнюгин С. А., Стесик О. Л.

Н50 Параллельное программирование для многопроцессорных
вычислительных систем. — СПб.: БХВ-Петербург, 2002. — 400 с.: ил.

ISBN 5-94157-188-7

Книга является практическим руководством для разработки прикладного программного обеспечения параллельных многопроцессорных систем. Приводятся сведения об архитектуре высокопроизводительных систем параллельного программирования — MPI (Message Passing Interface), PVM (Parallel Virtual Machine), HPF (High Performance Fortran). Излагается методика параллельного программирования для создания своих эффективных параллельных (и векторизованных) программ. Представленные примеры помогут разобраться в тонкостях работы многопроцессорных систем, а задания-упражнения для самостоятельной работы — закрепить изложенный материал. В приложениях описываются способы отладки параллельных кластеров, методы исследования производительности, дан обзор средств визуализации исполнения многопроцессорных приложений.

*Для программистов, преподавателей и студентов,
чья деятельность связана с высокопроизводительными вычислениями*

УДК 681.3.06
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Анатолий Адаменко</i>
Зав. редакцией	<i>Анна Кузьмина</i>
Редактор	<i>Вячеслав Кашистанов</i>
Компьютерная верстка	<i>Ольги Сергеенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Оформление серии	<i>Via Design</i>
Дизайн обложки	<i>Игоря Цырульниковца</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 05.09.02.

Формат 70×100¹/₁₆. Печать офсетная. Усл. печ. л. 32,25.

Тираж 3000 экз. Заказ №

"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-188-7

© Немнюгин С. А., Стесик О. Л., 2002
© Оформление, издательство "БХВ-Петербург", 2002

Содержание

- Предисловие..... 11**
- Глава 1. Архитектура высокопроизводительных ЭВМ..... 17**
 - Классификация Флинна..... 18
 - SISD-компьютеры 19
 - SIMD-компьютеры..... 19
 - MISD-компьютеры..... 21
 - MIMD-компьютеры 21
 - Традиционная архитектура фон Неймана 24
 - Регистры 25
 - Выполнение команды..... 25
 - Набор команд процессора..... 26
 - Шины..... 27
 - Память 27
 - Виртуальная память..... 30
 - Устройства ввода/вывода..... 31
 - Основные элементы архитектуры высокопроизводительных
вычислительных систем..... 31
 - Процессоры..... 32
 - Конвейеры 33
 - Суперскалярные процессоры..... 36
 - Процессоры с сокращенным набором команд (RISC)..... 38
 - Процессоры со сверхдлинным командным словом..... 40
 - Векторная обработка данных..... 42
 - Процессоры для параллельных компьютеров..... 45
 - Процессоры Pentium..... 46
 - Процессор Compaq Alpha EV67/68..... 46
 - Многопоточная архитектура 46

Оперативная память	48
Разделяемая память	49
Чередуемая память.....	50
Распределенная память	50
Связь между элементами параллельных вычислительных систем	51
Статические топологии	52
Маршрутизация.....	55
Динамические топологии.....	55
Многокаскадные сети.....	55
Методы коммутации (переключения)	59
Схемы классификации архитектур параллельных компьютеров.....	60
Классификация по способу взаимодействия процессоров с оперативной памятью	60
Схема Хандлера (эрлангерская схема)	61
Основные типы архитектур высокопроизводительных вычислительных систем.....	62
SIMD-архитектуры с разделяемой памятью.....	62
SIMD-машины с распределенной памятью	62
MIMD-машины с разделяемой памятью.....	63
MIMD-машины с распределенной памятью.....	64
Архитектура ccNUMA.....	65
Кластеры рабочих станций.....	66
Мультипроцессорные и мультимикрокомпьютерные системы.....	67
Симметричные многопроцессорные системы.....	68
Примеры архитектур суперкомпьютеров	68
Архитектура суперкомпьютера NEC SX-4	71
Compaq AlphaServer SC	72
Архитектура суперкомпьютера Cray SX-6.....	73
Cray T3E (1350)	73
Cray MTA-2.....	74
Глава 2. Особенности программирования параллельных вычислений.....	77
Последовательная и параллельная модели программирования.....	77
Другие модели параллельного программирования	81
Передача сообщений	82
Параллелизм данных	82
Модель разделяемой памяти.....	82
Закон Амдала	83
Две парадигмы параллельного программирования.....	84
Параллелизм данных	85
Управление данными.....	86
Операции над массивами.....	86
Условные операции	86
Операции приведения	86
Операции сдвига	87
Операции сканирования	87

Операции пересылки данных	87
Программирование в модели параллелизма данных	87
Параллелизм задач	88
Разработка параллельного алгоритма	88
Декомпозиция (сегментирование)	90
Проектирование коммуникаций	94
Укрупнение	96
Планирование вычислений	96
Количественные характеристики быстродействия	99
Программные средства высокопроизводительных вычислений	100

Глава 3. Введение в параллельное программирование с использованием MPI..... 105

Что такое MPI?	105
Операции обмена сообщениями	108
MPI — "Интерфейс Передачи Сообщений"	111
Организация MPICH	112
Привязка к языку C	117
Привязка к языку FORTRAN	118
Коды завершения	120
Как устроена MPI-программа	120
Программа написана, что дальше?	122
Трансляция	122
"Устройства" MPICH	124
Выполнение параллельной программы	124
Особенности выполнения программ на кластерах рабочих станций	125
Особенности выполнения программ MPICH на SMP-кластерах	128
Особенности выполнения программ MPI на гетерогенных системах	129
Отладка	130

Глава 4. Обмен данными в MPI..... 131

Двухточечный обмен сообщениями	131
Блокирующие операции обмена	134
Стандартный обмен	134
Синхронный блокирующий обмен	141
Буферизованный обмен	142
Обмен "по готовности"	144
Подпрограммы-пробники	144
Совместные прием и передача	148
Неблокирующие операции обмена	152
Инициализация неблокирующего обмена	153
Проверка выполнения обмена	155
Отложенные обмены	158
Отмена "ждущих" обменов	159

Коллективный обмен данными	164
Широковещательная рассылка.....	165
Обмен с синхронизацией	166
Управление областью взаимодействия и группой процессов.....	167
Группы процессов	168
Создание групп процессов	169
Получение информации о группе	171
Управление коммутаторами.....	172
Операции обмена между группами процессов (интеробмен).....	177
Глава 5. Коллективный обмен данными в MPI	181
Разновидности коллективного обмена.....	181
Широковещательная пересылка	182
Распределение и сбор данных.....	183
Операции приведения и сканирования	189
Топологии	195
Декартовы топологии.....	197
Топология графа.....	202
Производные типы данных	204
Конструкторы производных типов.....	206
Регистрация и удаление производных типов	211
Вспомогательные подпрограммы.....	211
Операции упаковки и распаковки данных	217
Типы <i>MPI_BYTE</i> и <i>MPI_PACKED</i>	220
Атрибуты	220
Ввод и вывод.....	225
Глава 6. Введение в параллельное программирование с использованием PVM.....	235
Общая характеристика PVM	235
Гетерогенные вычислительные системы и PVM.....	236
Программирование в PVM	238
Архитектура PVM	239
Как работает PVM	240
Демон PVM	241
Идентификатор задачи.....	241
Модель передачи сообщений в PVM	242
Асинхронные уведомления об особых событиях.....	243
Настройка PVM и трансляция PVM-программ	243
Структура каталога PVM	247
Консоль PVM	248
Возможные проблемы при работе с PVM	254
Конфигурационный файл PVM-консоли	255
Файл описания виртуальной машины	256

Наша первая PVM-программа	257
Управление процессами.....	259
Обмен сообщениями.....	261
Создание буфера.....	262
Упаковка данных.....	263
Подпрограммы передачи и приема данных.....	264
Распаковка данных.....	264
Коды ошибок.....	265
Сходство и различие PVM и MPI	266
Глава 7. Программирование с использованием PVM	275
Управление процессами	275
Подпрограммы для сбора информации о процессах.....	275
Изменение состояния процессов.....	279
Работа с буфером.....	281
Передача и прием сообщений	284
Управление виртуальной машиной.....	289
Сбор информации о состоянии виртуальной параллельной машины	289
Модификация виртуальной машины	293
Группы процессов	298
Управление группой.....	299
Групповые рассылки сообщений и синхронизация процессов.....	301
Глава 8. Высокопроизводительный FORTRAN	307
Общие сведения о HPF	307
Новое в FORTRAN-90.....	308
Программные единицы и структура программы.....	308
Лексические единицы и запись исходного текста	309
Главная программа и внешние подпрограммы	309
Внутренние подпрограммы.....	311
Интерфейсы.....	311
Модули и модульные процедуры	312
Рекурсивные процедуры и функции.....	313
Формальные параметры подпрограмм.....	313
Типы данных и операторы описания.....	314
Операторы описания	314
Базовые типы и их разновидности	314
Указатели	315
Производные типы	316
Массивы в FORTRAN-90	317
Строение массивов.....	318
Сечения массивов	318
Конструкторы массивов.....	319
Динамические массивы.....	319

Инструкция <i>WHERE</i>	320
Встроенные функции для работы с массивами.....	321
Директивы HPF.....	324
Инструкции и встроенные функции HPF.....	324
Распределение данных.....	325
Директива <i>DISTRIBUTE</i>	325
Распределение многомерных массивов.....	327
Распределение динамических массивов.....	328
Скалярные переменные.....	328
Выравнивание. Директива <i>ALIGN</i>	329
Процедуры и распределение данных.....	331
Параллелизм исполнения в HPF.....	333
Инструкция <i>FORALL</i>	333
<i>PURE</i> -процедуры.....	334
Директива <i>INDEPENDENT</i>	335

Приложение 1. Средства отладки и мониторинга параллельных MPI-программ 341

MPE — многопроцессорное окружение.....	341
Создание log-файлов.....	341
Форматы log-файлов.....	342
Параллельная графика.....	342
Другие MPE-программы.....	343
Библиотеки профилирования.....	343
Адаптированное протоколирование.....	344
Утилиты MPE.....	345
Переменные окружения.....	345
Присоединение библиотек.....	346
Автоматическая генерация библиотек.....	347
Описание определений оболочки.....	354
Графические инструменты.....	358

Приложение 2. Средства отладки и мониторинга параллельных PVM-программ 361

Окна просмотра.....	363
Окно просмотра конфигурации виртуальной машины.....	363
Окно просмотра пространственно-временной диаграммы.....	364
Окно просмотра степени использования системы.....	365
Окна просмотра данных трассировки и программного вывода.....	366

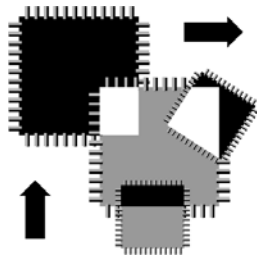
Приложение 3. Настройка Linux-кластера для параллельных приложений 369

Введение.....	369
Общие положения.....	369

Чем кластер отличается от сети	370
Общая установка	371
Уровень подготовки	371
Сетевое оборудование	372
Адреса	373
Входной узел	373
Конфигурация коммутатора	373
Прямой доступ	373
Решения вопросов безопасности	373
Образец файла hosts	374
Администрирование пользователей	375
Администрирование CMS	376
Компиляторы	376
Библиотеки передачи сообщений	376
Тесты производительности	378
Синхронизация часов	378

Приложение 4. Ресурсы Интернета, посвященные параллельному программированию	381
Информация о MPI	382
Информация о PVM	383
Информация о кластерах	384
Список литературы	385
Предметный указатель	387

ГЛАВА 1



Архитектура высокопроизводительных ЭВМ

Прежде чем мы перейдем к изучению методов и средств параллельного программирования, полезно познакомиться с некоторыми особенностями устройства высокопроизводительных вычислительных систем. Преимуществом использования языков программирования высокого уровня является универсальность программ и их простая переносимость между различными компьютерами (разумеется, если на этих компьютерах имеется необходимое программное обеспечение, прежде всего, трансляторы). Следует учитывать, что эффективность выполнения параллельных программ определяется не только аппаратной частью, но и способностью транслятора генерировать эффективный исполняемый код. Оба эти фактора взаимосвязаны и порой сложно определить, какой из них имеет решающее значение.

Обычно, программист полагается на эффективность транслятора, считая, что при грамотном программировании сгенерированный транслятором исполняемый код будет обладать хорошими показателями по быстродействию; в этом случае отпадает необходимость в применении языков низкого уровня (ассемблеров), требующих глубоких знаний архитектуры процессора.

Но, бывает и так, что программисту все же требуется знание устройства компьютера, принципов и даже некоторых деталей его работы. Можно привести следующий пример. При разработке программ для параллельных ЭВМ используются специализированные библиотеки, позволяющие организовать обмен данными между отдельными подзадачами и их синхронизацию. Для эффективной организации обмена данными необходимо знать, как происходит пересылка информации между процессорами, какова ее скорость и т. д.

В вычислительной технике используются три термина, связанные с устройством электронно-вычислительной машины:

1. *Архитектура компьютера* — это описание основных компонентов компьютера и их взаимодействия. Можно считать, что архитектура — это те атрибуты вычислительной системы, которые видны программисту: набор

команд, разрядность машинного слова, механизмы ввода/вывода, методы адресации. Можно дать и другое определение: архитектура — это внутренняя структура системы или микропроцессора, которая определяет их функциональные возможности и быстродействие.

2. *Организация компьютера* — это описание конкретной реализации архитектуры, ее воплощения "в железе". Оно включает перечень используемых сигналов управления, интерфейсов, технологий памяти, реализацию арифметических, логических и других операций. Так, умножение может быть реализовано на аппаратном уровне или посредством многократного повторения операции суммирования. Ряд суперкомпьютеров Cray, например, имеют сходную архитектуру. С точки зрения программиста у них одинаковое число внутренних регистров, используемых для временного хранения данных, одинаковый набор машинных команд, одинаковый формат представления данных. Организация же компьютеров Cray разных моделей может существенно различаться. У них может быть разное число процессоров, разный размер оперативной памяти, разное быстродействие и т. д.
3. *Схема компьютера* — детальное описание его электронных компонентов, их соединений, устройств питания, охлаждения и др. Программисту довольно часто требуется знание архитектуры компьютера, реже — его организации и никогда — схемы компьютера. Действительно, зачем специалисту, который занимается, например, расчетом прочностных свойств новой модели автомобиля, знать, какая марка вентилятора охлаждает процессор компьютера во время выполнения его программы?

Классификация Флинна

Одной из наиболее известных схем классификации компьютерных архитектур является *таксономия Флинна*, предложенная Майклом Флинном в 1972 году. В ее основу положено описание работы компьютера с потоками команд и данных. В классификации Флинна имеется четыре класса архитектур:

1. **SISD** (Single Instruction Stream — Single Data Stream) — один поток команд и один поток данных.
2. **SIMD** (Single Instruction Stream — Multiple Data Stream) — один поток команд и несколько потоков данных.
3. **MISD** (Multiple Instruction Stream — Single Data Stream) — несколько потоков команд и один поток данных.
4. **MIMD** (Multiple Instruction Stream — Multiple Data Stream) — несколько потоков команд и несколько потоков данных.

Рассмотрим эту классификацию более подробно.

SISD-компьютеры

SISD-компьютеры (рис. 1.1) — это обычные последовательные компьютеры, выполняющие в каждый момент времени только одну операцию над одним элементом данных. Большинство современных персональных ЭВМ принадлежат именно этой категории.

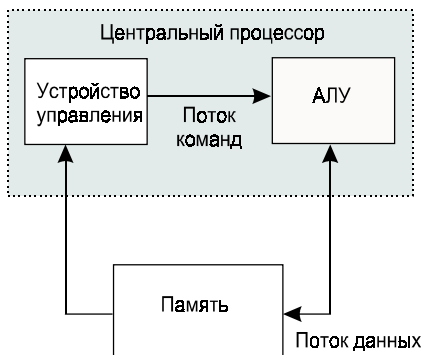


Рис. 1.1. Схема SISD-компьютера

Многие современные вычислительные системы включают в свой состав несколько процессоров, но каждый из них работает со своим независимым потоком данных, относящимся к независимой программе. Такой компьютер является, фактически, набором SISD-машин, работающих с независимыми множествами данных.

SIMD-компьютеры

SIMD-компьютеры (рис. 1.2 и 1.3) состоят из одного командного процессора (управляющего модуля), называемого *контроллером*, и нескольких модулей обработки данных, называемых *процессорными элементами* (ПЭ). Количество модулей обработки данных таких машин может быть от 1024 до 16 384. Управляющий модуль принимает, анализирует и выполняет команды. Если в команде встречаются данные, контроллер рассылает на все ПЭ команду, и эта команда выполняется либо на нескольких, либо на всех процессорных элементах. Процессорные элементы в SIMD-компьютерах имеют относительно простое устройство, они содержат арифметико-логическое устройство (АЛУ), выполняющее команды, поступающие из устройства управления (УУ), несколько регистров и локальную оперативную память. Одним из преимуществ данной архитектуры считается эффективная реализация логики вычислений. До половины логических команд обычного процессора связано с управлением процессом выполнения машинных команд, а остальная их часть относится к работе с внутренней памятью процессора

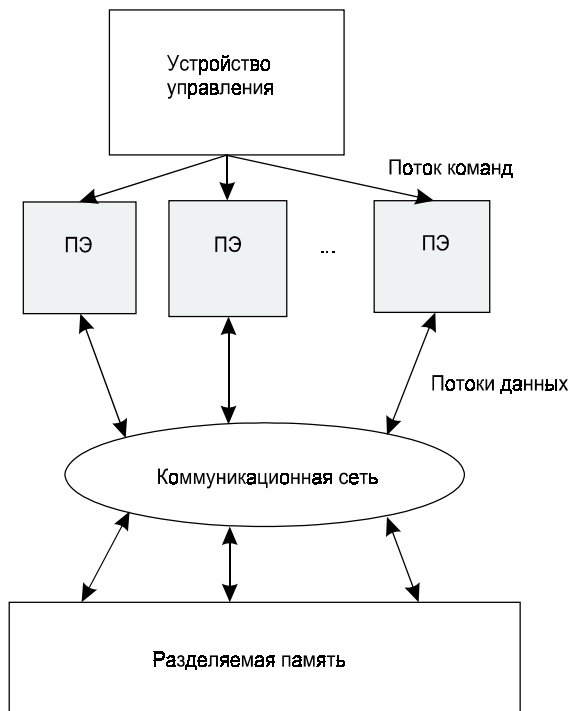


Рис. 1.2. Схема SIMD-компьютера с разделяемой памятью

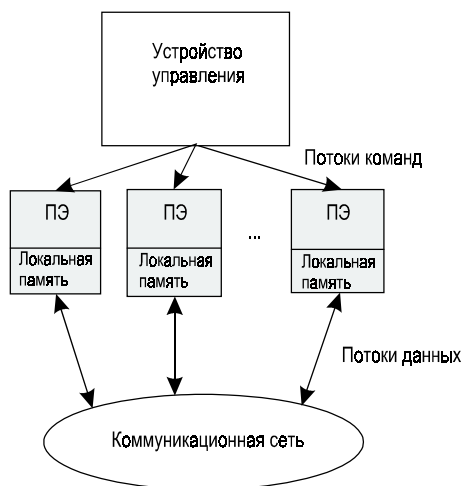


Рис. 1.3. Схема SIMD-компьютера с распределенной памятью

и выполнению арифметических операций. В SIMD-компьютере управление выполняется контроллером, а "арифметика" отдана процессорным элементам. Подклассом SIMD-компьютеров являются векторные компьютеры. Пример такой вычислительной системы — Hitachi S3600.

Другой пример SIMD-компьютера — *матричные процессоры* (Array Processor). В качестве примера можно привести вычислительную систему Thinking Machines CM-2, где 65 536 ПЭ связаны между собой сетью коммуникаций с топологией "гиперкуб". Часто компьютеры с SIMD-архитектурой специализированы для решения конкретных задач, допускающих матричное представление. Это, например, могут быть задачи обработки изображений, где каждый модуль обработки данных работает на получение одного элемента конечного результата.

MISD-компьютеры

Вычислительных машин такого класса мало. Один из немногих примеров — *систолический массив* процессоров, в котором процессоры находятся в узлах регулярной решетки. Роль ребер в ней играют межпроцессорные соединения, все ПЭ управляются общим тактовым генератором. В каждом цикле работы любой ПЭ получает данные от своих соседей, выполняет одну команду и передает результат соседям. На рис. 1.4 дана схема фрагмента систолического массива.

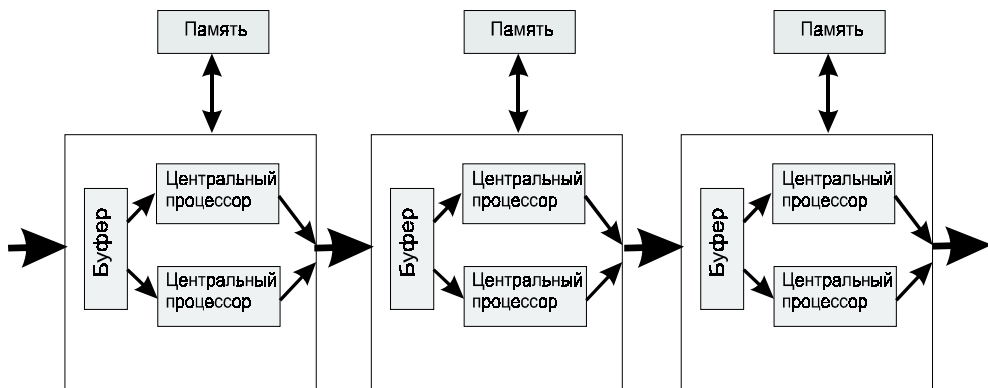


Рис. 1.4. Схема MISD-компьютера

MIMD-компьютеры

Этот класс архитектур (рис. 1.5 и 1.6) наиболее богат примерами успешных реализаций. В него попадают симметричные параллельные вычислительные системы, рабочие станции с несколькими процессорами, кластеры рабочих

станций и т. д. Довольно давно появились компьютеры с несколькими независимыми процессорами, но вначале на них был реализован только принцип параллельного исполнения заданий, т. е. на разных процессорах одновременно выполнялись независимые программы. Разработке первых компьютеров для параллельных вычислений были посвящены проекты под условным названием CM^* и $S.MMP$ в университете Карнеги (США). Технической базой для этих проектов были процессоры DEC PDP-11. В начале 90-х годов прошлого века именно MIMD-компьютеры вышли в лидеры на рынке высокопроизводительных вычислительных систем.

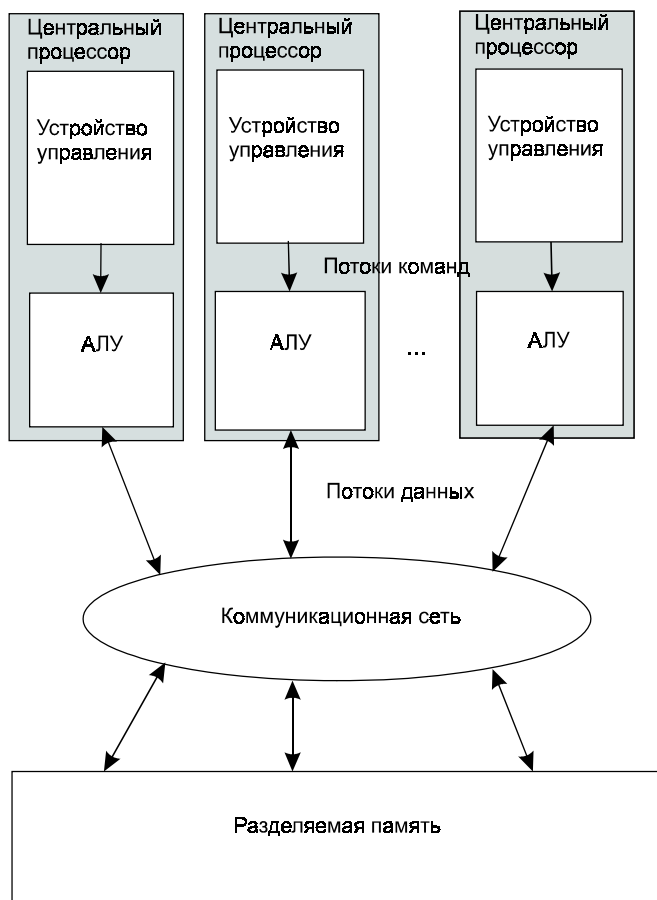


Рис. 1.5. Схема MIMD-компьютера с разделяемой памятью

Имеются и гибридные конфигурации, в которых, например, объединены несколько SIMD-компьютеров, в результате чего получается MSIMD-ком-

пьютер, позволяющий создавать виртуальные конфигурации, каждая из которых работает в SIMD-режиме.

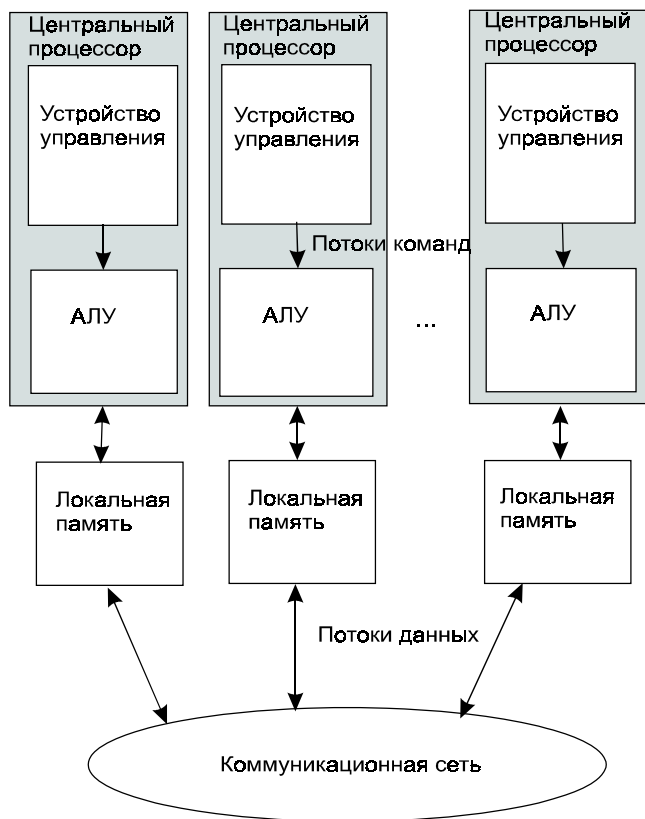


Рис. 1.6. Схема MIMD-компьютера с распределенной памятью

Классификация Флинна не дает исчерпывающего описания разнообразных архитектур MIMD-машин, порой существенно отличающихся друг от друга. Например, существуют такие подклассы MIMD-компьютеров, как системы с разделяемой памятью и системы с распределенной памятью. Системы с разделяемой памятью могут относиться по классификации Флинна как к MIMD, так и к SIMD-машинам. То же самое можно сказать и о системах с распределенной памятью.

Развитием концепции MIMD-архитектуры с распределенной памятью является *распределенная обработка*, когда вместо набора процессоров в одном корпусе используются компьютеры, связанные достаточно быстрой сетью. Концептуального отличия от MIMD-архитектуры с распределенной памятью нет, а особенностью является медленное сетевое соединение.

Традиционная архитектура фон Неймана

Традиционная фон-неймановская архитектура компьютера представлена на рис. 1.7. Она основана на следующих принципах:

- ❑ программа хранится в компьютере;
- ❑ программа во время выполнения и необходимые для ее работы данные находятся в оперативной (главной) памяти;
- ❑ имеется *арифметико-логическое устройство*, выполняющее арифметические и логические операции с данными;
- ❑ имеется *устройство управления*, которое интерпретирует команды, выбираемые из памяти, и выполняет их;
- ❑ *устройства ввода и вывода (ВВ)* используются для ввода программ и данных и для вывода результатов расчетов. Работают под управлением УУ.

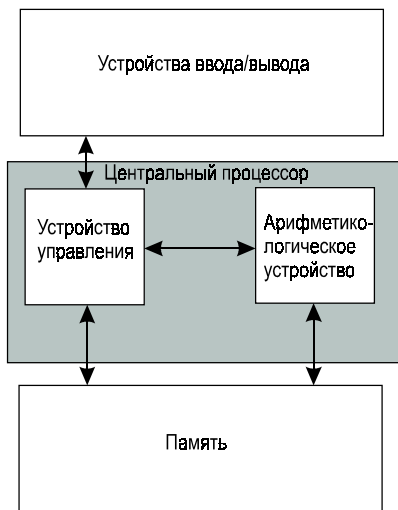


Рис. 1.7. Схема компьютера фон Неймана

Устройство управления и арифметико-логическое устройство (АЛУ) вместе составляют *центральный процессор* (ЦП). Можно считать, что все остальные функциональные блоки компьютера обслуживают АЛУ, а оно выполняет основную работу. Арифметические операции выполняются с целыми и вещественными числами, для представления которых используется двоичный формат. Для отрицательных значений применяется дополнительный код, что позволяет упростить реализацию арифметических и логических операций. Управляющее же устройство управляет работой остальных функциональных узлов компьютера.

В состав "традиционного" фон-неймановского компьютера входит один ЦП, основной задачей которого является выполнение машинных команд, выбираемых из оперативной памяти по очереди, одна за другой. Устройство управления интерпретирует очередную команду, которая входит в набор команд процессора, включающих арифметические и логические операции, операции пересылки данных и некоторые другие. Затем выполняется выборка данных, их обработка и запись результата выполнения в оперативную память. ЦП имеет набор регистров — устройств для временного хранения промежуточных результатов и данных, необходимых для выполнения команд (операндов).

Таким образом, основными компонентами компьютера являются:

- ☐ центральный процессор и оперативная память, образующие ядро системы;
- ☐ вторичная ("внешняя") память и устройства ввода/вывода, образующие "периферию";
- ☐ коммуникации между компонентами системы, осуществляемые посредством шин.

Регистры

Регистры центрального процессора представляют собой вспомогательную память, находящуюся на вершине иерархической организации памяти (об иерархии памяти мы поговорим чуть позже). Количество регистров различается в процессорах разного типа. Программисту доступны:

- ☐ регистры общего назначения (General Purpose Registers);
- ☐ регистры данных (Data Registers);
- ☐ регистры адреса (Address Registers);
- ☐ регистр кодов условий (Condition Codes Register).

Имеются также управляющие регистры и регистры состояния:

- ☐ счетчик команд (Program Counter);
- ☐ регистр декодирования команд (Instruction Register);
- ☐ регистр адресации памяти (Memory Addressing Register);
- ☐ регистр буфера памяти (Memory Buffer Register).

Выполнение команды

Выполнение команды состоит из двух этапов:

- ☐ выборка;
- ☐ выполнение.

Первый шаг — выборка — выполняется следующим образом. Вначале из счетчика команд выбирается адрес очередной команды, при этом значение, содержащееся в счетчике команд, увеличивается. Адрес команды помещается в регистр адресации памяти и передается в адресную шину. УУ считывает данные из оперативной памяти. Результат считывания помещается в шину данных, копируется в регистр буфера памяти, а затем в регистр декодирования команд.

При выборке данных анализируется регистр декодирования команд и производится выборка данных. Младшие биты регистра буфера памяти передаются в регистр адресации памяти, затем УУ формирует запрос к памяти и результат выполнения этого запроса (адрес или операнд) пересылается в регистр буфера памяти.

Второй шаг — выполнение. Во время этого цикла происходит пересылка данных между центральным процессором и оперативной памятью и между центральным процессором и модулем ввода/вывода; выполняются арифметические и логические операции над данными. При этом, если есть переходы, может изменяться последовательность выполнения команд.

Набор команд процессора

Центральный процессор выполняет машинные команды из набора, определенного для данной разновидности процессоров. Для представления этих команд используется двоичный формат. Каждая машинная команда включает код операции, ссылку на операнды, адрес записи результата операции и ссылку на следующую по порядку команду.

Набор команд процессора характеризуется:

- ☐ типами операндов;
- ☐ форматом команд;
- ☐ допустимыми командами;
- ☐ количеством адресов в команде;
- ☐ доступом к регистрам;
- ☐ режимами адресации.

Большинство машинных команд можно разделить на четыре категории:

- ☐ арифметические и логические операции;
- ☐ операции пересылки данных между оперативной памятью и регистрами центрального процессора;
- ☐ операции ввода/вывода;
- ☐ команды управления (проверки, ветвления).