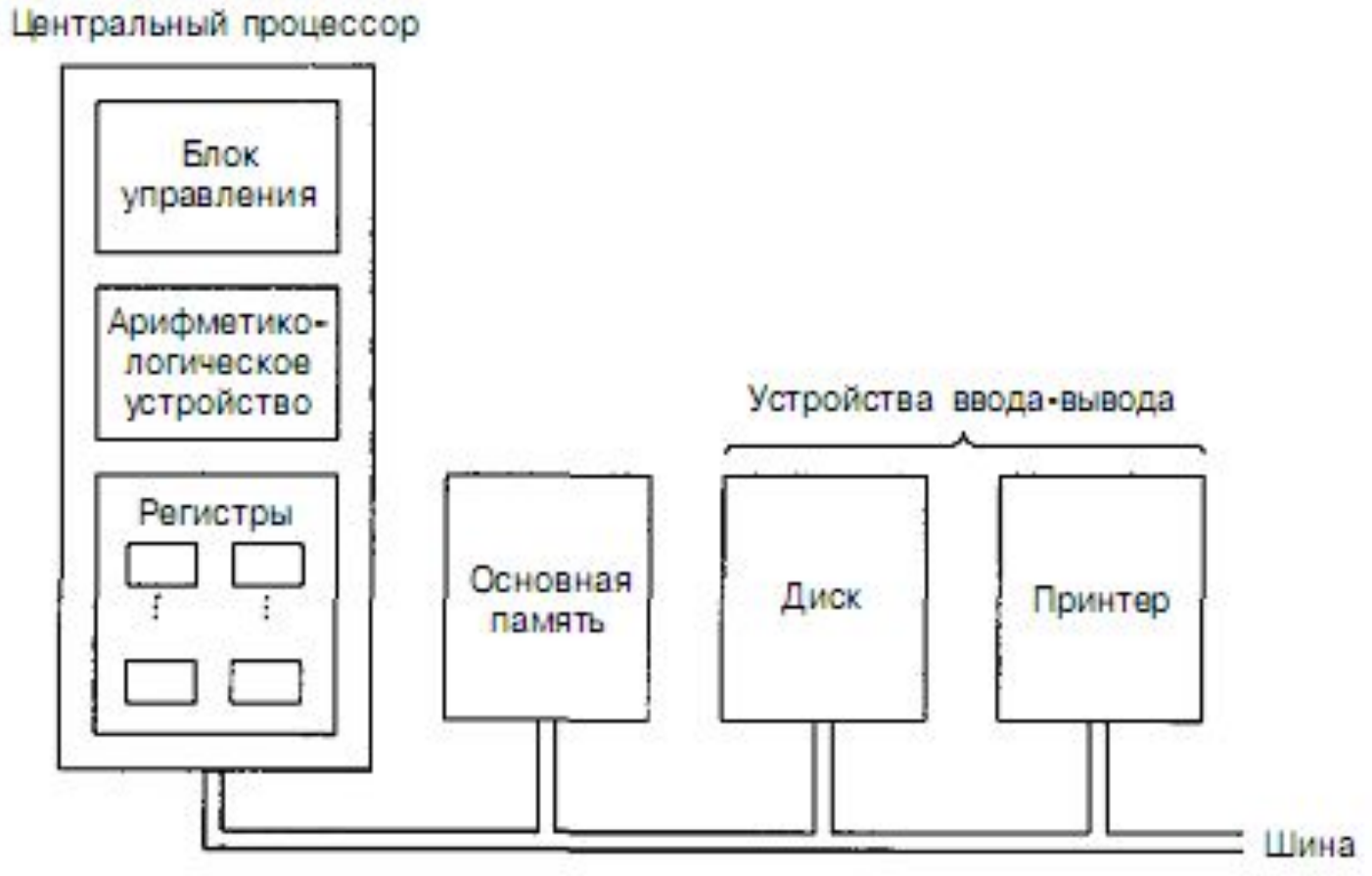
Контрольные вопросы

1. Сформулируйте закон Мура.
2. Дайте определения компьютерной системы, компьютерной сети и информации.
3. Сущность многоуровневого подхода к организации компьютерных систем.
4. Трансляция и интерпретация.
5. Сравнительная характеристика уровней компьютерных систем.
6. Операционная система и ее функции.
7. Сравнительная характеристика компьютерных систем различных поколений.

Процессор

Цифровой компьютер состоит из связанных между собой процессоров, памяти и устройств ввода-вывода.

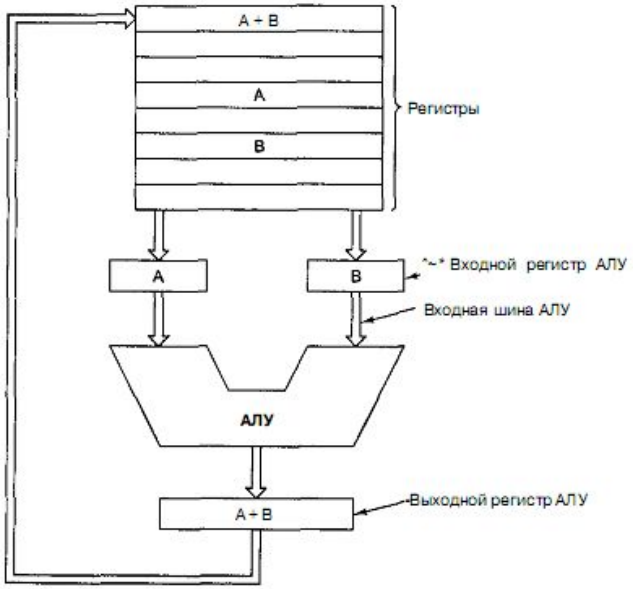
Процессор



Устройство центрального процессора:

• Блок управления отвечает за вызов команд из памяти и определение их типа.
• Арифметико-логическое устройство выполняет арифметические операции (например, сложение) и логические операции (например, логическое И).
• Память для хранения промежуточных результатов и некоторых команд управления. Состоит из нескольких регистров, каждый из которых выполняет определенную функцию. Самый важный регистр - **счетчик команд**, который указывает, какую команду нужно выполнять дальше. В **регистре команд** находится команда, выполняемая в данный момент.

Арифметико-логическое устройство, регистры и соединяющие их шины (набор параллельно связанных проводов) образуют **тракт данных**.



Содержимое регистров поступает во входные регистры арифметико-логического устройства. Арифметико-логическое устройство выполняет сложение, вычитание и другие простые операции над входными данными и помещает результат в выходной регистр.

Выполнение команд

1) вызывает команду из памяти и переносит ее в регистр команд;
2) меняет положение счетчика команд, который теперь должен указывать на следующую команду
3) определяет тип вызванной команды;
4) если команда использует слово из памяти, определяет, где находится это слово;
5) переносит слово, если это необходимо, в регистр центрального процессора;
6) выполняет команду;
7) переходит к шагу 1.

«Слова» - это элементы данных, которые перемещаются между памятью и регистрами. Слова – это числа.

Описание работы центрального процессора можно представить в виде программы. Т.е. программа не обязательно должна выполняться реальным процессором, относящимся к аппаратному обеспечению. Вызывать из памяти, определять тип команд и выполнять эти команды может другая программа - **интерпретатор**.

При наличии в языке программирования сложных команд программы, написанные на этом языке, выполняются быстрее. При условии, что мы имеем процессор, способный выполнять сложные программы, как правило дорогостоящий.

Пример сложной команды
выполнение операций с плавающей запятой – 1,23 или 0,123E+1 или 0,0123E+2 или 12,3E-1 или 123E-2 и т.д. Фиксированная запятая – 1,23.

Как построить дешевый компьютер, который будет выполнять все сложные команды, предназначенные для высокоэффективных дорогостоящих машин?

Решением этой проблемы стала интерпретация. Эта технология, впервые предложенная Уилксом в 1951 году, позволяла разрабатывать простые дешевые компьютеры, которые, могли выполнять сложные команды.

В результате в 1964 году IBM создала архитектуру System/360 - семейство совместимых компьютеров, различных по цене и производительности.

Затраты на разработку System/360 составили около 5 млрд. долларов США (что соответствует 30 млрд. в современных ценах). Это был второй по стоимости проект 1960-х годов после программы «Аполлон».

Во всех моделях компьютеров серии System/360 использовались одинаковые сложные команды. В дорогостоящих машинах команды выполнялись непосредственно аппаратным обеспечением. В дешевых компьютерах применялась интерпретация.

К концу 70-х годов интерпретаторы стали применяться практически во всех моделях, кроме самых дорогостоящих машин.

Главное преимущество интерпретации заключалось в том, что можно было разработать простой процессор, а вся сложность сводилась к созданию интерпретатора. Таким образом, разработка сложного аппаратного обеспечения замещалась разработкой сложного программного обеспечения.

Все преимущества использования интерпретаторов при разработке новых машин продемонстрировал успех Motorola 68000 с большим набором интерпретируемых команд и одновременный провал Zilog Z8000, у которого был столь же обширный набор команд, но не было интерпретатора. При том, что Z80 (предшественник Zilog Z8000) пользовался большей популярностью, чем Motorola 6800 (предшественник Motorola 68000).
Еще один фактор в пользу интерпретации - создание быстрых постоянных запоминающих устройств для хранения интерпретаторов.

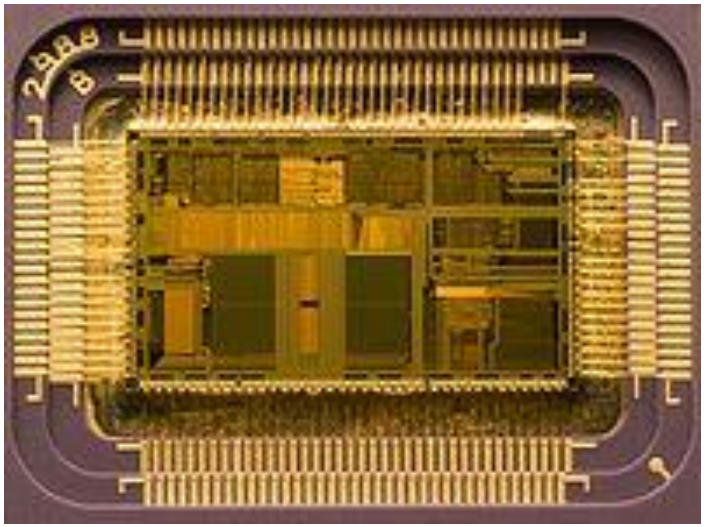
RISC и CISC

В 80-ых годах с развитием электронных технологий были предприняты попытки создания процессоров без использования интерпретации: Reduced Instruction Set Computer (RISC) - компьютер с сокращенным набором команд.

С этого момента началась идеологическая война между сторонниками RISC и разработчиками CISC (Complex Instruction Set Computer - компьютер с полным набором команд).

Что быстрее: выполнить несколько простых команд или одну сложную команду, эквивалентную нескольким простым?

Победил «гибридный» подход: процессоры Intel, начиная с 486-го, содержат ядро RISC, которое выполняет самые простые (и обычно самые распространенные) команды за один цикл тракта данных, а по обычной технологии CISC интерпретируются более сложные команды.



Выпускался 1989 по 2007 годы.

Принципы разработки современных компьютеров

Требования совместимости с другими машинами

1. ***Все команды непосредственно выполняются аппаратным обеспечением***. Они не интерпретируются микрокомандами. Устранение уровня интерпретации обеспечивает высокую скорость выполнения большинства команд.
2. ***Компьютер должен начинать выполнение большого числа команд***. Параллелизм может играть главную роль в улучшении производительности.

3. ***Команды должны легко декодироваться***. Декодирование команд осуществляется для того, чтобы определить, какие ресурсы им необходимы и какие действия нужно выполнить.
4. ***Должно быть большое количество регистров.*** Доступ к памяти происходит медленно, поэтому в компьютере должно быть много регистров. Если команда однажды вызвана из памяти, при наличии большого числа регистров она может содержаться в регистре до тех пор, пока будет не нужна. Возвращение ее из регистра в память и новая загрузка в регистр нежелательны. Лучший способ избежать излишних перемещений - наличие достаточного количества регистров.

Параллелизм на уровне команд

Один из способов заставить процессоры работать быстрее - увеличение их скорости. Для достижения лучшей производительности при данной скорости работы процессора используют параллелизм (возможность выполнять две или более операций одновременно).

Две формы параллелизма: параллелизм на уровне команд (обеспечивает выполнение большого количества команд в секунду) и параллелизм на уровне процессоров (над одной задачей работают одновременно несколько процессоров).

Конвейеры

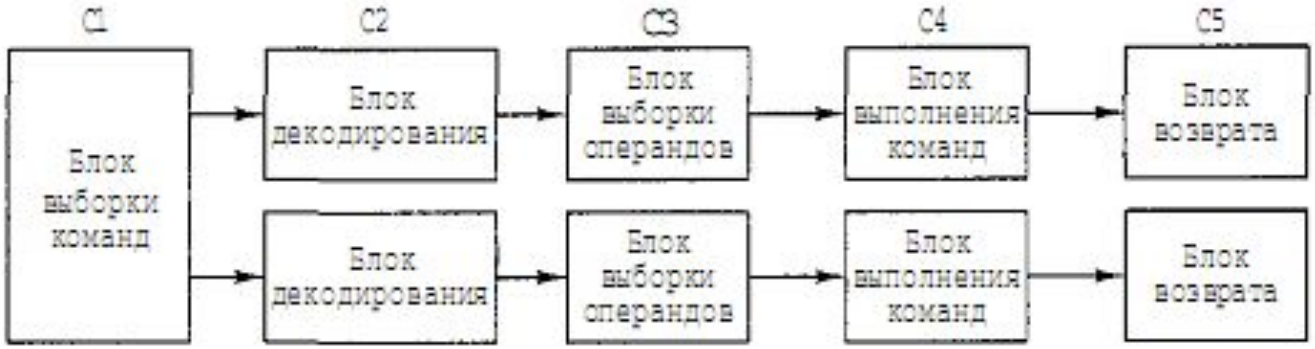
Главным препятствием высокой скорости выполнения команд является их вызов из памяти. Для разрешения этой проблемы придумал вызывать команды из памяти заранее, чтобы они имелись в наличии в тот момент, когда будут необходимы. Эти команды помещались в набор регистров, который назывался **буфером выборки с упреждением**.

IBM Stretch, 1959 год

C1 вызывает команду из памяти и помещает ее в буфер, где она хранится до тех пор, пока не будет нужна.
C2 декодирует эту команду, определяя ее тип и тип операндов, над которыми она будет производить определенные действия.
C3 определяет местонахождение операндов и вызывает их или из регистров или из памяти.
C4 выполняет команду, обычно путем провода операндов через тракт данных.
C5 записывает результат обратно в нужный регистр.

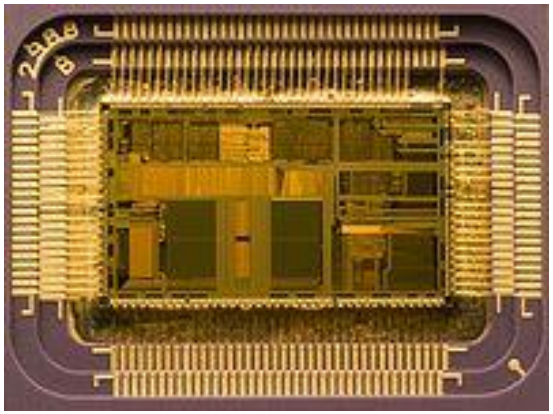
Во время цикла 1 стадия С1 работает над командой 1, вызывая ее из памяти. Во время цикла 2 стадия С2 декодирует команду 1, в то время как стадия С1 вызывает из памяти команду 2. Во время цикла 3 стадия С3 вызывает операнды для команды 1, стадия С2 декодирует команду 2, а стадия С1 вызывает третью команду. Во время цикла 4 стадия С4 выполняет команду 1, С3 вызывает операнды для команды 2, С2 декодирует команду 3, а С1 вызывает команду 4 и т.д.

Суперскалярные архитектуры (процессоры с двойным конвейером)



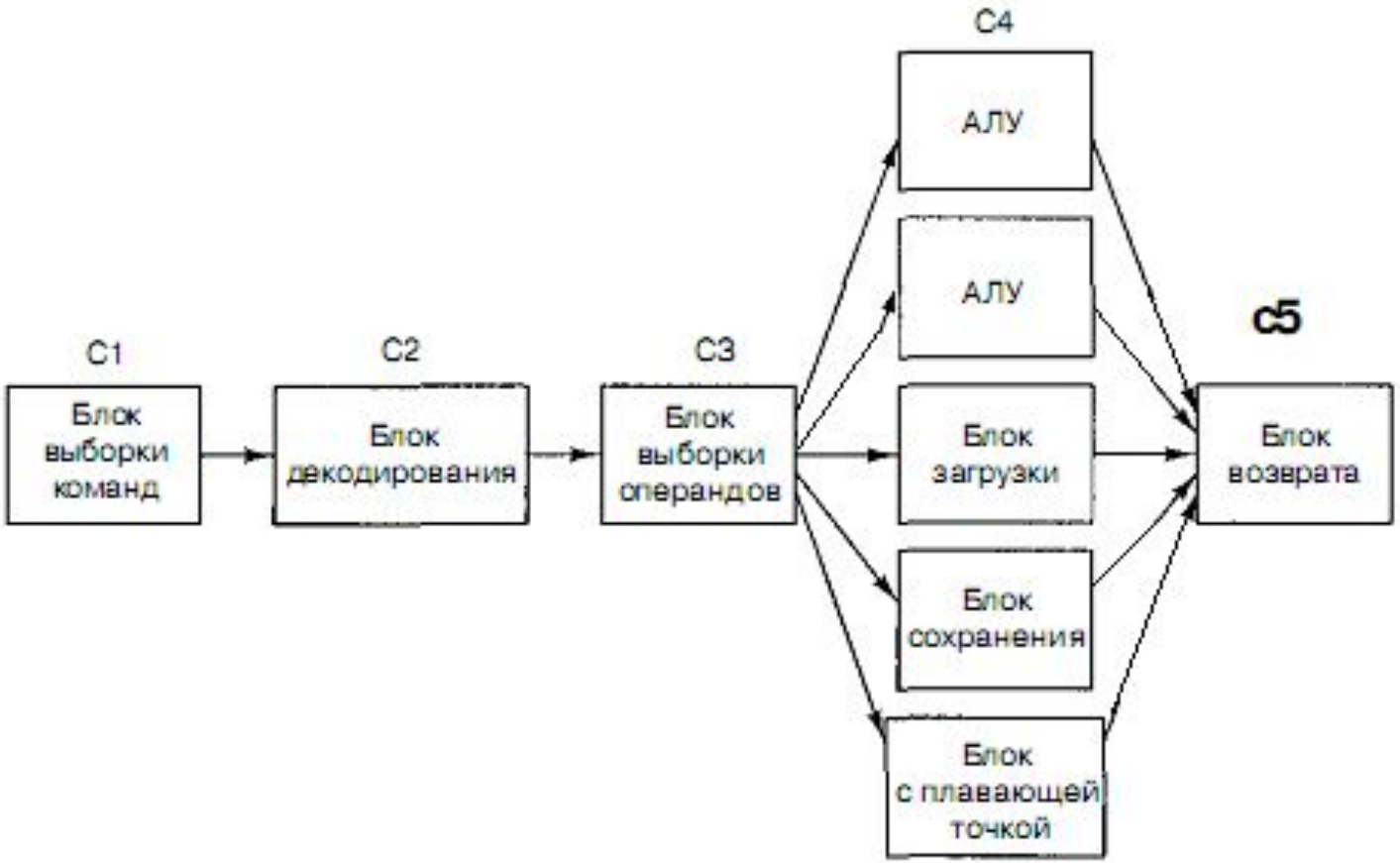
Общий отдел вызова команд берет из памяти сразу по две команды и помещает каждую из них в один из конвейеров. Чтобы могли выполняться параллельно две команды, они не должны конфликтовать при использовании ресурсов (например, регистров), и ни одна из них не должна зависеть от результата выполнения другой.

Первые конвейеры появились в процессорах компании Intel только начиная с 486-й модели (1 конвейер).



В процессорах пятого поколения Pentium появилось два конвейера из пяти стадий.

Переход к трем, четырем и т.д. конвейерам возможен, но это потребовало бы создания громоздкого аппаратного обеспечения. Вместо этого используется другой подход. Основная идея - один конвейер с большим количеством функциональных блоков.



В 1987 году для обозначения этого подхода был введен термин **суперскалярная архитектура**. Впервые подобная идея нашла воплощение в 1964 году в компьютере CDC 6600.

CDC 6600

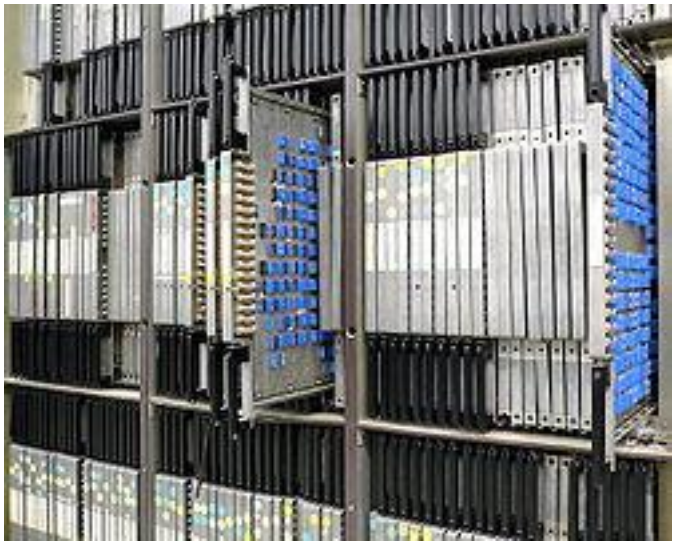
Pentium II

Параллелизм на уровне процессоров

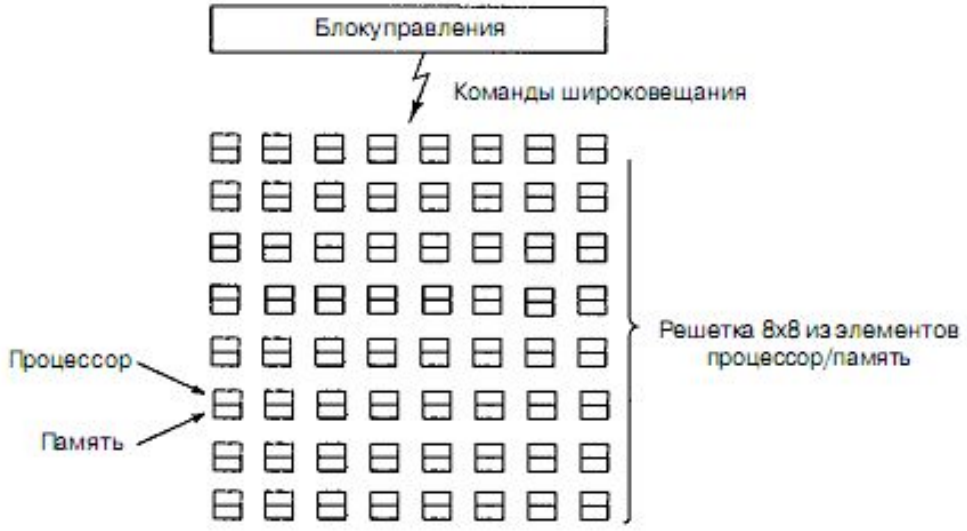
Параллелизм на уровне команд помогает в какой-то степени, но конвейеры и суперскалярная архитектура обычно увеличивают скорость работы всего лишь в 5-10 раз. Чтобы улучшить производительность в 50, 100 и более раз, нужно разрабатывать компьютеры с несколькими процессорами.

Массивно-параллельный процессор (array processor)

состоит из большого числа сходных процессоров, которые выполняют одну и ту же последовательность команд применительно к разным наборам данных.



Первым в мире таким процессором был ILLIAC IV, 1976 г.

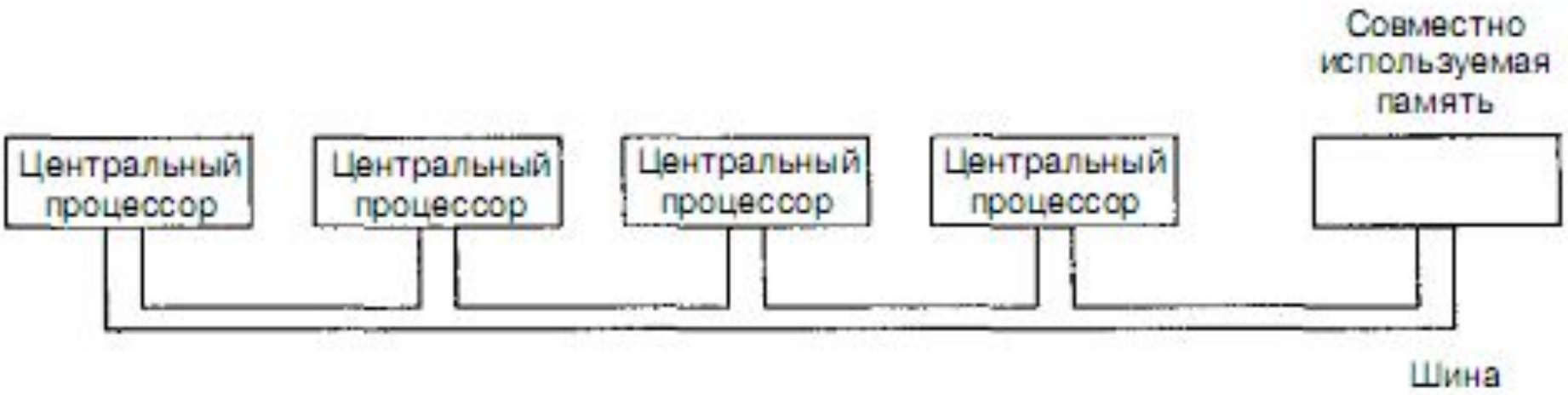


Машина состояла из четырех секторов, каждый из которых содержал решетку 8x8 элементов процессор/память. Для каждого сектора имелся один блок контроля. Он рассылал команды, которые выполнялись всеми процессорами одновременно, при этом каждый процессор использовал свои собственные данные из своей собственной памяти. Мощность восьми таких секторов равнялась мощности компьютеров всего мира.

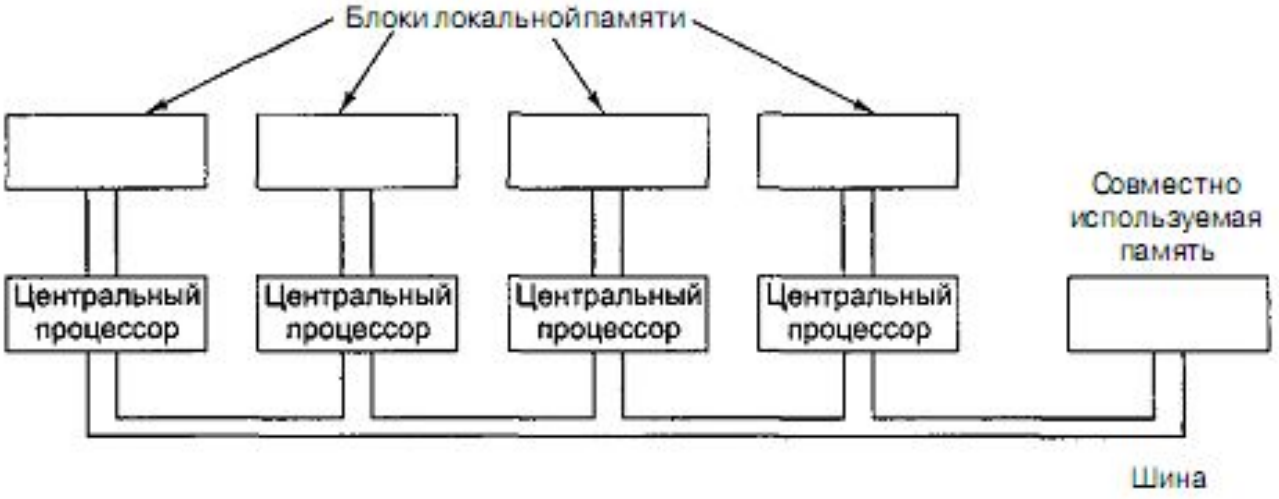
Cray-1, 1974 г.

Мультипроцессоры

Элементы массивно-параллельного процессора связаны между собой, поскольку их работу контролирует один блок управления. Система нескольких параллельных процессоров, разделяющих общую память, называется **мультипроцессором**.



Наличие одной шипы, соединяющей несколько процессоров и одну общую память. При наличии большого числа быстро работающих процессоров, которые постоянно пытаются получить доступ к памяти через одну и ту же шину, будут возникать конфликты. Чтобы разрешить эту проблему и повысить производительность компьютера, была разработана модель, в которой каждый процессор имеет свою собственную локальную память, которая недоступна для других процессоров.



Эта память используется для программ и данных, которые не нужно разделять между несколькими процессорами. При доступе к локальной памяти главная шина не используется, и, таким образом, поток информации в этой шине снижается.

Мультикомпьютеры

Мультипроцессоры с небольшим числом процессоров (~ 50) сконструировать легко. Создание больших мультипроцессоров представляет трудности: связать все процессоры с памятью. Поэтому стали создавать системы, состоящие из большого числа взаимосвязанных компьютеров, у каждого из которых имеется своя собственная память, а общей памяти нет. Такие системы называются **мультикомпьютерами**. Сейчас создаются мультикомпьютеры, содержащие ~ 10 000 процессоров.

Контрольные вопросы

1. Структура центрального процессора.
2. Тракт данных. Счетчик команд и регистр команд.
3. Последовательность выполнения команд центральным процессором.
4. Сущность интерпретации.
5. Процессоры RISC и CISC.
6. Принципы разработки современных компьютеров.
7. Параллелизм на уровне команд и на уровне процессоров.
8. Конвейеры. Суперскалярная архитектура.
9. Массивно-параллельный процессор.
10. Мультипроцессоры и мультикомпьютеры.

Спасибо за внимание!