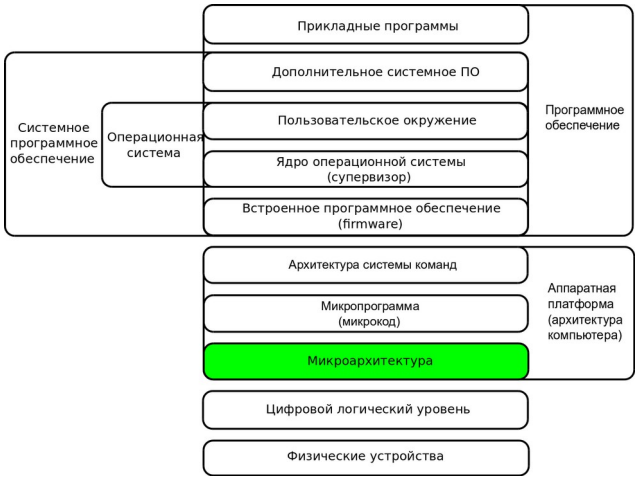
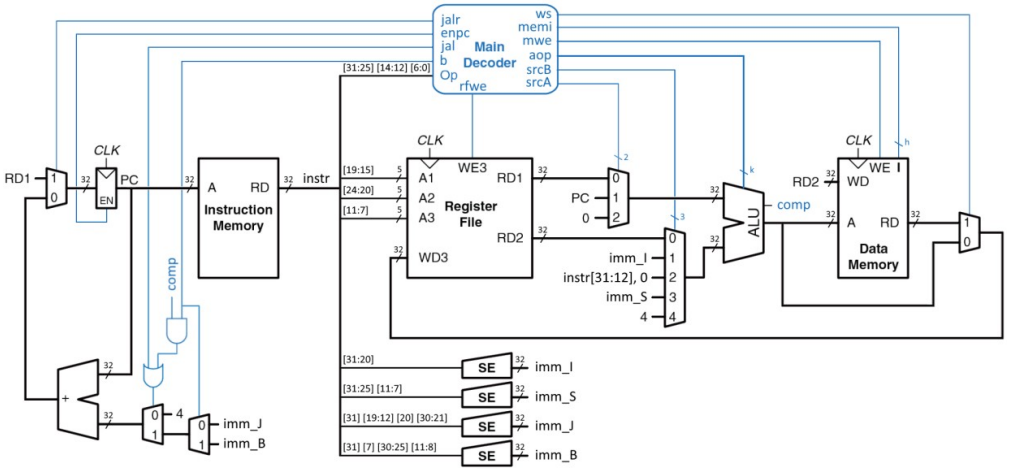
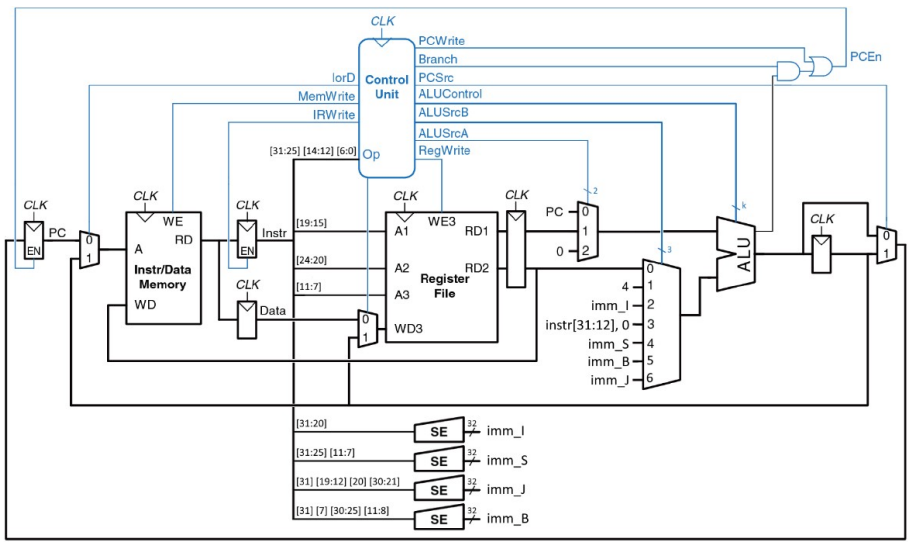
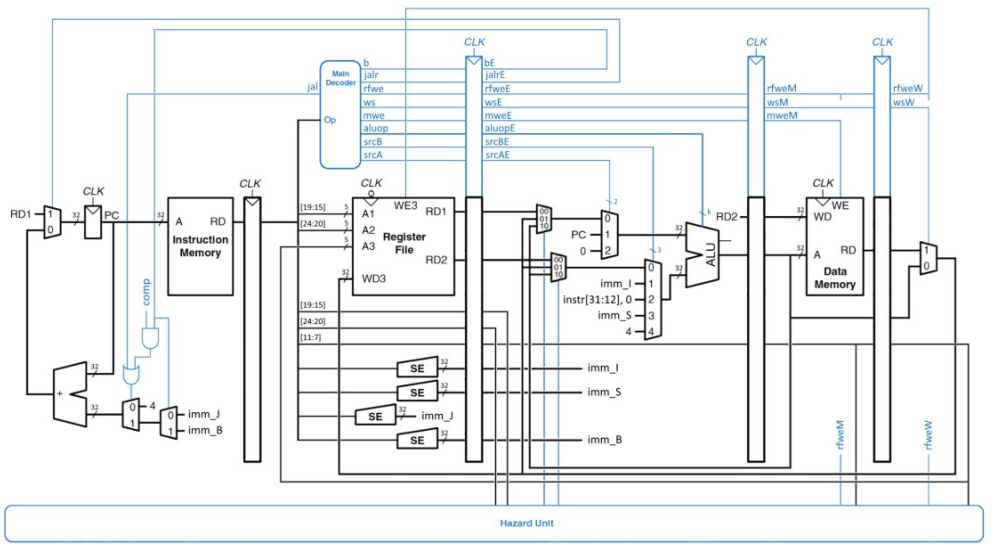
Микроархитектура









Определения

Микроархитектура (microarchitecture, μarch, uarch, организация компьютера) — способ, которым данная архитектура набора команд (ISA, АНК) реализована в процессоре

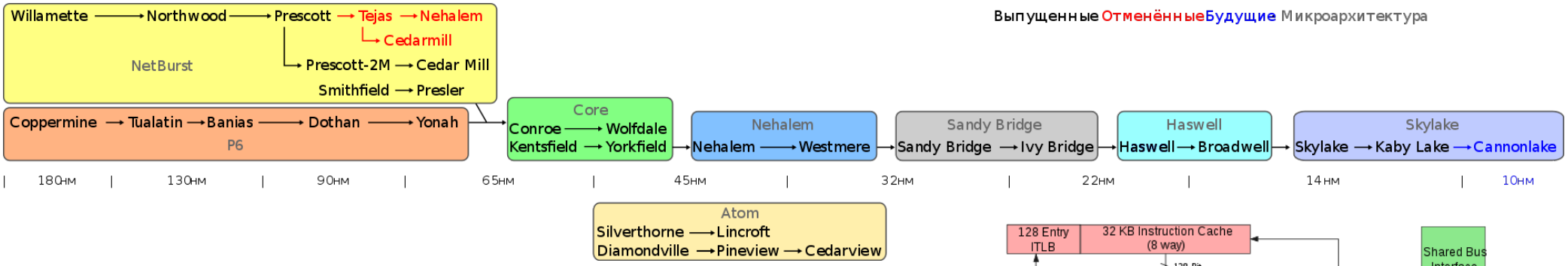
Интерпретация, выполнение набора команд

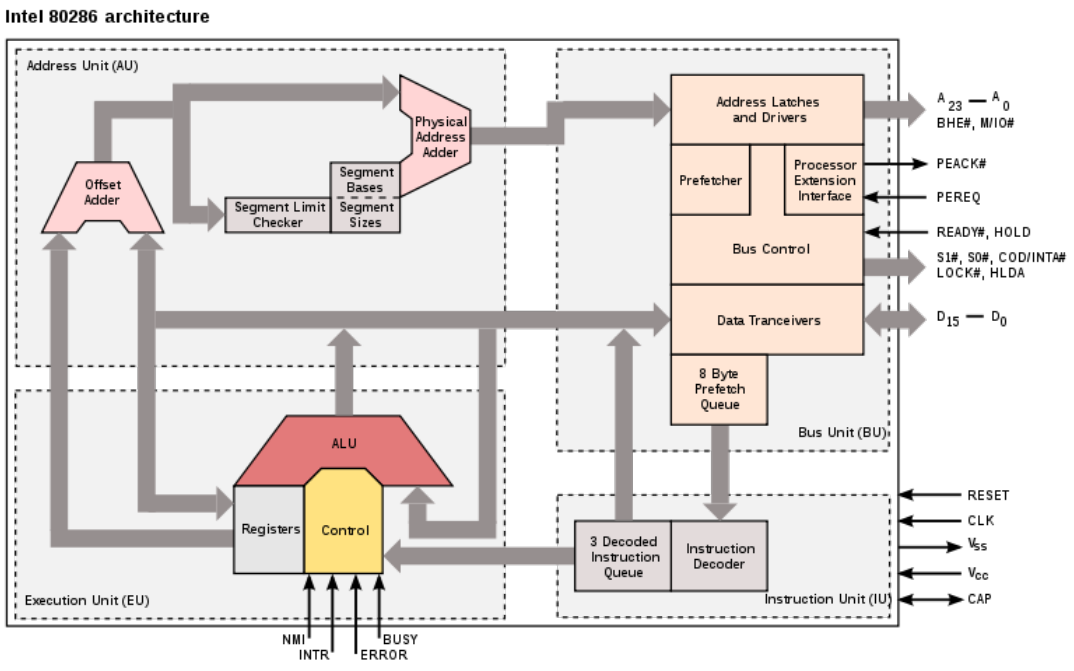
Каждая АНК может быть реализована с помощью различных микроархитектур

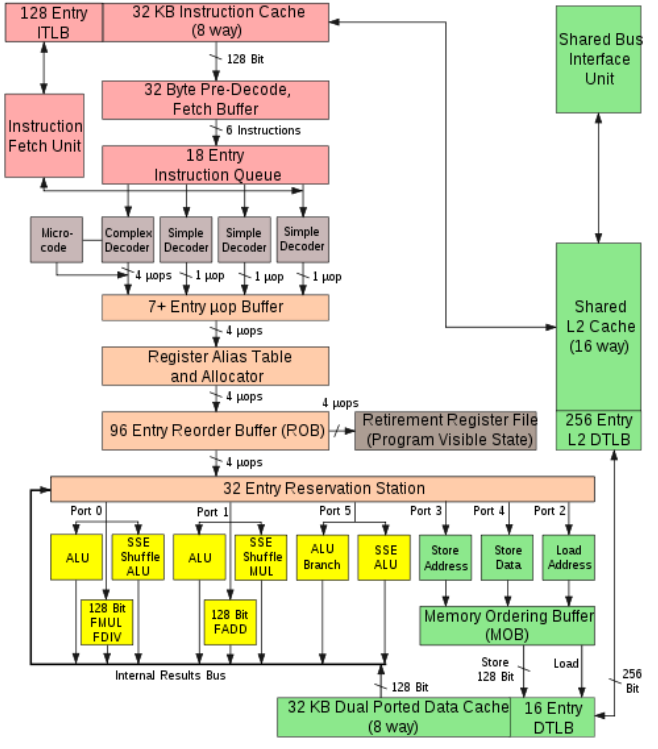
Различные системы команд могут быть реализованы на базе одной микроархитектуры.

Новые микроархитектуры и/или схемотехнические решения вместе с прогрессом в полупроводниковой промышленности позволяют новым поколениям процессоров достигать более высокой производительности, используя ту же АНК.

Поколения процессоров Intel

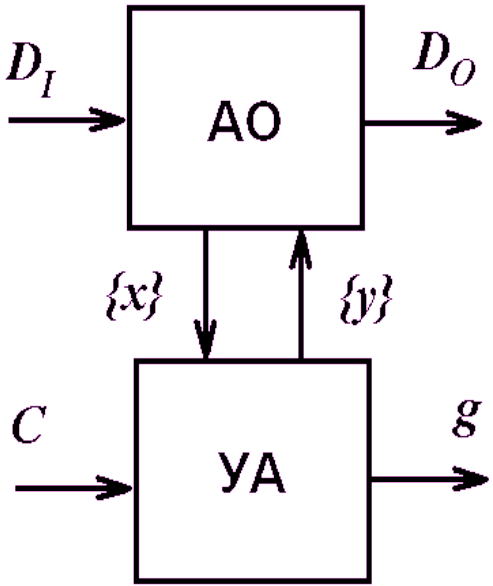






Intel Core 2 Architecture

Процессор как комбинация операционного и управляющего автоматов*



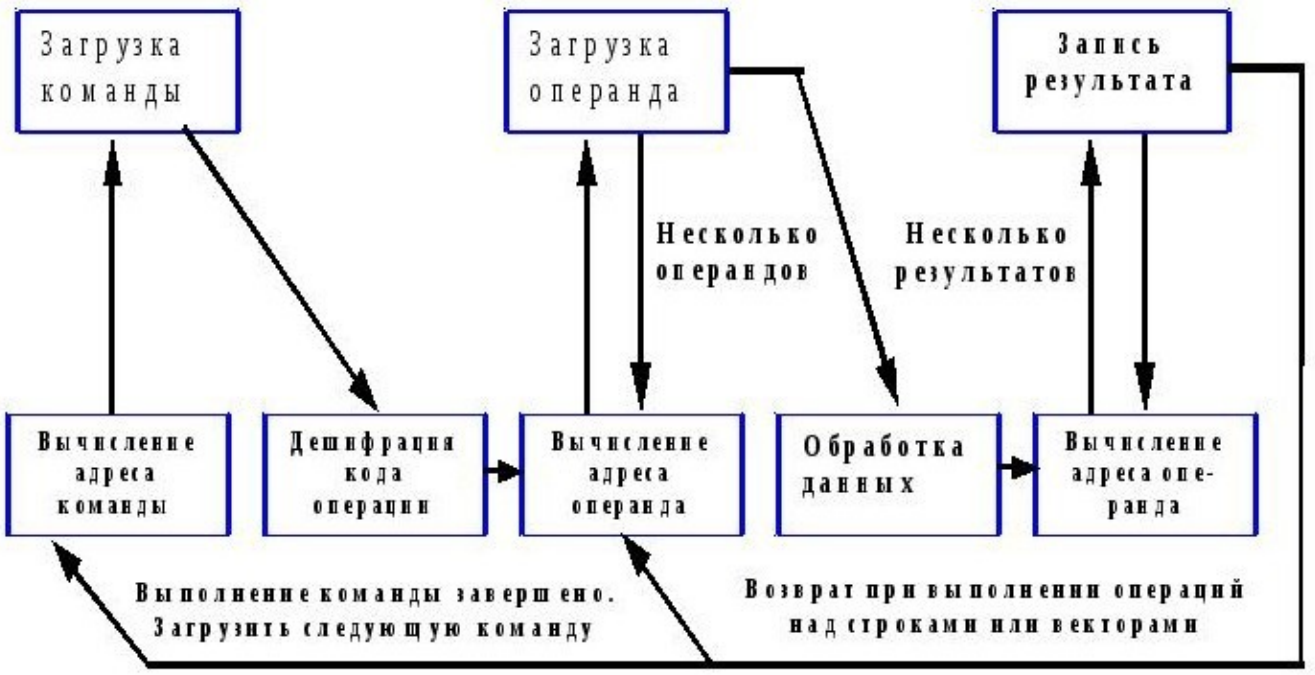
АО (автомат операционный) служит для хранения блоков информации (“*слов*”), выполнения набора микроопераций и вычисления значений логических условий (является структурой, организованной для выполнения действий над информацией). На вход АО подаются входные данный D_I, которые в соответствии с алгоритмом операции преобразуются в выходные D_O; АО вырабатывает множество {x} осведомительных сигналов (логических условий) для *УА (управляющего автомата)*.

УА (управляющий автомат) генерирует последовательность управляющих сигналов {y}, обеспечивающую выполнение в АО заданной последовательности элементарных действий по реализации алгоритма выполняемой операции. Эта управляющая последовательность генерируется в соответствии с заданным алгоритмом при учёте логических условий {x}, формируемых АО. На вход УА поступает команда C, определяющая тип выполняемой операции. УА формирует сигнал g, отмечающий окончание операции и готовность выходных данных D_O.

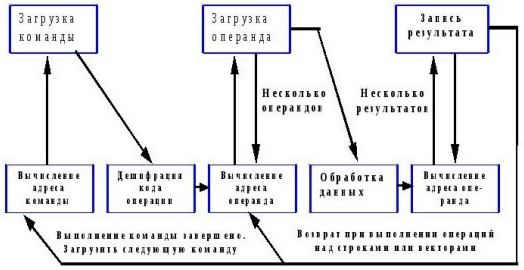
* ***АВТОМАТ*** – математическое понятие (математическая модель устройств, выполняющих некоторый процесс без непосредственного участия человека). ***А.*** часто представляют устройством (“*чёрным ящиком*”), имеющим конечное число входных и выходных параметров и некоторое множество внутренних состояний (*например – машина Тьюринга*).

Цикл выполнения машинной команды (команды уровня АНК)

1. Чтение инструкции и её декодирование
2. Поиск всех связанных данных, необходимых для обработки команды
3. Обработка инструкции
4. Запись результатов



Цикл выполнения машинной команды (команды уровня АНК)

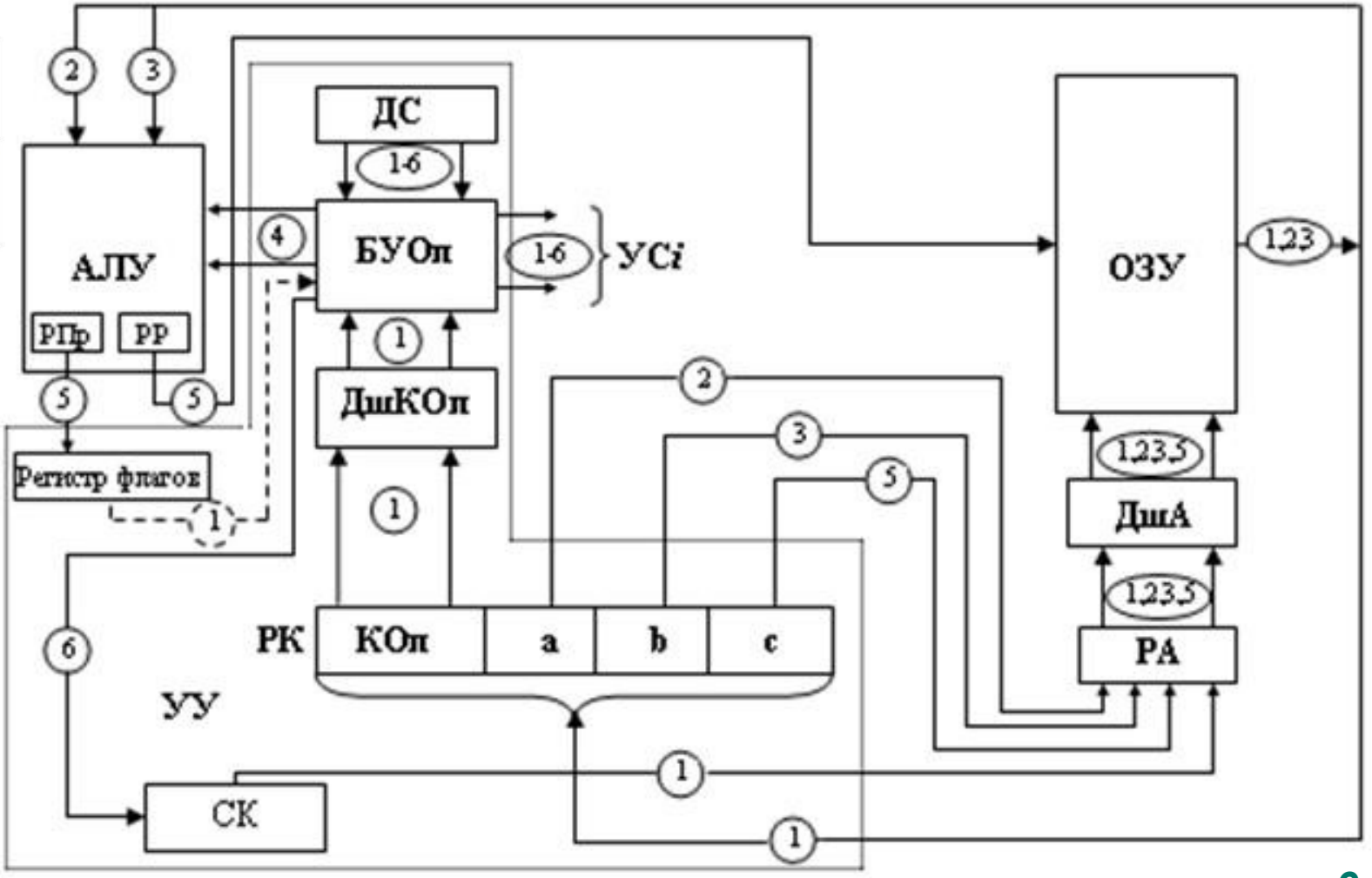


УУ – устройство управления (управляющий автомат)

- РК – регистр команды
- СК – счетчик команд
- РА – регистр адреса
- Дша – дешифратор адреса
- ОЗУ – оперативное запоминающее устройство
- Yc_i – управляющие сигналы
- БупОп – блок управления операциями
- ДшКОп – дешифратор кода операции
- ДС – дополнительные сигналы

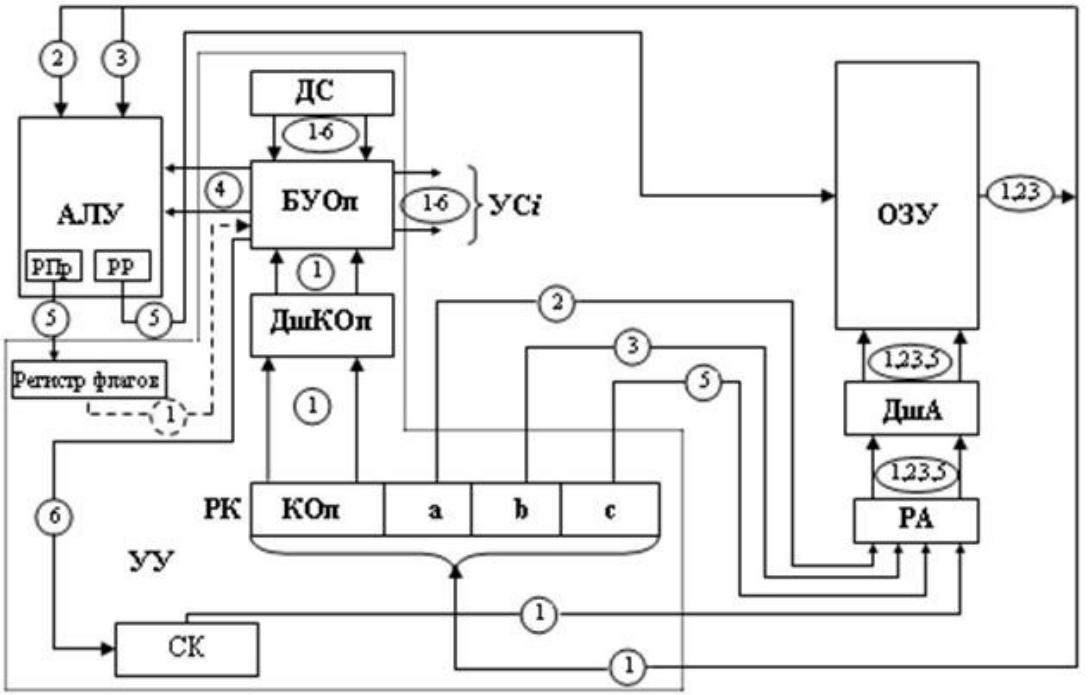
АЛУ – арифметико-логическое устройство (операционный автомат)

- РР – регистр результата
- РПР – регистр признаков

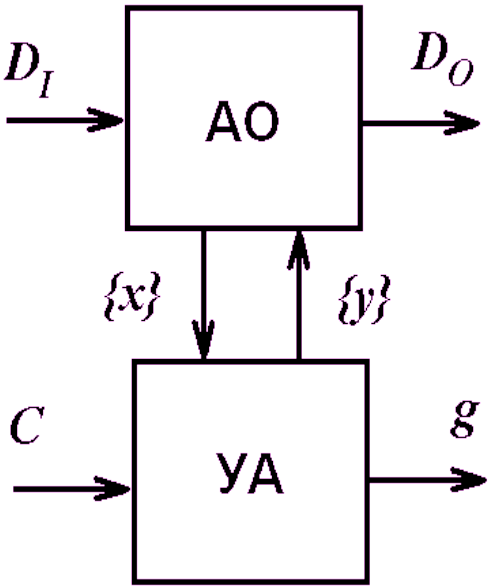


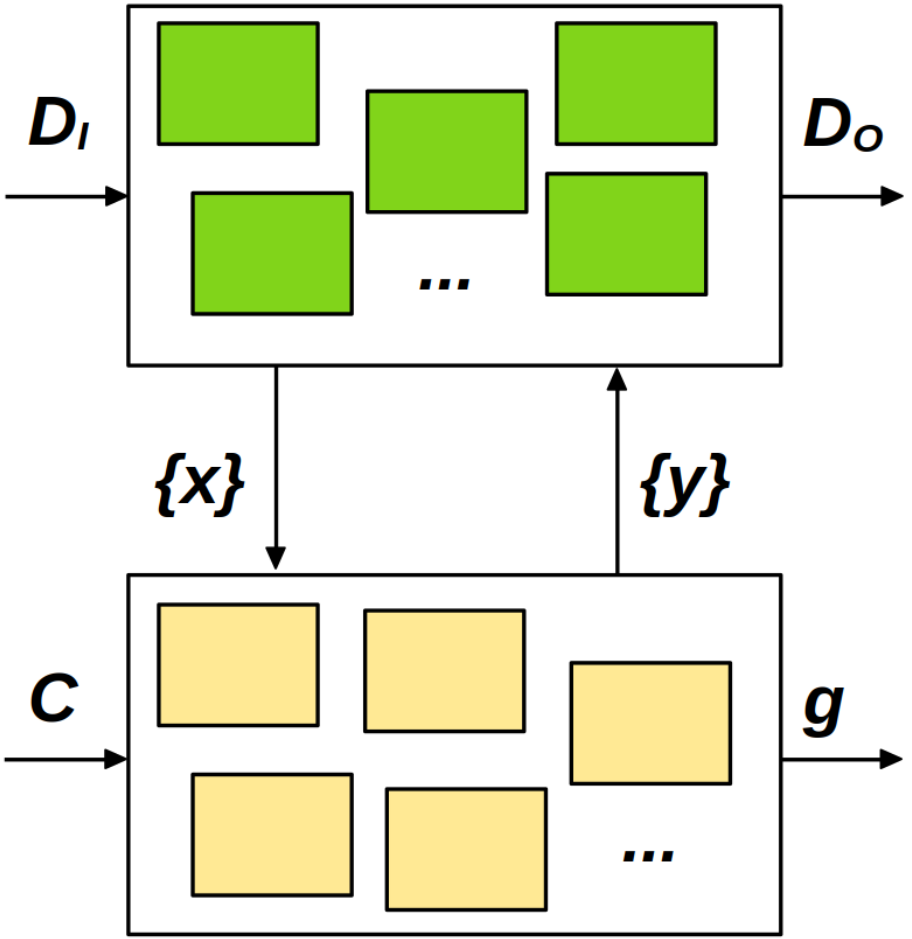
Цикл выполнения машинной команды (команды уровня АНК)

1) Выборка исполняемой команды из ОЗУ. Адрес со счетчика команд (СК) передается РА. Команда поступает в регистр команд (РК).
Дешифратор кода операции (ДшКОп) определяет и настраивает блок управления операциями (БУОп), на выходах которого формируются управляющие сигналы (УСi), для автоматического выполнения всего цикла команды вплоть до занесения в РК новой команды. Если данная команда не является командой перехода, то реализуется следующая последовательность этапов как продолжение первого.
2) Выборка первого операнда (a) с передачей в АЛУ.
3) Выборка второго операнда (b).
4) Выполнение операции. Результат операции сохраняется во внутреннем регистре результата (РР), а признаки результата – в регистре признаков АЛУ.
5) Запись результата операции в ОЗУ. Признаки результата записываются в регистр флагов, из которого они передаются в БУОп на случай, если очередная считанная в РК команда окажется командой условного перехода.
6) Формирование адреса следующей команды. Считая, что выполненная команда занимает в памяти Δ ячеек, получим, что суть этого этапа заключается в следующем изменении счетчика команд: СК = СК + Δ. На этом заканчивается цикл выполнения команды: в СК сформирован адрес следующей команды k +Δ. Выполнение этого этапа может совмещаться с выполнением предшествующих этапов, что и реализовано в большинстве ЭВМ.



От чего зависит эффективность управления операционным автоматом?



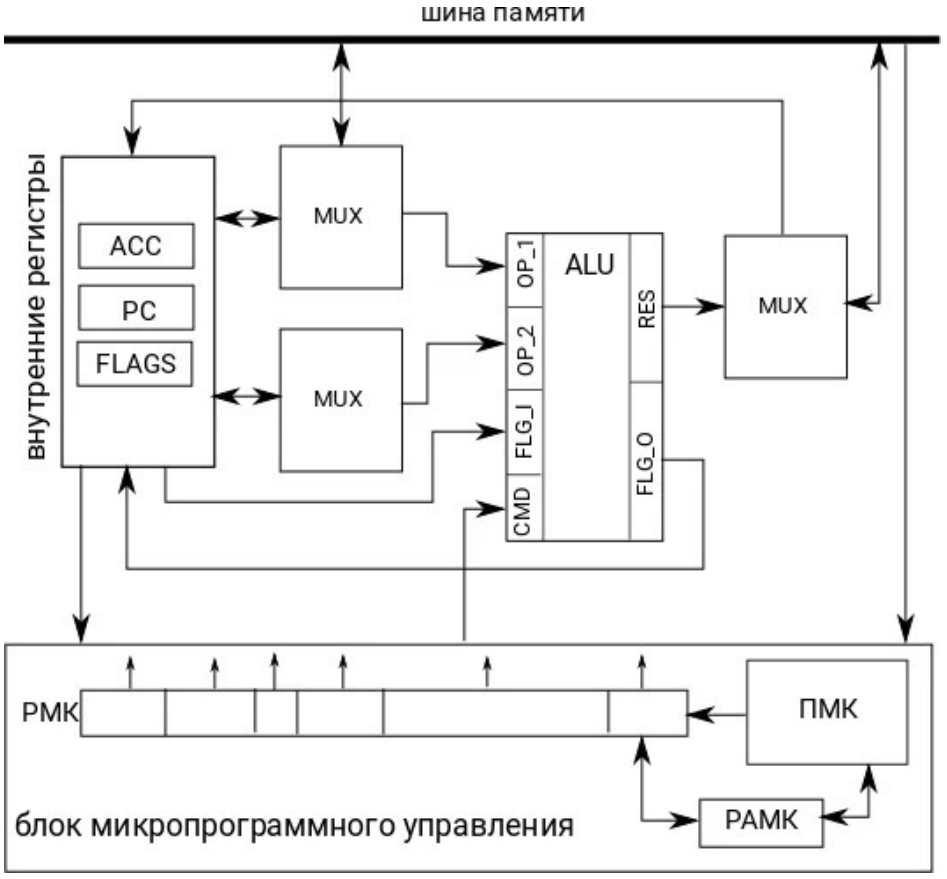


Подходы к реализации цикла выполнения команд

1. *Микрокод*
2. *Конвейеризация команд*
3. *Кэширование*
4. *Прогнозировани ветвлений*
5. *Суперскалярность*
6. *Внеочередное исполнение*
7. *Переименование регистров*
8. *Многопроцессорность и многопоточность*

Микрокод

Микропрограммирование процессоров (микрокод) предложен в **1953** г. (Вилкс и Стрингер). Появление микропрограммного устройства управления вместо прямых аппаратных решений.



Микрокод

Микропрограммы состоят из **микроинструкций** (элементарных операций), управляющих процессором на самом низком уровне. Примеры микроинструкций:

- Подсоединить Регистр 1 ко входу «А» АЛУ
- Подсоединить Регистр 7 ко входу «Б» АЛУ
- Настроить АЛУ на выполнение операции сложения
- Установить разряд переноса АЛУ в ноль
- Сохранить результат операции в Регистр 8
- Обновить «коды состояния» из флагов АЛУ («Отрицательное», «Ноль», «Переполнение», «Перенос»)
- Установить указатель микрокоманд на микроинструкцию номер nnn

Микроинструкции (микрокоманды) состоят из **микроопераций**.

Микрооперации — элементарные преобразования над данными; Примеры: скопировать операнд в регистр;

- установить флаг в 1;
- выставить операнд на шину;
- увеличить содержимое верхушки стека на 1, и т. д.

Микрокод

Достоинства:

- более простой способ разработки управляющего устройства процессора (написать микропрограмму проще, чем разработать соответствующую схему)
- уменьшение аппаратных затрат (компактность)
- легкость в исправлении ошибок
- удобство в расширении набора команд (стремление к CISC)

Недостатки:

- снижение производительности по сравнению со схемными решениями.
- невозможно повысить скорость выполнния команд за счет фиксации микропрограммного устройства управления

Конвейеризация команд

Одна из первых техник повышения производительности процессоров с последовательным выполнением команд

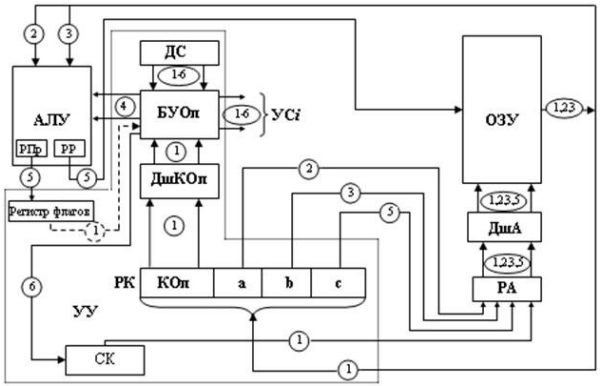
Одновременная обработка нескольких команд в конвейере команд с возможным их параллельным или конвейерным выполнением в разных АЛУ при определении независимости друг от друга.

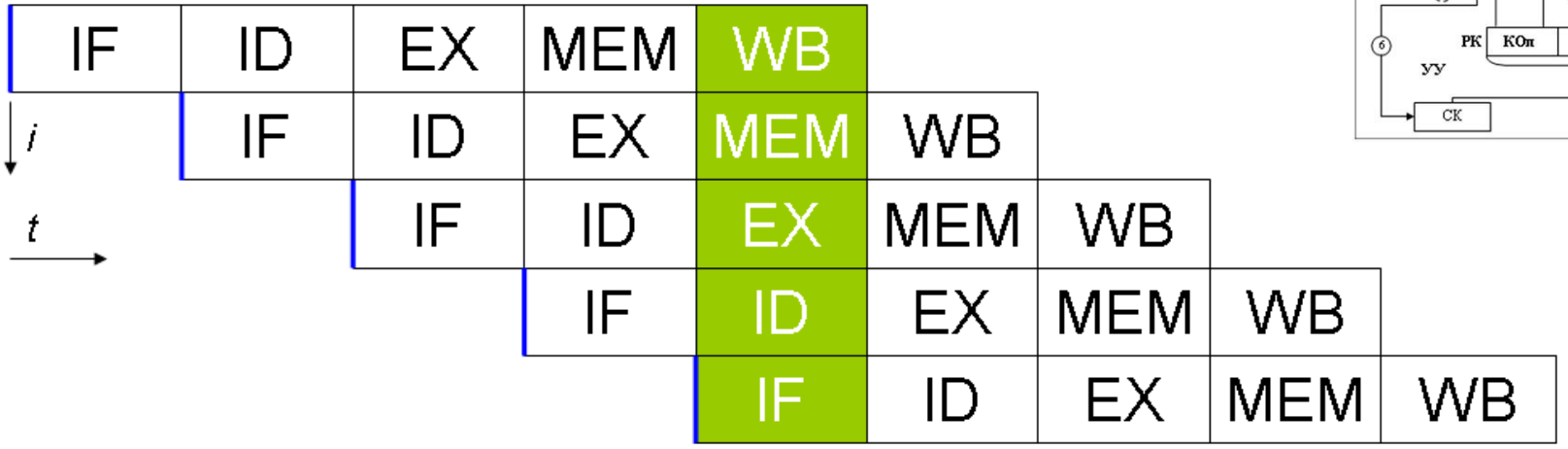
Процессор в целом функционирует на манер сборочной линии с инструкциями, поступающими с одной стороны и результатами, выходящими с другой.

Практически все современные процессоры имеют конвейеры команд. Зачастую в сочетании с микрокодом.

Конвейеризация команд

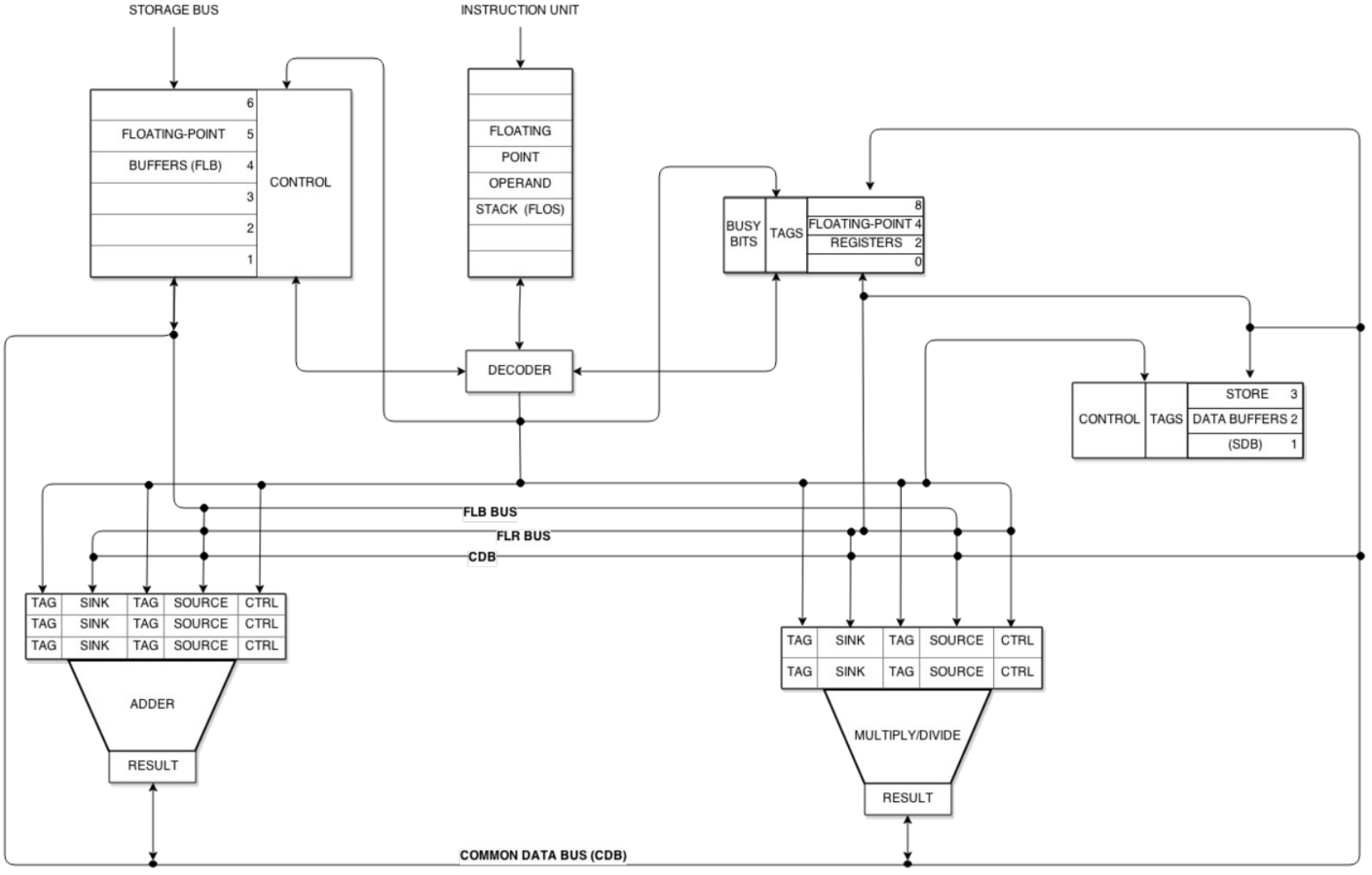
IF (Instruction Fetch) — получение инструкции,
ID (Instruction Decode) — раскодирование инструкции,
EX (Execute) — выполнение,
MEM (Memory access) — доступ к памяти,
WB (Register write back) — запись в регистр.





Простой пятиуровневый конвейер в RISC-процессорах.

Конвейеризация команд



Конвейер команд в IBM 360/91

Конвейеризация команд

Конфликты конвейера (hazards)

Препятствуют выполнению очередной команды из потока команд в предназначенном для неё такте. Уменьшают реальное ускорение, могут вызвать остановку конвейера.

Структурные конфликты
Возникают из-за конфликтов ресурсов. Аппаратура не может поддерживать все возможные комбинации одновременно выполняемых команд. Например, некоторый функциональный блок не полностью конвейеризован.

Конфликты по данным
Возникают при зависимости одной команды от результатов предыдущей. Конвейер приостанавливается до разрешения конфликта.

Конфликты по управлению
Возникают при конвейерном выполнении условных передач управления и других команд, изменяющих значение программного счетчика.

Для разрешения конфликта нужно, чтобы некоторые команды в конвейере могли продолжать выполняться, в то время как другие были задержаны.

Конвейеризация команд

Достоинства:

- Время цикла процессора уменьшается, таким образом увеличивая скорость обработки инструкций в большинстве случаев.
- Некоторые комбинационные логические элементы, такие, как сумматоры или умножители, могут быть ускорены путём увеличения количества логических элементов. Использование конвейера может предотвратить ненужное наращивание количества элементов.

Недостатки:

- Бесконвейерный процессор исполняет только одну инструкцию. Это предотвращает задержки веток инструкций, и проблемы, связанные с последовательными инструкциями, которые исполняются параллельно. Следовательно, схема такого процессора проще, и он дешевле для изготовления.
- Задержка инструкций в бесконвейерном процессоре слегка ниже, чем в конвейерном эквиваленте. Это происходит из-за того, что в конвейерный процессор должны быть добавлены дополнительные триггеры.
- У бесконвейерного процессора скорость обработки инструкций стабильна. Производительность конвейерного процессора предсказать намного сложнее, и она может значительно различаться в разных программах.

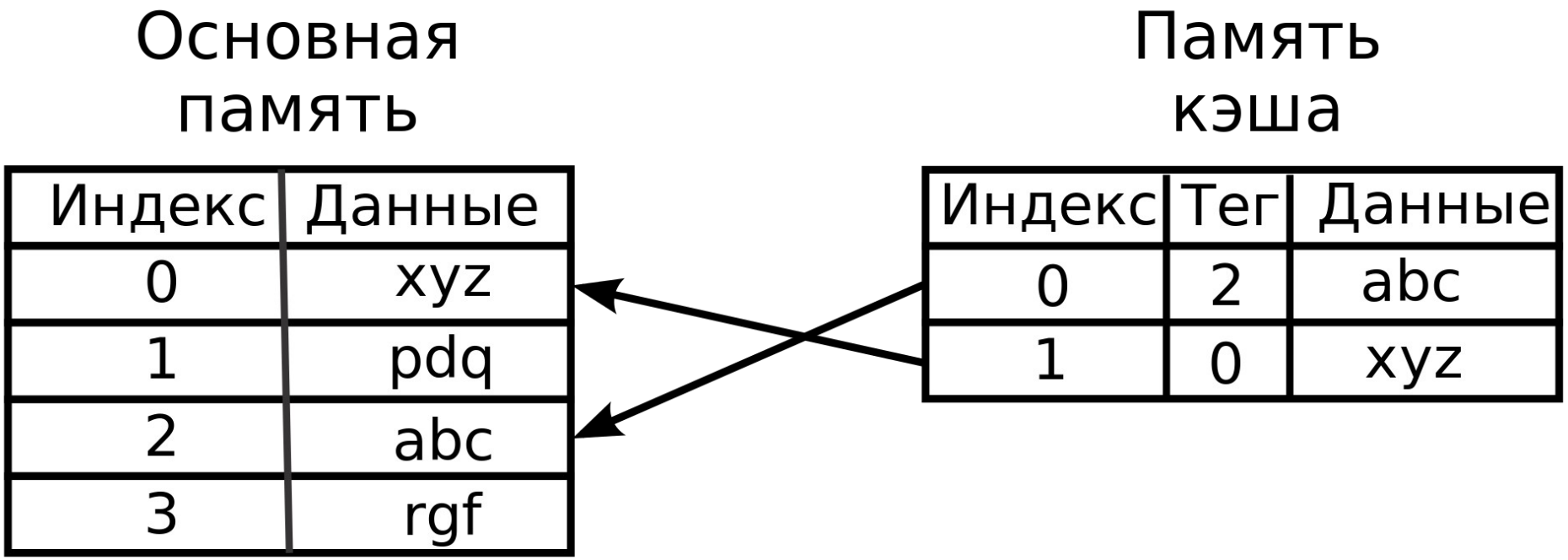
Кэширование

Кэш (кеш, cache) — промежуточный буфер с быстрым доступом к нему, содержащий информацию, которая может быть запрошена с наибольшей вероятностью.

Кэширование – размещение на кристалле очень быстрой кэш памяти, доступ к которой происходил всего за несколько тактов процессора, в отличие от большого числа тактов (и меньшей тактовой частоте) при работе с основной памятью.

Кэш микропроце́ссора — кэш (сверхоперативная память), используемый микропроцессором компьютера для уменьшения среднего времени доступа к компьютерной памяти. Использует небольшую, очень быструю память, которая хранит копии часто используемых данных из основной памяти. Если большая часть запросов в память будет обрабатываться кэшем, средняя задержка обращения к памяти будет приближаться к задержкам работы кэша.

Кэш память и конвейеры хорошо дополняют друг друга.



Отображение кэша памяти процессора в основной памяти.

Прогнозирование ветвлений

Прогнозирование ветвления — оборудование делает обоснованное решение о том, какая из ветвей будет выбрана для исполнения. Используются достаточно сложные статистические системы прогнозирования. Это позволяют аппаратуре предварительно считать инструкции, не дожидаясь результата из регистра.

Спекулятивное исполнение — дальнейшее развитие прогнозирования, при котором инструкции из предсказанного пути не только считываются, но и исполняются до того, как становится точно известно, будет ли выбрана ветвь. *Помогает повысить производительность, но вызывает риск большой потери времени, при ошибочном решении (**?!**).*

Предвыборка кода — выдача запросов со стороны процессора в оперативную память для считывания инструкций заблаговременно, до того момента, как эти инструкции потребуется исполнять. В результате инструкции загружаются в кэш.

Прогнозирование ветвлений

Модуль предсказания переходов (прогнозирования ветвлений) (предсказатель переходов, branch prediction unit) — устройство, предсказывающее, будет ли выполнен условный переход в исполняемой программе. Позволяет сократить время простоя конвейера за счёт предварительной загрузки и исполнения инструкций, которые должны выполниться после выполнения инструкции условного перехода.

Статические методы предсказания ветвлений являются наиболее простыми. Различные типы переходов:

- либо выполняются всегда;
- либо не выполняются никогда.

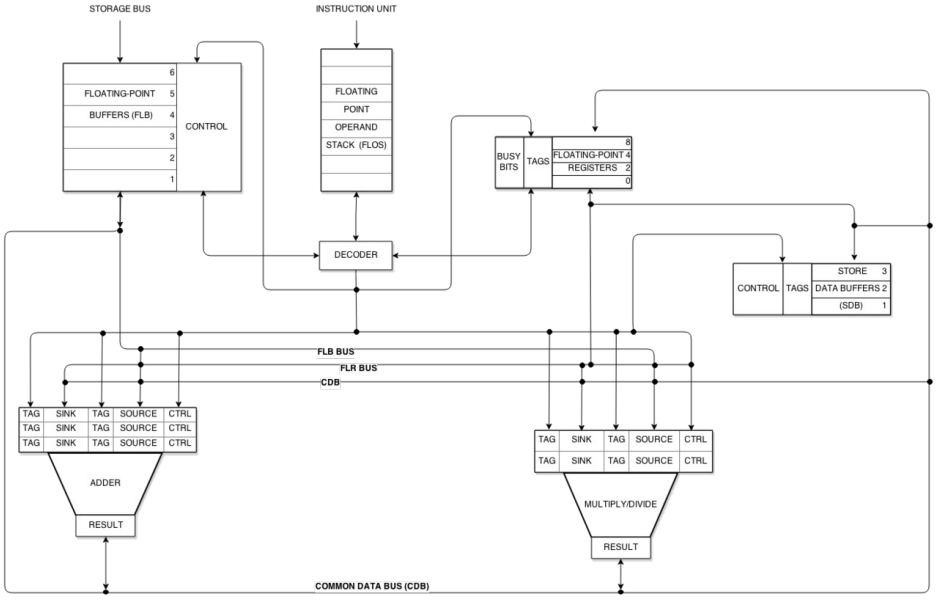
Динамические методы подразумевают анализ истории ветвлений.

В современных процессорах статические методы используются лишь тогда, когда использование динамических методов невозможно.

Суперскалярность

Использование нескольких одинаковых функциональных блоков, таких как АЛУ и др.

Суперскалярный процессор (superscalar processor) — процессор, поддерживающий параллелизм на уровне инструкций за счёт включения в состав его вычислительного ядра нескольких одинаковых функциональных узлов (таких как АЛУ, FPU, умножитель (integer multiplier), сдвигающее устройство (integer shifter) и другие устройства). Планирование исполнения потока инструкций осуществляется динамически вычислительным ядром (не статически компилятором).



Внеочередное исполнение

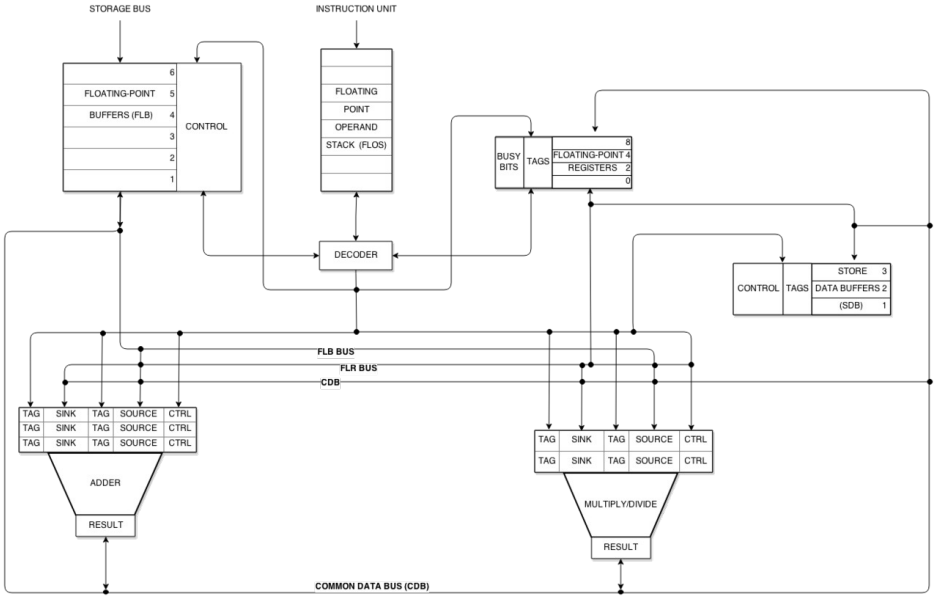
Внеочередное исполнение (out-of-order execution) команд — исполнение не в порядке следования в коде (in-order execution), а в порядке готовности к выполнению. Осуществляются следующие действия:

- считывание команды;
- помещение команды в очередь I (instruction queue);
- ожидание, пока операнды любой команды в очереди I, станут доступны;
- передача команды, операнды которой доступны, на выполнение соответствующему исполнительному модулю;
- выполнение команды соответствующим модулем;
- запись результата выполнения команды в очередь II;
- извлечение из очереди II результатов выполнения тех команд, перед которыми в очереди I не осталось невыполненных команд, и запись результатов в регистровый файл; удаление таких команд из очереди I.

Внеочередное исполнение

Особенности:

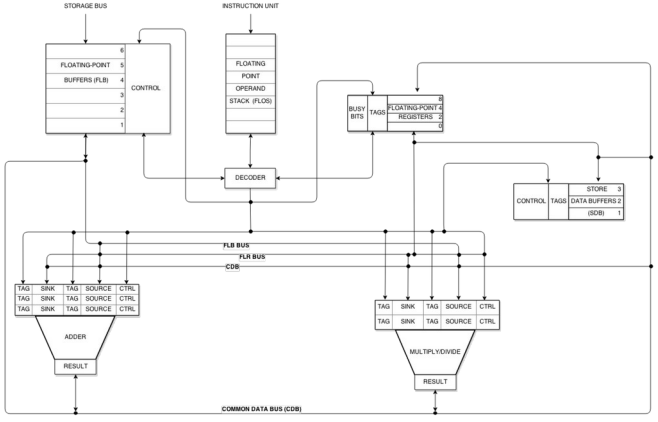
- *раньше других может быть выполнена команда, операнды которой будут готовы раньше операндов других команд;*
- *команда, поставленная в очередь I позже, может быть выполнена раньше;*
- *время простоя, вызванное ожиданием готовности операндов, уменьшается за счёт ожидания готовности операндов нескольких команд;*
- *за счёт использования очереди II запись результатов в регистровый файл осуществляется в порядке следования команд в машинном коде (создаётся видимость выполнения команд по порядку).*



Переименование регистров

Техника, позволяющая избежать ненужного последовательного выполнения команд программы вследствие использования ими одних и тех же регистров.

Если две группы команд, используют один регистр, то одна должна предшествовать другой только для того, чтобы освободить этот регистр. Но если вторую группу инструкций перенаправить на другой однотипный регистр, то обе группы могут выполняться параллельно.



Переименование регистров - преобразование программных ссылок на архитектурные регистры в ссылки на физические регистры. Позволяет ослабить влияние ложных взаимозависимостей за счёт использования большого количества физических регистров вместо ограниченного количества архитектурных (x86-совместимые процессоры архитектуры Intel P6 содержат 40 физических регистров).

Процессор отслеживает, состояние каких физических регистров соответствует состоянию архитектурных, а выдача результатов осуществляется в порядке, который предусмотрен программой.

Многопроцессорность и многопоточность

Многопроцессорность (Мультипроцессорность, Многопроцессорная обработка, Multiprocessing) — использование двух или большего количества физических процессоров в одной компьютерной системе.

Устройство называется многопроцессорным, если в его составе используется два или более физических процессора.

Многопото́чность (Multithreading) — свойство платформы (процессора, операционной системы, виртуальной машины и т. д.) или приложения, состоящее в том, что процесс, порождённый в операционной системе, может состоять из нескольких потоков, выполняющихся «параллельно», то есть без предписанного порядка во времени. При выполнении некоторых задач такое разделение может достичь более эффективного использования ресурсов вычислительной машины.

Такие ***потоки*** называют также ***потоками выполнения (thread of execution)***; иногда называют «***нитями***» (***thread***).

Используемые источники

1. Таненбаум Э. Архитектура компьютера. 6-е изд. — СПб.: Изд. Питер, 2017. – 816 с.

2. Коуги П.М. Архитектура конвейерных ЭВМ. — М.: Радио и связь, 1985. – 360 с.

3. Микроархитектура: https://ru.wikipedia.org/wiki/Микроархитектура

4. Микроконтроллеры для начитающих. Часть 4. Очень кратко о микропрограммах: https://zen.yandex.ru/media/id/5b935f60343d6c00a9f52b06/mikrokontrollery-dlia-nachitaiuscih-chast-4-ochen-kratko-o-mikroprogrammah-5e0d74f9d7859b00b4e61e96?utm_source=serp

5. Вычислительный конвейер: https://ru.wikipedia.org/wiki/Вычислительный_конвейер

6. Кэш: https://ru.wikipedia.org/wiki/Кэш

7. Кэш процессора: https://ru.wikipedia.org/wiki/Кэш_процессора

8. Предвыборка кода: https://ru.wikipedia.org/wiki/Предвыборка_кода

9. Предсказатель переходов: https://ru.wikipedia.org/wiki/Предсказатель_переходов

10. Суперскалярность: https://ru.wikipedia.org/wiki/Суперскалярность

11. Внеочередное исполнение: https://ru.wikipedia.org/wiki/Внеочередное_исполнение

12. Переименование регистров: https://ru.wikipedia.org/wiki/Переименование_регистров

Используемые источники