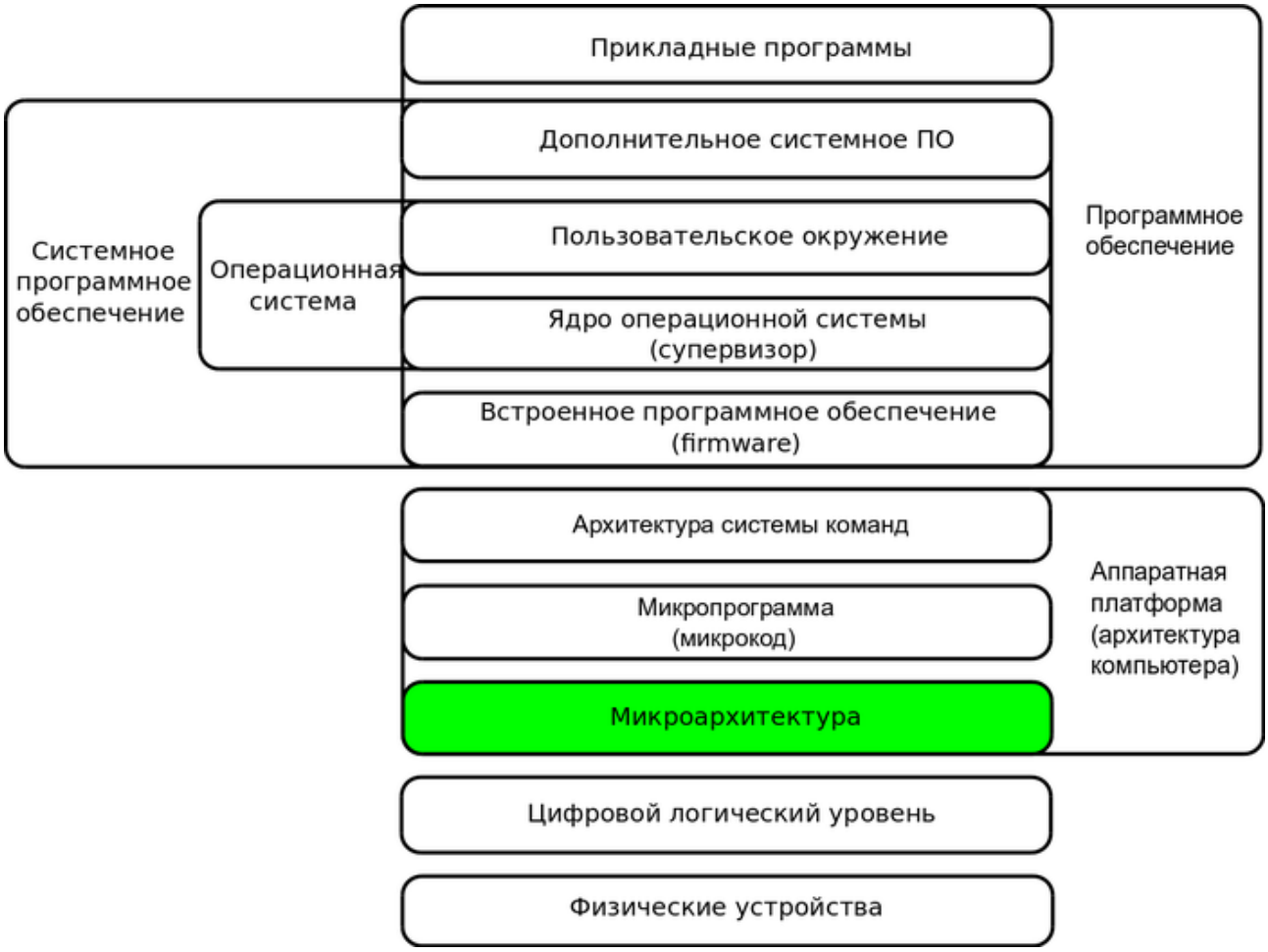
ЭВМ подготовка Лабы 3-5



Микроархитектура микропроцессора – способ реализации архитектуры набора команд (АНК) на микросхеме процессора. На этом уровне определяется:
конструкция и взаимосвязь основных блоков микропроцессора; структура ядер, исполнительных устройств, АЛУ, а также их взаимодействия;
количество блоков предсказания переходов;
организация конвейеров;
организация кэш-памяти;
взаимодействие с внешними устройствами.
Пониманию эволюции микроархитектур процессоров способствует знание «тик-так» стратегии. Эта стратегия была разработана фирмой Intel и заключается в следующем: разработка микропроцессоров делится на два этапа «тик» и «так». «Тик» предполагает уменьшение техпроцесса при создании нового микропроцессора, при этом собственно микроархитектура не изменяется. На следующем этапе «так» происходит изменение микроархитектуры на основе существующего техпроцесса.
Микропроцесор Intel с кодовым именем Cannonlake, созданный по техпроцессу 10 нм, является усовершенствованным Skylake с техпроцессом 14 нм.

Архитектура набора команд (АНК) определяет программируемую часть ядра микропроцессора. На этом уровне определяются реализованные в микропроцессоре конкретного типа:
архитектура памяти;
взаимодействие с внешними устройствами ввода-вывода;
режимы адресации;
регистры;
машинные команды;
различные типы внутренних данных (например, с плавающей запятой, целочисленные типы и т.д.);
обработчики прерываний и исключительных состояний.
Можно сказать, что архитектура компьютера является комбинацией микроархитектуры, микрокода и АНК.
Способы повышения производительности

Суперскалярность

Суперскалярная архитектура позволяет поддерживать параллелизм на уровне инструкций, т.е. выполнять эти инструкции одновременно (за один такт) благодаря использованию нескольких одинаковых функциональных узлов

Конвейеризация

Конвейеризация предполагает использование конвейера инструкций. Условно, любой процессор, выполняя короткую программу,
совершает следующие шаги:

1. Читает инструкцию.
2. Декодирует инструкцию.
3. Ищет все данные, необходимые для реализации инструкции.
4. Обрабатывает инструкцию.
5. Записывает результат.

Динамическое исполнение

Технология динамического исполнения объединяет три метода обработки данных, позволяющих обеспечить, в том числе, и эффективную работу конвейеров.

1. Анализ потока данных – просмотр исполняемой программы на несколько шагов вперед и составление графика исполнения инструкций в оптимальной, а не естественной последовательности.

2. Множественное предсказание ветвлений. Процессор может предвидеть разделение потока команд. С большой точностью (более 90 %) он предсказывает, в какой области памяти можно найти следующие команды. Это оказывается возможным, поскольку в процессе исполнения команды процессор просматривает программу на несколько шагов вперед.

3. Спекулятивное исполнение. Процессор выполняет инструкции по мере их поступления в оптимизированной последовательности (спекулятивно). Поскольку выполнение инструкций происходит на основе предсказания

ветвлений, результаты сохраняются как предположительные или «спекулятивные». На конечном этапе порядок инструкций восстанавливается.

Hyper-Threading

Технология Hyper-Threading реализует идеи одновременной многопоточности, когда каждое физическое ядро процессора видится операционной системой, как два логических, что позволяет выполнять две задачи одновременно.

В ядре сохраняется состояние сразу двух потоков, у каждого логического ядра есть свой набор регистров, свой счетчик команд и свой блок работы с прерываниями для каждого потока. Однако остальные

элементы ядра для обоих потоков – общие, и делятся между ними. Когда по какой-либо причине один из потоков освобождает процессорный блок (исполнительное устройство), другой поток его использует

Универсальные микропроцессоры предназначены для решения задач цифровой обработки различного типа информации от инженерных расчетов до работы с базами данных, не связанных жесткими

ограничениями на время выполнения задания. Этот класс микропроцессоров наиболее широко распространен и включает в себя CISC и RISC-микропроцессоры.

CISC (Complex instruction set computer) – это процессоры со сложным набором команд. Архитектура CISC характеризуется:

сложными и многоплановыми инструкциями; большим набором различных инструкций; нефиксированной длиной инструкций; многообразием режимов адресации.

RISC (Reduced Instruction Set Computer) – процессоры с сокращенным набором инструкций. Архитектура RISC характеризуется:

фиксированной длиной инструкций;

небольшим набором стандартизированных инструкций;

большим количеством регистров общего назначения;

отсутствием микрокода;

меньшим энергопотреблением, по сравнению с CISC- процессорами аналогичной производительности;

более простым внутренним устройством;

меньшим количеством транзисторов, по сравнению с CISC- процессорами аналогичной производительности;

отсутствием сложных специализированных блоков в ядре процессора.

Процессоры с архитектурой CISC появились первыми. Разработчики стремились сделать ЭВМ более функциональными и в то же время простыми для программирования. Естественно, для программистов вначале было удобнее иметь широкий набор команд, чем реализовывать каждую функцию целой отдельной подпрограммой. Однако такая ситуация продолжалась недолго. Во-первых, с появлением языков высокого уровня отпала необходимость программирования в машинных кодах и на ассемблере, и, во-вторых, со временем количество различных команд сильно выросло, а сами инструкции

усложнились. Однако большинство программистов, использовали какой-то определенный набор инструкций, игнорируя наиболее сложные инструкции.
В процессорах с RISC-архитектурой используются простые и стандартизированные инструкции. Следовательно, такие инструкции проще декодировать и выполнять, устройство процессора становится проще. В результате, процессор становится дешевле, и появляется возможность дополнительно поднять его тактовую частоту, за счет упрощения внутренней структуры и уменьшения количества транзисторов, или снизить энергопотребление.
Также простые RISC-инструкции гораздо проще распараллеливать, чем CISC-инструкции, а, следовательно, появляется возможность больше загрузить конвейер, ввести дополнительные блоки обработки инструкций и т.д.
Несмотря на явные преимущества RISC-архитектура не получила такого же широкого распространения, как и CISC, связанно это в первую очередь с тем, что огромное количество программного обеспечения, написанного для CISC-процессоров не совместима с RISC-процессорами. Тем не менее разработчики вышли из положения создав гибридные структуры, где в CISC-процессорах используется RISC-ядро, а сложные команды заменяются на микропрограммы.
Так, формально процессоры, построенные по архитектуре Intel x86 – CISC-процессоры, однако начиная с Intel486DX, они стали CISC- процессорами с RISC-ядром. Непосредственно перед исполнением они преобразуют CISC-инструкции процессоров x86 в более простой набор внутренних инструкций RISC. Для этого в микропроцессор встраивается аппаратный транслятор, превращающий команды x86 в команды внутреннего RISC-процессора. При этом одна команда x86 может порождать несколько RISC-команд. Исполнение команд происходит на суперскалярном конвейере одновременно по несколько штук.
Архитектура RISC продолжает развиваться, эволюционируя в MISC и VLIW-архитектуры.

Классификация процессоров - по количеству и скорости выполнения команд
(RISK и CISK это архитектура процессора)
- RISK - сокращенный набор команд
- CISK(например x86/IA32 - интеловская архитектура 32) - длинна команд разная
- сейчас гибридная архитектура(CISK процессор имеет RISK ядра)
Классификация процессоров по применению:

– *однопрограммные процессоры* (выполняют одновременно одну программу) и *многопрограммные* (одновременно выполняют несколько программ);
– *мультипроцессоры* предполагают *параллелизм на уровне процессоров,* являются высокопроизводительными системами. В них активными могут быть одновременно несколько процессоров, работа которых согласовывается единым программным обеспечением в целях исключения пересечений решаемых задач;

– *многоядерные процессоры* реализуются размещением нескольких ядер на одном кристалле. В частности, «двухъядерность» означает наличие на кристалле двух независимых процессоров, работающих одновременно (параллельно), что способствует увеличению производительности ЭВМ на 80-100%. Современные двухъядерные процессоры компании AMD построены на основе ядер Opteron и Athlon;
– *конвейерные процессоры* предполагают *параллелизм на уровне команд*. Выполнение любой команды разбивается в конвейере на несколько этапов (стадий), выполняемых определённой частью аппаратного обеспечения (блоками АЛУ). Причём эти блоки могут работать *параллельно*, т.е. после обработки текущей стадии команды освободившийся блок может сразу же переключиться на выполнение аналогичной стадии следующей команды, не дожидаясь, пока все стадии текущей команды будут окончательно выполнены. Различают *скалярные процессоры* (с одним конвейером) и *суперскалярные* (с двумя и более конвейерами, выполняющими параллельно независимые инструкции). Применение конвейерной технологии увеличивает скорость работы процессора в 5-10 раз при той же тактовой частоте;
– *сопроцессоры* (арифметические расширители) – дополнительные процессоры, предназначенные для расширения списка команд, выполняемых центральным процессором. Самостоятельно не используются. Сопряжение микропроцессора с сопроцессором обычно выполняется простым объединением их выводовч без дополнительных микросхем, а их взаимодействие поддерживается с помощью контроллера прерываний;
– *нейропроцессоры* – процессоры для нейроподобных систем [9], предназначенных для реализации систем искусственного интеллекта. Современные нейропроцессоры, например, 64-разрядный процессор Neuromatrix Л1879ВМ1, позволяют за счёт особенностей архитектуры (наличие векторного сопроцессора) достигать высокой производительности при работе с большими потоками данных и выполнении матричных операций;
– *сигнальные процессоры* – класс микропроцессоров для работы в измерительных и управляющих устройствах. Включают аналого-цифровые и цифроаналоговые преобразователи.

АНК(архитектура набора команд) - например x86…
Суперскалярность - распараллеливания между блоками

Суперскалярный процессор (англ. superscalar processor) — процессор, поддерживающий так называемый параллелизм на уровне инструкций (то есть, процессор, способный выполнять несколько инструкций одновременно) 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).

Способы увеличения производительности, которые могут использоваться совместно:

- Использование конвейера (англ. pipelining)
- увеличение количества функциональных узлов процессора (суперскалярность)
- увеличение количества ядер (многоядерность)
- увеличение количества процессоров (многопроцессорность)

Семейство процессоров и микроархитектура
Микроархитектура - составные части процессора, их взаимодействие

ПЗУ в процессоре, там находится микрокод с помощью микрокосма CISKовые команды бьют на короткие, так реализовали RISK
Операндов в операции сложения от 1 до 3
Система команд - полный перечень команд, который способна выполнять данная вычислительная машина
Счетчик адреса команд
АСК - архитектура системы команд - могут быть стековые, аккумуляторные, регистровыми, с выделенным доступом к памяти

Регистры сделаны на тригерах, которые не требуют доп подзарядки.
АЛУ- Выполняет любые математические операции. Состоит из регистров, сумматоров и блока УУ. Работает с переменными помещенными в регистры. Выполняемая команда имеют однотипную структуру: код операции и адресная часть.
Физически регистр - это узел ЭВМ, способный обнулять себя, смещать слова, преобразовывать последовательный код в параллельный.
Последовательный код - познаковая передача кода. Параллельный - передача кода "целиком"
Команда, код операции и адресная часть(адреса операндов, индексный и адресный регисры)
Последовательный и параллельный код
Машинное слово - величина измеряемая в битах равная размерности регистров или передающей шины.
Цикл процессора - период времени, за который выполняется команда исходного вида (может состоять из нескольких тактов)
Такт - (тактовый генератор колеблется с некой частотой, генерируя тактовые импульсы)
Микрокоманда - операция соответствующая 1 такту и реализуемая конкретной логической схемой.
РОН(регистры общего назначения) - передаваемые из или в память
Регистр адреса команды - содержит адрес команды
Счётчик адреса команды - увеличивает адрес текущей команды на 1
Регистр адреса (числа) - содержит адрес операнда
Регистр числа - содержит сам операнд

Регистр результатов
Сумматор - реализует операции сложения
Индексный регистр - для работы с адресами
Аккумулятор (регистр) - хранение промежуточных данных (буферная память)
Регистр флагов (состояний) - отслеживание корректности протекания процессов

Конвейеризация - Программная конвейеризация циклов (англ. software pipelining) — это техника, используемая компиляторами для оптимизации циклов по аналогии с вычислительным конвейером в микропроцессорах. Является формой внеочередного исполнения с той разницей, что переупорядочивание выполняется не процессором, а компилятором (либо, в случае ручной оптимизации, программистом). Некоторые компьютерные архитектуры, например Intel IA-64, имеют явную аппаратную поддержку для упрощения программной конвейеризации циклов.

Гипертрейдинг - Гиперпоточность (официальное название — hyper-threading technology, HTT или HT) — технология, разработанная компанией Intel для повышения производительности процессоров собственного производства. Стала исторически первой полноценной реализацией концепции одновременной многопоточности (англ. simultaneous multithreading, SMT), созданной в развитие технологии суперпоточности (англ. super-threading, реализовывавшей временную многопоточность). После включения гиперпоточности одно физическое процессорное ядро определяется операционной системой как два отдельных логических ядра. При определённых рабочих нагрузках использование гиперпоточности позволяет увеличить производительность процессора. Суть технологии: передача «полезной работы» (англ. useful work) бездействующим исполнительным устройствам (англ. execution units).

Почитать таблицу из методичек

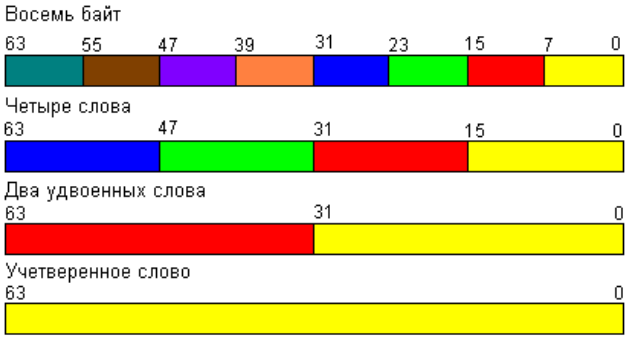
Тип команд	Выполняемые действия (или другие признаки)		
Команды пересылки	Пересылка данных между двумя регистрами или между регистром и ячейкой памяти. В некоторых процессорах реализуется пересылка между двумя ячейками памяти, а также групповая пересылка содержимого нескольких регистров в память или их загрузка из памяти		
Команды ввода и вывода	Реализуют пересылку данных из регистра процессора (ОП) во внешнее устройство или прием данных из внешнего устройства в регистр (ОП)		
Команды обработки данных (O1 — первый операнд, O2 — второй)	Короткие операции (один такт)	Логические	Логическое сложение (для каждого бита O1 и O2 осуществляется операция или)
			Логическое умножение (для каждого бита O1 и O2 осуществляется операция и)
			Инверсия (в O1 все единицы заменяются на нули и наоборот)
			Сравнение логическое (если O1 = O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Арифметические	Сложение операндов
			Вычитание (сложение в обратном коде)
			Сравнение арифметическое (если O1 > O2, или O1 = O2, или O1 < O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Команды сдвига	Осуществляют арифметические, логические и циклические сдвиги адресуемых операндов на один или несколько разрядов
	Длинные операции (несколько тактов)	Сложение/вычитание с плавающей запятой	
		Умножение/деление с фиксированной и плавающей запятой	
Операции управления	Безусловный переход (ветвление, branch)	Загружает в CчAK новое содержимое, являющееся адресом следующей выполняемой команды	
	Вызов подпрограммы	Производится путем безусловной передачи управления с сохранением адреса возврата управления	
	Условный переход (conditional branch)	Производит загрузку в PC нового содержимого, если выполняются определенные условия	
	Команды организации программных циклов	Условный переход в зависимости от значения содержимого заданного регистра, который используется как счетчик циклов	
	Команды прерывания	Переход к одной из программ обслуживания исключений и прерываний	

Тип команд	Выполняемые действия (или другие признаки)		
Команды пересылки	Пересылка данных между двумя регистрами или между регистром и ячейкой памяти. В некоторых процессорах реализуется пересылка между двумя ячейками памяти, а также групповая пересылка содержимого нескольких регистров в память или их загрузка из памяти		
Команды ввода и вывода	Реализуют пересылку данных из регистра процессора (ОП) во внешнее устройство или прием данных из внешнего устройства в регистр (ОП)		
Команды обработки данных (O1 — первый операнд, O2 — второй)	Короткие операции (один такт)	Логические	Логическое сложение (для каждого бита O1 и O2 осуществляется операция или)
			Логическое умножение (для каждого бита O1 и O2 осуществляется операция и)
			Инверсия (в O1 все единицы заменяются на нули и наоборот)
			Сравнение логическое (если O1 = O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Арифметические	Сложение операндов
			Вычитание (сложение в обратном коде)
			Сравнение арифметическое (если O1 > O2, или O1 = O2, или O1 < O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Команды сдвига	Осуществляют арифметические, логические и циклические сдвиги адресуемых операндов на один или несколько разрядов
	Длинные операции (несколько тактов)	Сложение/вычитание с плавающей запятой	
		Умножение/деление с фиксированной и плавающей запятой	
Операции управления	Безусловный переход (ветвление, branch)	Загружает в CчAK новое содержимое, являющееся адресом следующей выполняемой команды	
	Вызов подпрограммы	Производится путем безусловной передачи управления с сохранением адреса возврата управления	
	Условный переход (conditional branch)	Производит загрузку в PC нового содержимого, если выполняются определенные условия	
	Команды организации программных циклов	Условный переход в зависимости от значения содержимого заданного регистра, который используется как счетчик циклов	
	Команды прерывания	Переход к одной из программ обслуживания исключений и прерываний	

Тип команд	Выполняемые действия (или другие признаки)		
Команды пересылки	Пересылка данных между двумя регистрами или между регистром и ячейкой памяти. В некоторых процессорах реализуется пересылка между двумя ячейками памяти, а также групповая пересылка содержимого нескольких регистров в память или их загрузка из памяти		
Команды ввода и вывода	Реализуют пересылку данных из регистра процессора (ОП) во внешнее устройство или прием данных из внешнего устройства в регистр (ОП)		
Команды обработки данных (O1 — первый операнд, O2 — второй)	Короткие операции (один такт)	Логические	Логическое сложение (для каждого бита O1 и O2 осуществляется операция или)
			Логическое умножение (для каждого бита O1 и O2 осуществляется операция и)
			Инверсия (в O1 все единицы заменяются на нули и наоборот)
			Сравнение логическое (если O1 = O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Арифметические	Сложение операндов
			Вычитание (сложение в обратном коде)
			Сравнение арифметическое (если O1 > O2, или O1 = O2, или O1 < O2, то некий *флаг* или *регистр* устанавливается в «1», иначе — в «0»)
		Команды сдвига	Осуществляют арифметические, логические и циклические сдвиги адресуемых операндов на один или несколько разрядов
	Длинные операции (несколько тактов)	Сложение/вычитание с плавающей запятой	
		Умножение/деление с фиксированной и плавающей запятой	
Операции управления	Безусловный переход (ветвление, branch)	Загружает в CчAK новое содержимое, являющееся адресом следующей выполняемой команды	
	Вызов подпрограммы	Производится путем безусловной передачи управления с сохранением адреса возврата управления	
	Условный переход (conditional branch)	Производит загрузку в PC нового содержимого, если выполняются определенные условия	
	Команды организации программных циклов	Условный переход в зависимости от значения содержимого заданного регистра, который используется как счетчик циклов	
	Команды прерывания	Переход к одной из программ обслуживания исключений и прерываний	

Расширенные инструкции процессора СИМД
Потоковое Расширенние SIMD (Streaming SIMD Extensions (Single Instruction, Multiple Data - одна команда, несколько элементов данных)) это обобщающее название всех возможностей процессора, которые созданы для повышения производительности в мультимедиа и информационных приложениях.

MMX - расширение появилось в Pentium MMX (P55, январь 1997) и включало в себя 57 новых команд, предназначенных для обработки звуковых и видеосигналов. Позднее их поддержка появилась в K6 (Little Foot) от AMD и в 6x86MX от Cyrix.



SSE - Данное расширение появилось в Pentium III (ядро Katmai, сентябрь 1999) и насчитывало 70 новых команд. Позднее в Athlon XP (начиная с Palomino) его стали поддерживать и процессоры AMD. Аббревиатура SSE расшифровывается как *Streaming SIMD Extensions* (потоковые SIMD расширения).

SSE2 - Следующее расширение, являющееся логическим продолжением MMX и SSE появилось в Pentium 4 (начиная с Willamette). В Athlon 64 появилось начиная с Clawhammer.

SSE3 - Следующий набор появился в Pentium 4 начиная с Prescott и Athlon 64 начиная с Venice. Это расширение, имевшее поначалу имело рабочее название *Prescott New Instruction*, но получившее в итоге не совсем верное с технической точки зрения название SSE3, призвано облегчить оптимизацию программ под SSE и SSE2.
3DNow!-Набор инструкций 3DNow! появился в AMD K6-2 (Chomper). Данный набор, состоящий из 21 команды, был оптимизирован для еще более узкой области, нежели "универсально-мультимедийный" Intel MMX, а именно: для наиболее ресурсоемких расчетов, связанных с 3D-графикойВертикальная и горизонтальная арифметика