МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

Факультет Издательского дела и полиграфии

Кафедра Информационных систем и технологий

ОТЧЕТ

по лабораторной работе №

на тему:

**«ТЕСТИРОВАНИЕ ЦП. ИССЛЕДОВАНИЕ**

**ПРОИЗВОДИТЕЛЬНОСТИ ЦП»**

**Цель работы** — изучить существующие способы оценки производительности вычислительных машин и получить базовые навыки сравнения производительности вычислительных машин.

# Теоретическая часть

Основу для сравнения различных типов компьютеров между собой дают стандартные методики измерения производительности. В процессе развития вычислительной техники появилось несколько таких стандартных методик. Они позволяют разработчикам и пользователям осуществлять выбор между альтернативами на основе количественных показателей, что дает возможность постоянного прогресса в данной области.

Единицей измерения производительности компьютера является время: компьютер, выполняющий тот же объем работы за меньшее время является более быстрым. Время выполнения любой программы измеряется в секундах. Часто производительность измеряется как скорость появления некоторого числа событий в секунду, так что меньшее время подразумевает большую производительность.

Однако в зависимости от того, что мы считаем, время может быть определено различными способами. Наиболее простой способ определения времени называется астрономическим временем, временем ответа (responsetime), временем выполнения (executiontime) или прошедшим временем (elapsedtime). Это задержка выполнения задания, включающая буквально все: работу процессора, обращения к диску, обращения к памяти, ввод/вывод и накладные расходы операционной системы. Однако при работе в мультипрограммном режиме во время ожидания ввода/вывода для одной программы, процессор может выполнять другую программу, и система не обязательно будет минимизировать время выполнения данной конкретной программы.Производительность определяется количеством вычислительной работы, выполняемой за единицу времени.

При оценках чаще всего выделяют устройства: процессор, оперативную память и дисковую память. Номинальная производительность характеризует только потенциальные возможности устройств. При работе в составе системы эти возможности полностью не используются. Степень их использования зависит от характера и количества решаемых задач, интенсивности входного потока задач, операционной системы, выполняющей функции распределения ресурсов системы и организации управления ими.

Для измерения времени работы процессора используется специальный параметр — время ЦП (CPU time), которое не включает время ожидания ввода/вывода или время выполнения другой программы. Очевидно, что время ответа, видимое пользователем, является полным временем выполнения программы, а не временем ЦП. Время ЦП может далее делиться на время, потраченное ЦП непосредственно на выполнение программы пользователя и называемое пользовательским временем ЦП, и время ЦП, затраченное операционной системой на выполнение заданий, затребованных программой, и называемое системным временем ЦП.

В ряде случаев системное время ЦП игнорируется из-за возможной неточности измерений, выполняемых самой операционной системой, а также из-за проблем, связанных со сравнением производительности машин с разными операционными системами. С другой стороны, системный код на некоторых машинах является пользовательским кодом на других и, кроме того, практически никакая программа не может работать без некоторой операционной системы. Поэтому при измерениях производительности процессора часто используется сумма пользовательского и системного времени ЦП.

В большинстве современных процессоров скорость протекания процессов взаимодействия внутренних функциональных устройств определяется не естественными задержками в этих устройствах, а задается единой системой синхросигналов, вырабатываемых некоторым генератором тактовых импульсов, как правило, работающим с постоянной скоростью. Дискретные временные события называются тактами синхронизации (clockticks), просто тактами (ticks), периодами синхронизации (clockperiods), циклами (cycles) или циклами синхронизации (clockcycles).Разработчики компьютеров обычно говорят о периоде синхронизации, который определяется либо своей длительностью (например, 10 наносекунд), либо частотой (например, 100 МГц). Длительность периода синхронизации есть величина, обратная к частоте синхронизации.

Таким образом, время ЦП для некоторой программы может быть выражено двумя способами: количеством тактов синхронизации для данной программы, умноженным на длительность такта синхронизации, либо количеством тактов синхронизации для данной программы, деленным на частоту синхронизации.



Таким образом, производительность ЦП зависит от трех параметров: такта (или частоты) синхронизации, среднего количества тактов на команду и количества выполняемых команд. Невозможно изменить ни один из указанных параметров изолированно от другого, поскольку базовые технологии, используемые для изменения каждого из этих параметров, взаимосвязаны: частота синхронизации определяется технологией аппаратных средств и функциональной организацией процессора; среднее количество тактов на команду зависит от функциональной организации и архитектуры системы команд; а количество выполняемых в программе команд определяется архитектурой системы команд и технологией компиляторов. Когда сравниваются две машины, необходимо рассматривать все три компоненты, чтобы понять относительную производительность.

**MIPS**

Одной из альтернативных единиц измерения производительности процессора (по отношению к времени выполнения) является MIPS — (миллион целочисленных команд в секунду). Имеется несколько различных вариантов интерпретации определения MIPS.

В общем случае MIPS есть скорость операций с целыми числами в единицу времени, т.е. для любой данной программы MIPS есть просто отношение количества команд в программе к времени ее выполнения. Таким образом, производительность может быть определена как обратная к времени выполнения величина, причем более быстрые машины при этом будут иметь более высокий рейтинг MIPS.

Положительными сторонами MIPS является то, что эту характеристику легко понять, особенно покупателю, и что более быстрая машина характеризуется большим числом MIPS, что соответствует нашим интуитивным представлениям. Однако использование MIPS в качестве метрики для сравнения наталкивается на три проблемы. Во-первых, MIPS зависит от набора команд процессора, что затрудняет сравнение по MIPS компьютеров, имеющих разные системы команд. Во-вторых, MIPS даже на одном и том же компьютере меняется от программы к программе. В-третьих, MIPS может меняться по отношению к производительности в противоположенную сторону.

Классическим примером для последнего случая является рейтинг MIPS для машины, в состав которой входит сопроцессор плавающей точки. Поскольку в общем случае на каждую команду с плавающей точкой требуется большее количество тактов синхронизации, чем на целочисленную команду, то программы, используя сопроцессор плавающей точки вместо соответствующих подпрограмм из состава программного обеспечения, выполняются за меньшее время, но имеют меньший рейтинг MIPS. При отсутствии сопроцессора операции над числами с плавающей точкой реализуются с помощью подпрограмм, использующих более простые команды целочисленной арифметики и, как следствие, такие машины имеют более высокий рейтинг MIPS, но выполняют настолько большее количество команд, что общее время выполнения значительно увеличивается. Подобные аномалии наблюдаются и при использовании оптимизирующих компиляторов, когда в результате оптимизации сокращается количество выполняемых в программе команд, рейтинг MIPS уменьшается, а производительность увеличивается.

**MFLOPS**

Измерение производительности компьютеров при решении научно-технических задач, в которых существенно используется арифметика с плавающей точкой, всегда вызывало особый интерес. Именно для таких вычислений впервые встал вопрос об измерении производительности, а по достигнутым показателям часто делались выводы об общем уровне разработок компьютеров. Обычно для научно-технических задач производительность процессора оценивается в MFLOPS (миллионах чисел-результатов вычислений с плавающей точкой в секунду, или миллионах элементарных арифметических операций над числами с плавающей точкой, выполненных в секунду).

Как единица измерения, MFLOPS, предназначена для оценки производительности только операций с плавающей точкой, и поэтому не применима вне этой ограниченной области. Например, программы компиляторов имеют рейтинг MFLOPS близкий к нулю вне зависимости от того, насколько быстра машина, поскольку компиляторы редко используют арифметику с плавающей точкой.

Рейтинг MFLOPS зависит от машины и от программы. Этот термин менее безобидный, чем MIPS. Он базируется на количестве выполняемых операций, а не на количестве выполняемых команд. По мнению многих программистов, одна и та же программа, работающая на различных компьютерах, будет выполнять различное количество команд, но одно и то же количество операций с плавающей точкой. Именно поэтому рейтинг MFLOPS предназначался для справедливого сравнения различных машин между собой.

Ливерморские циклы — это набор фрагментов фортран-программ, каждый из которых взят из реальных программных систем, эксплуатируемых в Ливерморской национальной лаборатории им.Лоуренса (США). Обычно при проведении испытаний используется либо малый набор из 14 циклов, либо большой набор из 24 циклов.

Пакет Ливерморских циклов используется для оценки производительности вычислительных машин с середины 60-х годов. Ливерморские циклы считаются типичными фрагментами программ численных задач. Появление новых типов машин, в том числе векторных и параллельных, не уменьшило важности Ливерморских циклов, однако изменились значения производительности и величины разброса между разными циклами.

**Тесты TPC**

TPC определяет и управляет форматом нескольких тестов для оценки производительности OLTP (On-LineTransactionProcessing), включая тесты TPC-A, TPC-B и TPC-C. TPC требует только, чтобы при создании оценочного теста выполнялись определенные условия. Хотя упомянутые тесты TPC не являются характерными тестами для оценки производительности баз данных, системы реляционных баз данных являются ключевыми компонентами любой системы обработки транзакций.

Следует отметить, что как и любой другой тест, ни один тест TPC не может измерить производительность системы, которая применима для всех возможных сред обработки транзакций, но эти тесты действительно могут помочь пользователю справедливо сравнивать похожие системы. Однако, когда пользователь делает покупку или планирует решение о покупке, он должен понимать, что никакой тест не может заменить его конкретную прикладную задачу.

**AIM**

Одной из независимых организаций, осуществляющей оценку производительности вычислительных систем, является частная компания AIM Technology, которая была основана в 1981 году. Компания разрабатывает и поставляет программное обеспечение для измерения производительности систем, а также оказывает услуги по тестированию систем конечным пользователям и поставщикам вычислительных систем и сетей, которые используют промышленные стандартные операционные системы, такие как UNIX и OS/2.

За время своего существования компания разработала специальное программное обеспечение, позволяющее легко создавать различные рабочие нагрузки, соответствующие уровню тестируемой системы и требованиям по ее использованию. Это программное обеспечение состоит из двух основных частей: генератора тестовых пакетов (BenchmarkGenerator) и нагрузочных смесей (LoadMixes) прикладных задач.

Генератор тестовых пакетов представляет собой программную систему, которая обеспечивает одновременное выполнение множества программ. Он содержит большое число отдельных тестов, которые потребляют определенные ресурсы системы, и тем самым акцентируют внимание на определенных компонентах, из которых складывается ее общая производительность. При каждом запуске генератора могут выполняться любые отдельные или все доступные тесты в любом порядке и при любом количестве проходов, позволяя тем самым создавать для системы практически любую необходимую рабочую нагрузку. Все это дает возможность тестовому пакету моделировать любой тип смеси при постоянной смене акцентов (для лучшего представления реальной окружающей обстановки) и при обеспечении высокой степени конфигурирования.

Каждая нагрузочная смесь представляют собой формулу, которая определяет компоненты требуемой нагрузки. Эта формула задается в терминах количества различных доступных тестов, которые должны выполняться одновременно для моделирования рабочей нагрузки.

Используя эти две части программного обеспечения AIM, можно действительно создать для тестируемой системы любую рабочую нагрузку, определяя компоненты нагрузки в терминах тестов, которые должны выполняться генератором тестовых пакетов. Если некоторые требуемые тесты отсутствуют в составе генератора тестовых пакетов, то они могут быть легко туда добавлены.

Генератор тестовых пакетов во время своей работы “масштабирует” или увеличивает нагрузку на систему. Первоначально он выполняет и хронометрирует одну копию нагрузочной смеси. Затем одновременно выполняет и хронометрирует три копии нагрузочной смеси и т.д. По мере увеличения нагрузки, на основе оценки производительности системы, выбираются различные уровни увеличения нагрузки. В конце концов может быть нарисована кривая пропускной способности, показывающая возможности системы по обработке нагрузочной смеси в зависимости от числа моделируемых нагрузок. Это позволяет с достаточной достоверностью дать заключение о возможностях работы системы при данной нагрузке или при изменении нагрузки.

Очевидно, что сам по себе процесс моделирования рабочей нагрузки мало что дал бы для сравнения различных машин между собой при отсутствии у AIM набора хорошо подобранных смесей, которые представляют собой ряд важных для пользователя прикладных задач.

Все смеси AIM могут быть разделены на две категории: стандартные и заказные. Заказные смеси создаются для точного моделирования особенностей среды конечного пользователя или поставщика оборудования. Заказная смесь может быть тесно связана с определенными тестами, добавляемыми к генератору тестовых пакетов. В качестве альтернативы заказная смесь может быть связана с очень специфическим приложением, которое создает для системы необычную нагрузку. В общем случае заказные смеси разрабатываются на основе одной из стандартных смесей AIM путем ее “подгонки” для более точного представления определенной ситуации. Обычно заказные смеси разрабатываются заказчиком совместно с AIM Technology, что позволяет использовать многолетний опыт AIM по созданию и моделированию нагрузочных смесей.

**ЗАДАНИЯ ДЛЯ ВЫПОЛНЕНИЯ**

Запускаем SiSoftwareSandra. Переходим на вкладку «Эталонные тесты». Запускаем арифметический тест. Перед началом тестирования нажаем кнопку «Обновить».

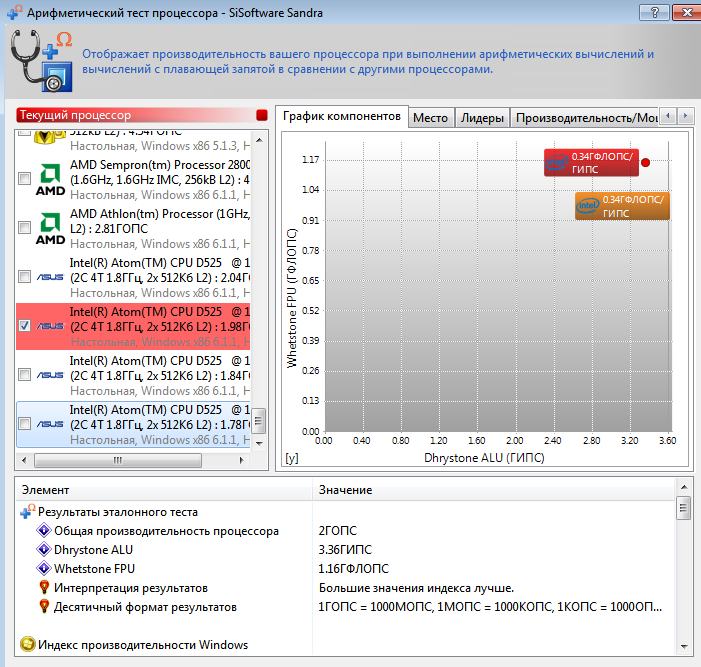
Данный тест (рис. 1) отображает производительность нашего процессора при выполнении арифметических вычислений и вычислений с плавающей запятой в сравнении с другими процессорами. Эталонные процессоры вы можете выбрать из списка.

Производительность измеряется в MIPS — (миллион целочисленных команд в секунду). В общем случае MIPS есть скорость операций с целыми числами в единицу времени, т.е. для любой данной программы MIPS есть просто отношение количества команд в программе к времени ее выполнения. Таким образом, производительность может быть определена как обратная к времени выполнения величина, причем более быстрые машины при этом будут иметь более высокий рейтинг MIPS.

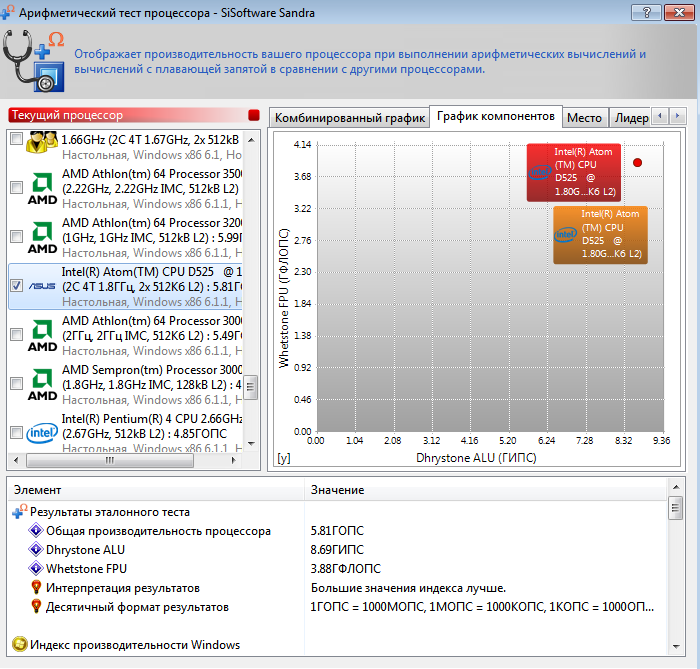
Обычно для научно-технических задач производительность процессора оценивается в MFLOPS (миллионах чисел-результатов вычислений с плавающей точкой в секунду, или миллионах элементарных арифметических операций над числами с плавающей точкой, выполненных в секунду). Он базируется на количестве выполняемых операций, а не на количестве выполняемых команд.

**Арифметический тест**

1. Со всеми отключенными модулями



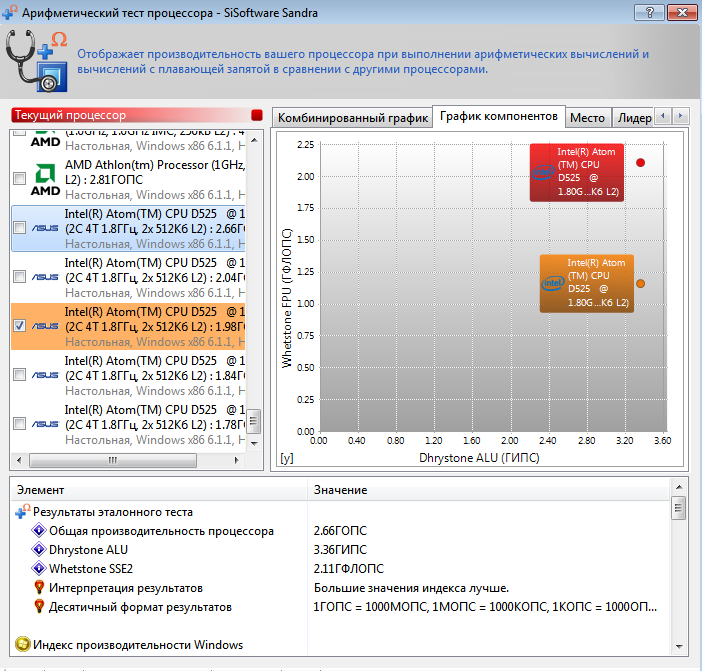
1. Многопоточный режим



3)**SSE2** набор инструкций, разработанный Intel и впервые представленный в процессорах серии Pentium 4. SSE2 расширяет набор инструкций SSE с целью полностью вытеснить MMX. Набор SSE2 добавил 144 новые команды к SSE, в котором было только 70 команд.

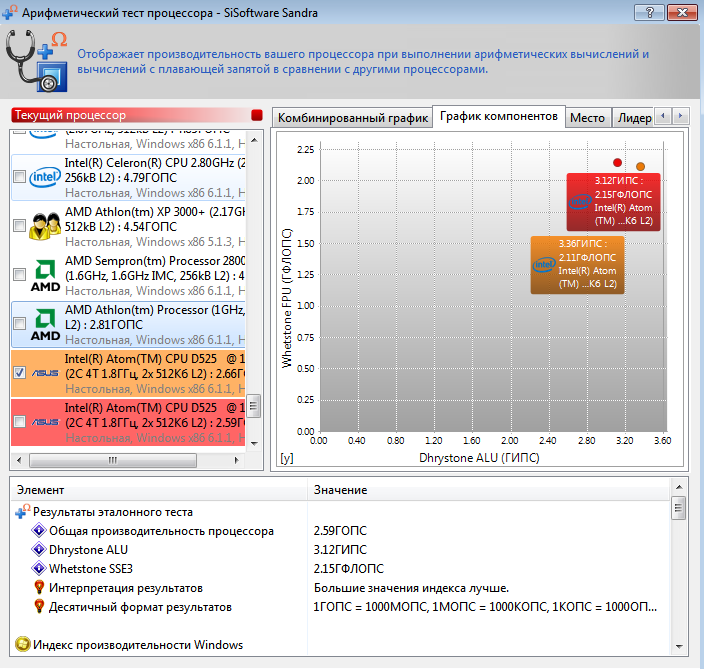
**Особенности**

* SSE2 использует восемь 128-битных регистров (xmm0 до xmm7), включённых в архитектуру x86 с вводом расширения SSE, каждый из которых трактуется как 2 последовательных значения с плавающей точкой двойной точности.
* SSE2 включает в себя набор инструкций, который производит операции со скалярными и упакованными типами данных.
* SSE2 содержит инструкции для потоковой обработки целочисленных данных в тех же 128-битных xmm регистрах, что делает это расширение более предпочтительным для целочисленных вычислений, нежели использование набора инструкций MMX, появившегося гораздо раньше.
* SSE2 включает в себя две части – продолжение SSE и продолжение MMX.SSE работает с вещественными числами.MMX работает с целыми.
* В SSE2 регистры по сравнению с MMX удвоились (64 бита -> 128 битов). Т.к. скорость выполнения инструкций не изменилась, при оптимизации под SSE2 программа получает двукратный прирост производительности. Если программа уже была оптимизирована под MMX, то оптимизация под SSE2 даётся сравнительно легко в силу сходности системы команд.
* SSE2 включает в себя ряд команд управления кэшем, предназначенных для минимизации загрязнения кэша при обработке неопределенных потоков информации.
* SSE2 включает в себя сложные дополнения к командам преобразования чисел



4)**SSE3** - третья версия SIMD-расширения [Intel](http://ru.wikipedia.org/wiki/Intel" \o "Intel), потомок [SSE](http://ru.wikipedia.org/wiki/SSE), [SSE2](http://ru.wikipedia.org/wiki/SSE2) и [MMX](http://ru.wikipedia.org/wiki/MMX). Набор SSE3 содержит 13 инструкций: FISTTP (x87), MOVSLDUP (SSE), MOVSHDUP (SSE), MOVDDUP (SSE2), LDDQU (SSE/SSE2), ADDSUBPD (SSE), ADDSUBPD (SSE2), HADDPS (SSE), HSUBPS (SSE), HADDPD (SSE2), HSUBPD (SSE2), MONITOR (нет аналога в SSE3 для [AMD](http://ru.wikipedia.org/wiki/AMD)), MWAIT (нет аналога в SSE3 для AMD).

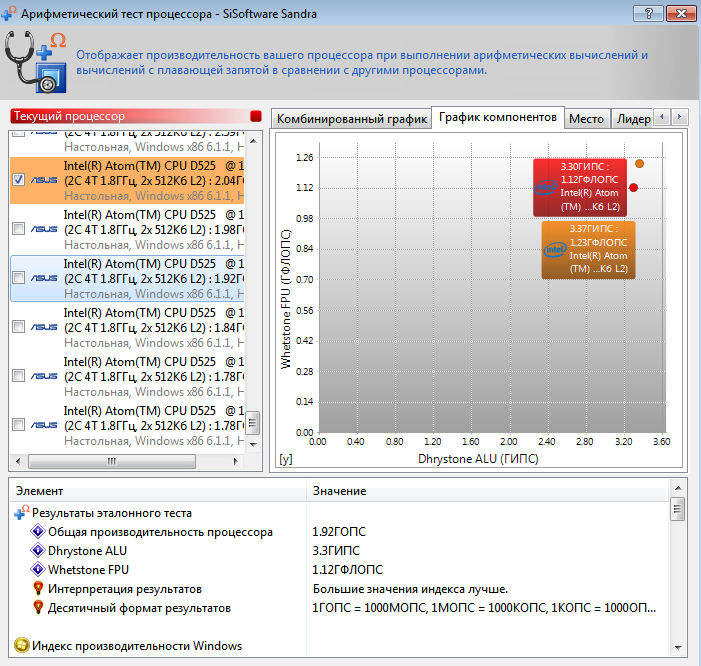
Наиболее заметное изменение - возможность горизонтальной работы с регистрами. Если говорить более конкретно, добавлены команды сложения и вычитания нескольких значений, хранящихся в одном регистре. Эти команды упростили ряд DSP и 3D-операций. Существует также новая команда для преобразования значений с плавающей точкой в целые без необходимости вносить изменения в глобальном режиме округления.



5) **SSE4** — новый набор команд микроархитектуры IntelCore.Он был анонсирован 27 сентября [2006 года](http://ru.wikipedia.org/wiki/2006_%D0%B3%D0%BE%D0%B4).

SSE4 состоит из 54 инструкций, 47 из них относят к SSE4.1. Полный набор команд (SSE4.1 и SSE4.2, то есть 47 + оставшиеся 7 команд) доступен только в процессорах Intel с микроархитектурой Nehalem, которые были выпущены в середине ноября 2008 года.

Добавленыинструкции, ускоряющие [компенсациюдвижения](http://software.intel.com/ru-ru/articles/motion-estimation-with-intel-streaming-simd-extensions-4-intel-sse4/) в [видеокодеках](http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D0%BA%D0%BE%D0%B4%D0%B5%D0%BA), быстрое [чтение из USWC](http://software.intel.com/ru-ru/articles/increasing-memory-throughput-with-intel-streaming-simd-extensions-4-intel-sse4-streaming-load/) памяти, множество инструкций для упрощения векторизации программ компиляторами. Кроме того, в SSE4.2 добавлены инструкции обработки строк 8/16 битных символов, вычисления [CRC32](http://ru.wikipedia.org/wiki/CRC32), [popcnt](http://ru.wikipedia.org/w/index.php?title=Popcnt&action=edit&redlink=1" \o "Popcnt (страница отсутствует)). Впервые в SSE4 регистр xmm0 стал использоваться как неявный аргумент для некоторых инструкций.

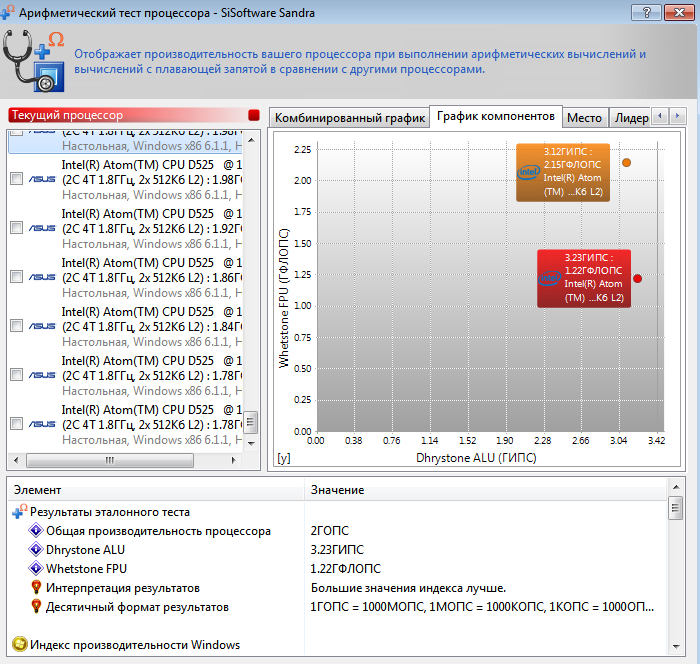


6)**AdvancedVectorExtensions** (AVX) — расширение системы команд x86 для [микропроцессоров](http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) [Intel](http://ru.wikipedia.org/wiki/Intel" \o "Intel) и [AMD](http://ru.wikipedia.org/wiki/AMD), предложенное Intel в марте 2008.

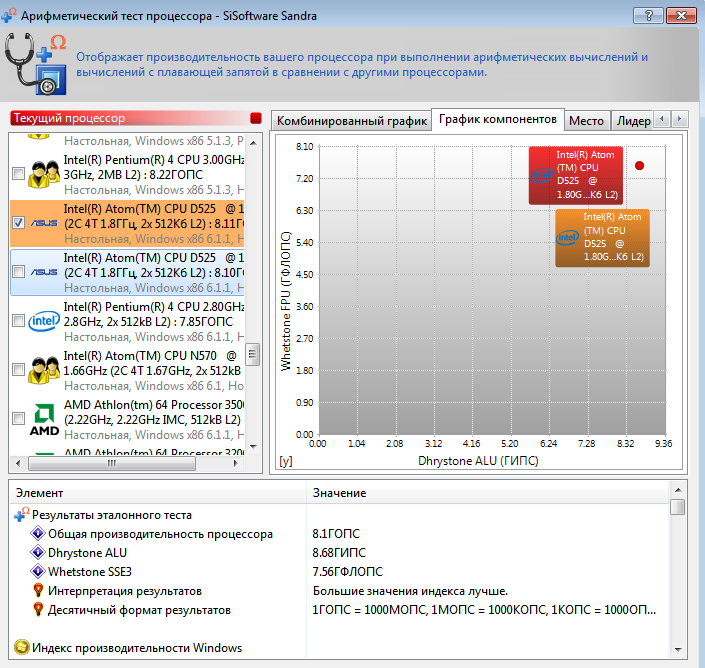
AVX предоставляет различные улучшения, новые инструкции и новую схему кодирования машинных кодов.

Размер векторных регистров [SIMD](http://ru.wikipedia.org/wiki/SIMD) увеличивается с 128 ([XMM](http://ru.wikipedia.org/w/index.php?title=XMM&action=edit&redlink=1)) до 256 бит (регистры YMM0 — YMM15). Существующие 128-битные инструкции будут использовать младшую половину новых YMM регистров. В будущем возможно расширение до 512 или 1024 бит.

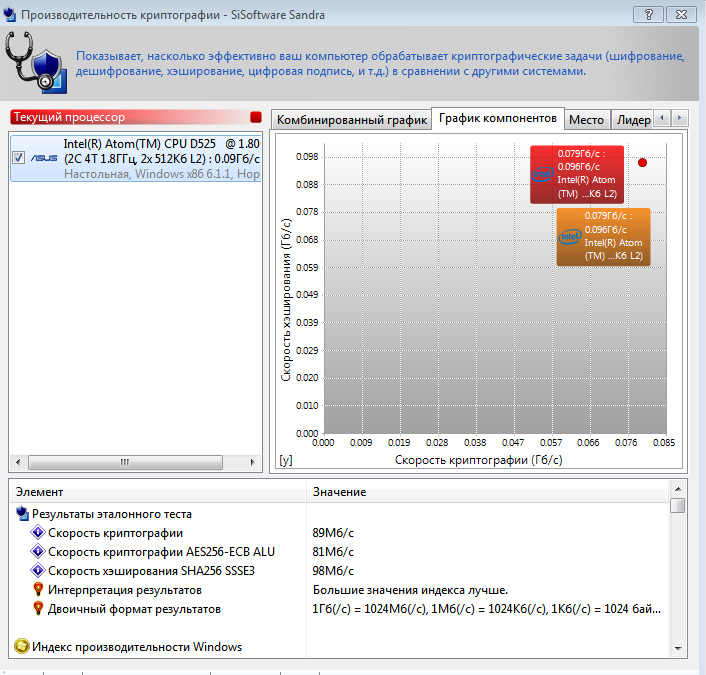
Набор инструкций AVX позволяет использовать любую двухоперандную инструкцию XMM в трёхоперандном виде без модификации двух регистров-источников, с отдельным регистром для результата. Например, вместо *a* = *a* + *b* можно использовать *c* = *a* + *b*, при этом регистр *a* остаётся неизменённым. AVX не поддерживает неразрушающие формы операций над обычными регистрами общего назначения, такими как EAX, но такая поддержка, возможно, будет добавлена в последующих расширениях.



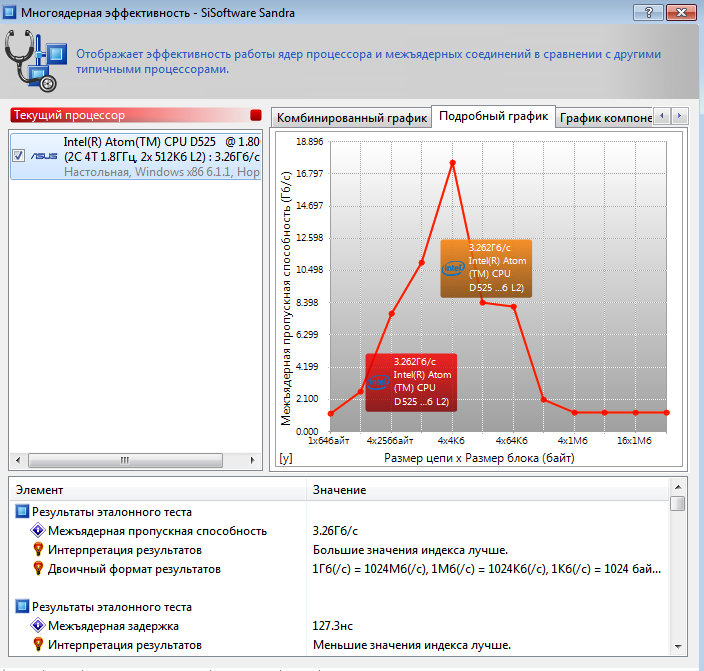
7) Включены все инструкции процессора (процессор нетбука работает на полную мощность).



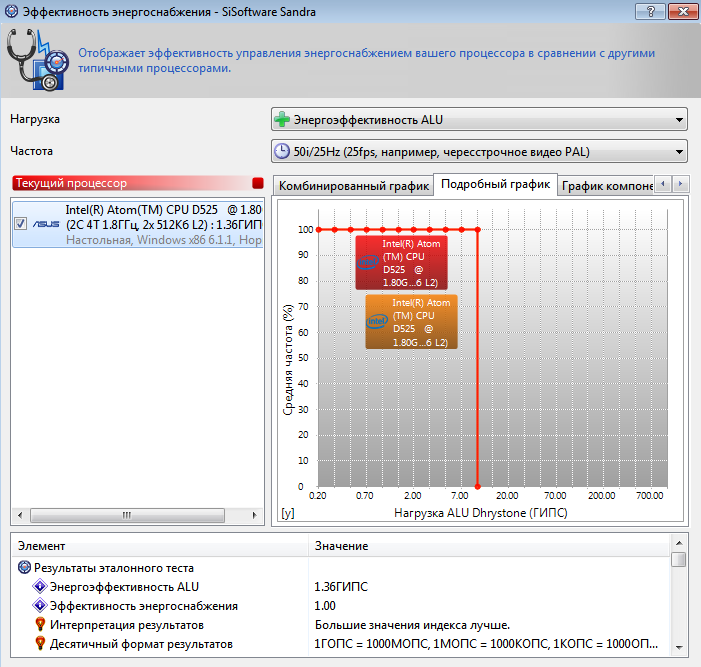
8) Шифрование и дешифрование



9) Эффективность работы многоядерного процессора



10) Эффективность энергоснабжения



В данном процессоре отсутствует динамическая система распределения нагрузки. Поэтому в тесте на энергопотребление отображается прямая линия. Совсем иной результат был бы на серии процессоров IntelI(i3, i5, i7), результатом теста для данных процессоров является кривая, которая отображает эффективность производительности от энергоэффективности.

**Вывод:** В ходе лабораторной работе был ознакомлен с основными принципами тестирования центрального процессора; проверил наличие различных инструкций процессора, сравнил результаты их работы.