Учреждение образования

«Белорусский государственный технологический университет»

Кафедра полиграфических производств

Отчет по лабораторной работе №5

«ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ»

Выполнила студентка   
Буранко Валерия Дмитриевна   
ФИТ 2 курс 5 группа

Минск 2021

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

**Теоретическая часть**

**Факторы, влияющие на производительность компьютера:**

*Конвейерные прерывания* - Один из методов увеличения производительности компьютера - выполнение нескольких команд одновременно. Это достигается при помощи конвейерной технологии: несколько последовательных команд находятся на разных стадиях выполнения - от декодирования до запоминания результата. В этом случае при передаче управления происходит отбрасывание частично выполненных команд, оставшихся на конвейере. Такие прерывания существенно ограничивают производительность.

*Быстродействие ОЗУ* - Если процессор по своей производительности значительно превосходит оперативную память системы, то при каждом обращении к ней процессор несколько циклов отрабатывает "вхолостую". В случае, когда 32-разрядный процессор работает с 16-разрядным ОЗУ, процессор должен ожидать завершения обработки двух последовательных запросов.

*Исчезновение циклов памяти* - В компьютерах, где системное ОЗУ используется центральным процессором и видеоадаптером совместно, то, поскольку, эти обращения не могут происходить одновременно, в самом простом случае, процессору и видеоадаптеру присваиваются альтернативные циклы доступа к памяти. По сути дела видеоадаптер как бы ворует циклы процессора по мере необходимости.

*Низкая эффективность программного обеспечения* - Программное обеспечение не может само по себе научиться использовать преимущества новейших достижений в сфере увеличения производительности аппаратуры. Если операционные системы обычно проверяют наличие сопроцессора или расширенной системы команд, то прикладные пакеты, как правило, такой проверки не производят, и, соответственно, все дополнительные возможности аппаратуры просто игнорируются.

*Быстродействие дисковой памяти* - Производительность дисковой подсистемы очень сильно влияет на общую производительность компьютера.

*Фактор чередования* - Обычно секторы на дорожке располагаются в порядке возрастания и порядковых номеров. Процедура чтения заключается в том, что контроллер диска устанавливает головки на нужную дорожку и начинает сканировать подряд все секторы для того чтобы найти требуемый сектор. Контроллер при поиске пользуется номером сектора, записанным в области служебной информации. После того как головка окажется над искомым сектором, начинается процесс считывания данных (512 байт) и записи их в оперативную память, как только все данные записаны в память, компьютер выдает контроллеру команду чтения следующего сектора.

*Кэширование дисковой памяти* - Во время работы операционная система и прикладные программы часто обращаются к одним и тем же файлам или к одним и тем же областям диска. Чтобы избежать повторения операции чтения диска при обращении к наиболее часто используемым программам применяется кэширование дисковой памяти: в области оперативной памяти выделяется некоторое пространство для хранения содержимого секторов диска – кэш буфер.

*Электронный диск -* Если в вашем компьютере имеется расширенная или дополнительная память, вы можете организовать так называемый электронный диск.

*Буферизация*- Еще один путь к увеличению быстродействия дисковой подсистемы лежит в использовании буферизации ввода/вывода.

**Тесты для оценки производительности, их отличие**

**1. Стресс-тестирование (Stress Test)**

Этот тест проводится первым. Нагрузка постепенно увеличивается до тех пор, пока приложение не перестанет работать корректно. В конце теста фиксируется количество пользователей, которое приложение выдерживало, соответствуя требованиям производительности, и сколько выдержать не смогло. Первое значение и будет пределом производительности вашего приложения. Часто этот вид тестирования проводится, если заказчик предвидит [резкое увеличение нагрузки на систему](https://www.a1qa.ru/blog/sajt-gotov-k-chernoj-pyatnitse-kiberponedelniku/). Например, для e-commerce это могут быть дни распродаж.

**2. Нагрузочный тест (Load Test)**

Нагрузка на систему подается на протяжении 4-8 часов. В это время собираются метрики производительности: количество запросов в секунду, транзакций в секунду, время отклика от сервера, процент ошибок в ответах, утилизация аппаратных ресурсов и т д. Собранные метрики проходят проверку на соответствие заданным требованиям. В результате получаем ответ на вопрос: соответствует ли система требованиям производительности?

Также на выходе имеем локализацию узких мест в производительности приложения и дефектов, подробное профилирование всех компонентов системы и утилизацию аппаратных ресурсов под целевой нагрузкой.

**3. Тестирование на больших объемах данных (Volume Test)**

Данный вид тестирования помогает сделать прогноз относительно работоспособности приложения. Форма подаваемой нагрузки та же, что и при нагрузочном тестировании. Задача теста – узнать, какое влияние окажет увеличение объема данных на систему. Таким образом, можем найти ответ на вопрос: как изменится производительность приложения спустя X лет, если аудитория приложения вырастет в Y раз?

**4. Тестирование отказоустойчивости (Stability Test)**

Продолжительность нагрузки может варьироваться в зависимости от целей и возможностей проекта, доходя до семи дней и более. В результате получаем представление о том, как изменится производительность системы в течение длительного периода времени под нагрузкой, например, в течение недели. Снизится ли уровень производительности? Способно ли приложение выдерживать стабильную нагрузку без критических сбоев?

**5. Тестирование масштабируемости (Scalability Test)**

Профиль нагрузки тот же, что и при нагрузочном тестировании. Что получаем в результате? Ответы на следующие вопросы:

* Увеличится ли производительность приложения, если добавить дополнительные аппаратные ресурсы?
* Увеличится ли производительность пропорционально количеству добавленных аппаратных средств?

О видах тестирования поговорили, теперь коснемся такого важного аспекта их проведения как нагрузка.

**Производительность компьютера прямо пропорциональна тактовой частоте его процессора.** Чем выше тактовая частота, тем выше производительность компьютера.

**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.

Другое определение MIPS связано с очень популярным когда-то компьютером VAX 11/780 компании DEC. Именно этот компьютер был принят в качестве эталона для сравнения производительности различных машин. Считалось, что производительность VAX 11/780 равна 1 MIPS (одному миллиону команд в секунду).

В то время широкое распространение получил синтетический тест Dhrystone, который позволял оценивать эффективность процессоров и компиляторов с языка C для программ нечисловой обработки. Он представлял собой тестовую смесь, 53% которой составляли операторы присваивания, 32% — операторы управления и 15% — вызовы функций. Это был очень короткий тест: общее число команд равнялось 100. Скорость выполнения программы из этих 100 команд измерялась в Dhrystone в секунду. Быстродействие VAX 11/780 на этом синтетическом тесте составляло 1757 Dhrystone в секунду. Таким образом 1 MIPS равен 1757 Dhrystone в секунду.

Следует отметить, что в настоящее время тест Dhrystone практически не применяется. Малый объем позволяет разместить все команды теста в кэш-памяти первого уровня современного микропроцессора и он не позволяет даже оценить эффект наличия кэш-памяти второго уровня, хотя может хорошо отражать эффект увеличения тактовой частоты.

Третье определение MIPS связано с IBM RS/6000 MIPS. Дело в том, что ряд производителей и пользователей (последователей фирмы IBM) предпочитают сравнивать производительность своих компьютеров с производительностью современных компьютеров IBM, а не со старой машиной компании DEC. Соотношение между VAX MIPS и RS/6000 MIPS никогда широко не публиковались, но 1 RS/6000 MIPS примерно равен 1.6 VAX 11/780 MIPS.

**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.

Ясно, что 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.

Однако и с MFLOPS не все обстоит так безоблачно. Прежде всего, это связано с тем, что наборы операций с плавающей точкой не совместимы на различных компьютерах. Например, в суперкомпьютерах фирмы Cray Research отсутствует команда деления (имеется, правда, операция вычисления обратной величины числа с плавающей точкой, а операция деления может быть реализована с помощью умножения делимого на обратную величину делителя). В то же время многие современные микропроцессоры имеют команды деления, вычисления квадратного корня, синуса и косинуса.

Другая, осознаваемая всеми, проблема заключается в том, что рейтинг MFLOPS меняется не только на смеси целочисленных операций и операций с плавающей точкой, но и на смеси быстрых и медленных операций с плавающей точкой. Например, программа со 100% операций сложения будет иметь более высокий рейтинг, чем программа со 100% операций деления.

Решение обеих проблем заключается в том, чтобы взять “каноническое” или “нормализованное” число операций с плавающей точкой из исходного текста программы и затем поделить его на время выполнения. Наиболее часто MFLOPS, как единица измерения производительности, используется при проведении контрольных испытаний на тестовых пакетах “Ливерморские циклы” и LINPACK.

**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.

На векторной машине производительность зависит не только от элементной базы, но и от характера самого алгоритма, т.е. коэффициента векторизуемости. Среди Ливерморских циклов коэффициент векторизуемости колеблется от 0 до 100%, что еще раз подтверждает их ценность для оценки производительности векторных архитектур. Кроме характера алгоритма, на коэффициент векторизуемости влияет и качество векторизатора, встроенного в компилятор.

На параллельной машине производительность существенно зависит от соответствия между структурой аппаратных связей вычислительных элементов и структурой вычислений в алгоритме. Важно, чтобы тестовый пакет представлял алгоритмы различных структур. В Ливерморских циклах встречаются последовательные, сеточные, конвейерные, волновые вычислительные алгоритмы, что подтверждает их пригодность и для параллельных машин. Однако обобщение результатов измерения производительности, полученных для одной параллельной машины, на другие параллельные машины или хотя бы некоторый подкласс параллельных машин, может дать неверный результат, ибо структуры аппаратных связей в таких машинах гораздо более разнообразны, чем, скажем, в векторных машинах.

**LINPACK** — это пакет фортран-программ для решения систем линейных алгебраических уравнений. Целью создания LINPACK отнюдь не было измерение производительности. Алгоритмы линейной алгебры весьма широко используются в самых разных задачах, и поэтому измерение производительности на LINPACK представляют интерес для многих пользователей. Сведения о производительности различных машин на пакете LINPACK публикуются сотрудником Аргоннской национальной лаборатории (США) Дж. Донгаррой и периодически обновляются.

В основе алгоритмов действующего варианта LINPACK лежит метод декомпозиции. Исходная матрица размером 100х100 элементов (в последнем варианте размером 1000х1000) сначала представляется в виде произведения двух матриц стандартной структуры, над которыми затем выполняется собственно алгоритм нахождения решения. Подпрограммы, входящие в LINPACK, структурированы. В стандартном варианте LINPACK выделен внутренний уровень базовых подпрограмм, каждая из которых выполняет элементарную операцию над векторами. Набор базовых подпрограмм называется BLAS (Basic Linear Algebra Subprograms). Например, в BLAS входят две простые подпрограммы SAXPY (умножение вектора на скаляр и сложение векторов) и SDOT (скалярное произведение векторов). Все операции выполняются над числами с плавающей точкой, представленными с двойной точностью. Результат измеряется в MFLOPS.

Использование результатов работы тестового пакета LINPACK с двойной точностью как основы для демонстрации рейтинга MFLOPS стало общепринятой практикой в компьютерной промышленности. При этом следует помнить, что при использовании исходной матрицы размером 100х100, она полностью может размещаться в кэш-памяти емкостью, например, 1 Мбайт. Если при проведении испытаний используется матрица размером 1000х1000, то емкости такого кэша уже недостаточно и некоторые обращения к памяти будут ускоряться благодаря наличию такого кэша, другие же будут приводить к промахам и потребуют большего времени на обработку обращений к памяти. Для многопроцессорных систем также имеются параллельные версии LINPACK и такие системы часто показывают линейное увеличение производительности с ростом числа процессоров.

Однако, как и любая другая единица измерения, рейтинг MFLOPS для отдельной программы не может быть обобщен на все случаи жизни, чтобы представлять единственную единицу измерения производительности компьютера, хотя очень соблазнительно характеризовать машину единственным рейтингом MIPS или MFLOPS без указания программы.

Независимая **тестовая** **организация** — **организация**, которая проводит тестирование продуктов, материалов, **программ** и др. на соответствие принятым требованиям. **Организация** может быть полностью независимой (независимая **тестовая** лаборатория), либо подчиненной государству или научной **организации**.

**Transaction Processing Performance Council, TPC** (Совет по обработке транзакций, http://www.tpc.org) - независимая некоммерческая организация, созданная для исследования обработки транзакций и производительности СУБД и распространения объективной и воспроизводимой информации о производительности в тестах**TPC** для компьютерной индустрии.

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.

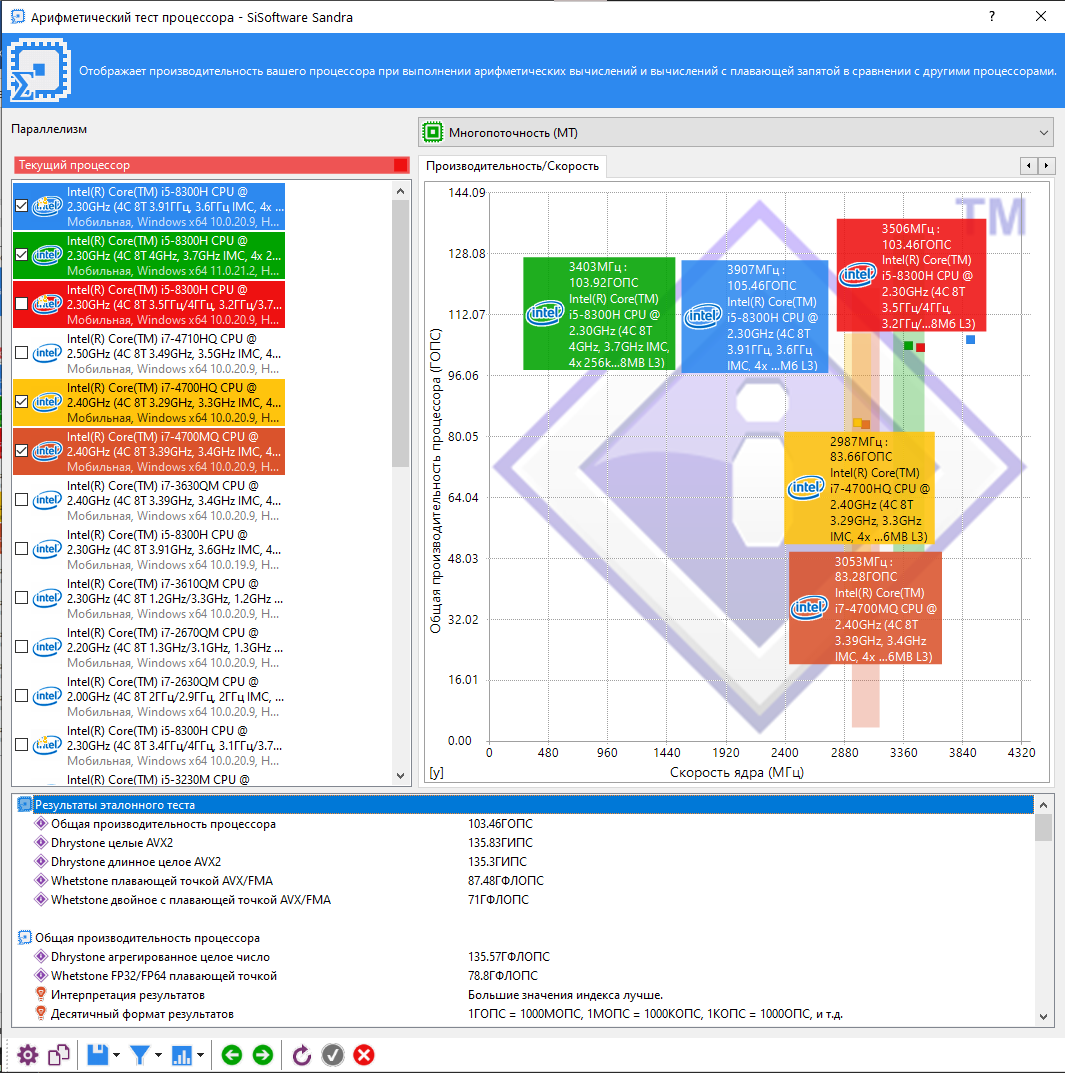
За время своего существования компания разработала специальное программное обеспечение, позволяющее легко создавать различные рабочие нагрузки, соответствующие уровню тестируемой системы и требованиям по ее использованию. Это программное обеспечение состоит из двух основных частей: генератора тестовых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

**Whetstone**.Тест спроектирован для оценки характеристик систем при интенсивном использовании операций с плавающей запятой. Это смесь плавающих и целочисленных операций, вычисления трансцедентных функций, обработки массивов и вызовов процедур.

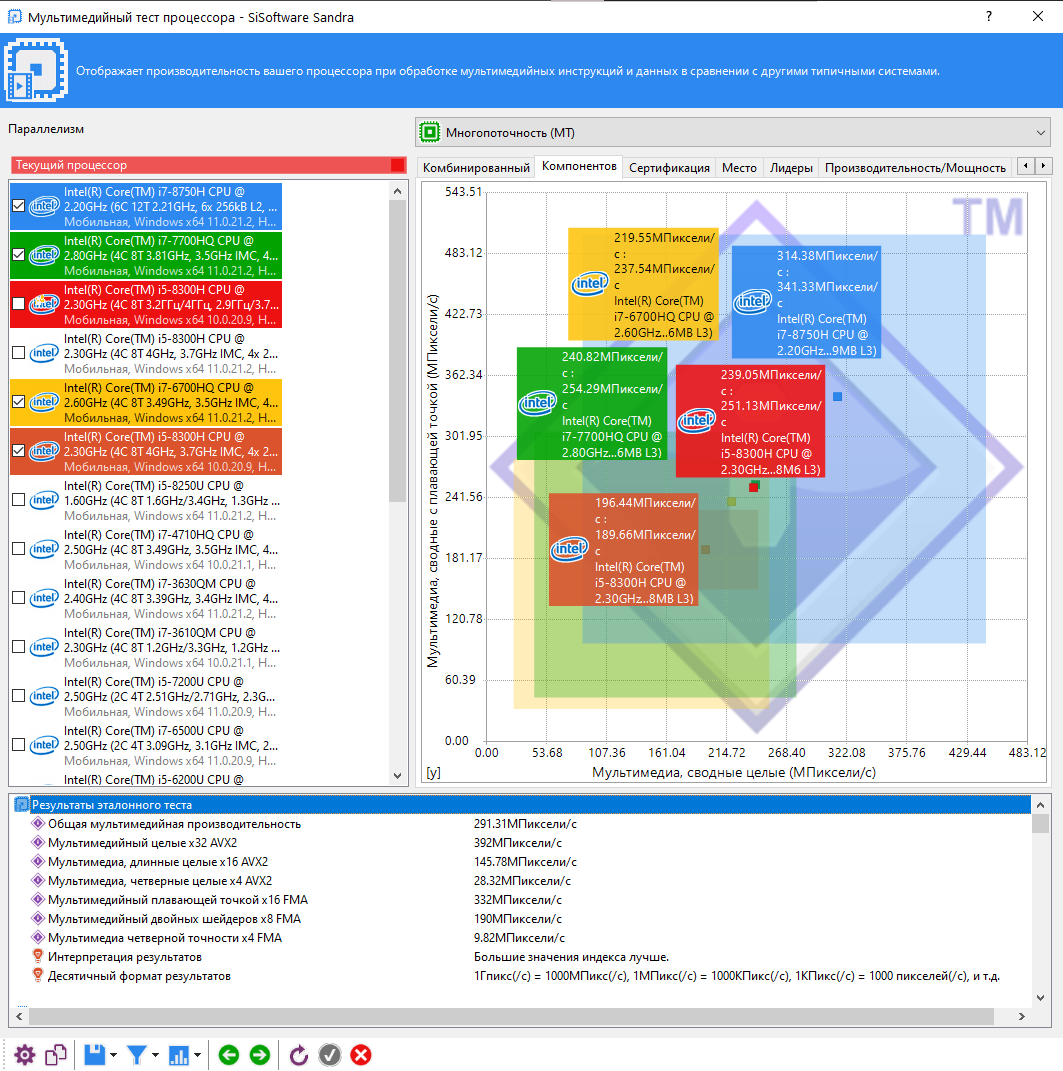
**Dhrystone** — синтетический тест производительности компьютеров, разработанный в 1984 году [Reinhold P. Weicker](https://ru.wikipedia.org/w/index.php?title=Reinhold_P._Weicker&action=edit&redlink=1). Нацелен на тестирование системной (целочисленной) производительности процессоров общего назначения.

**Практическая часть**

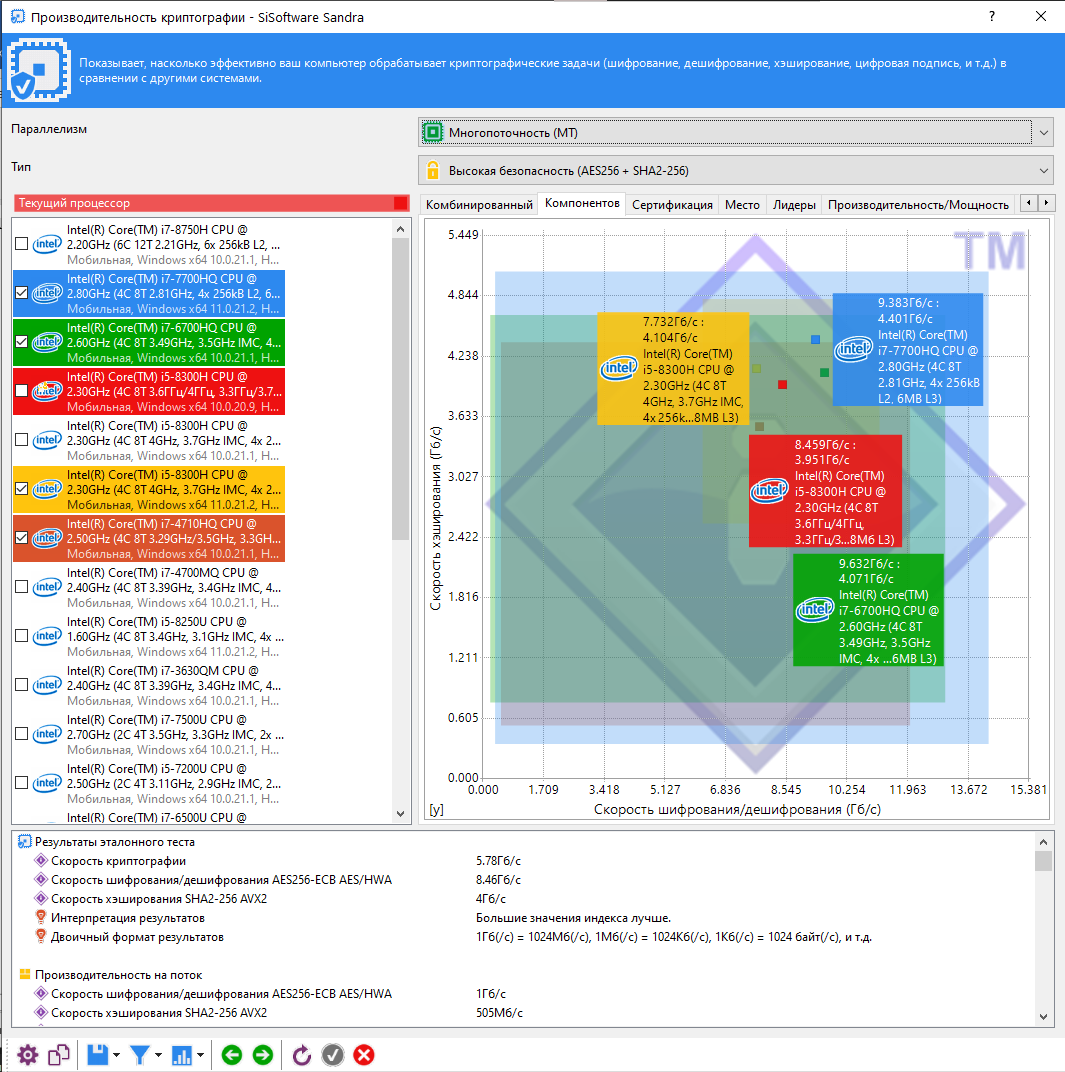
Арифметический тест



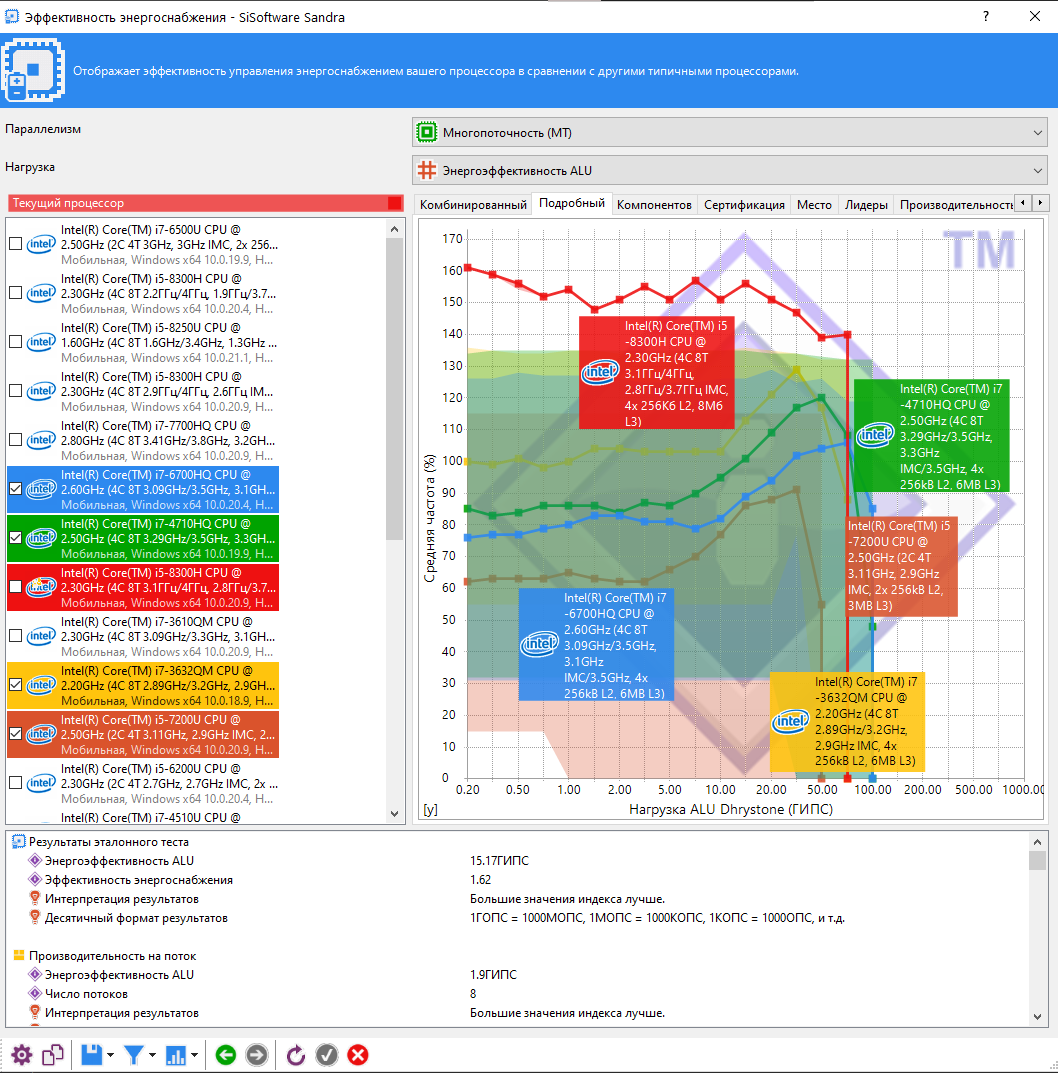
Мультимедийный тест



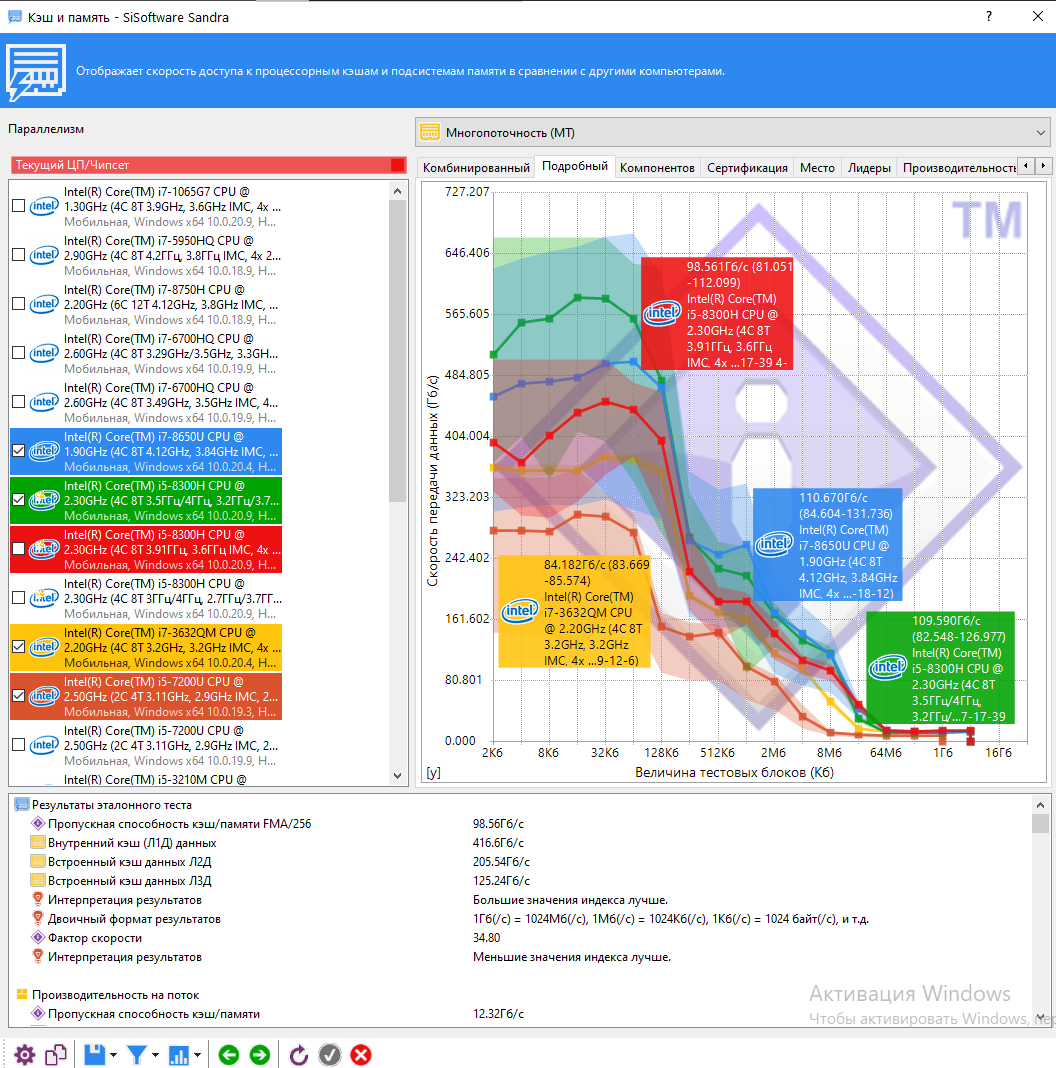
Производительность криптографии



Эффективность энергоснабжения



Кэш и память



Информация о системе

