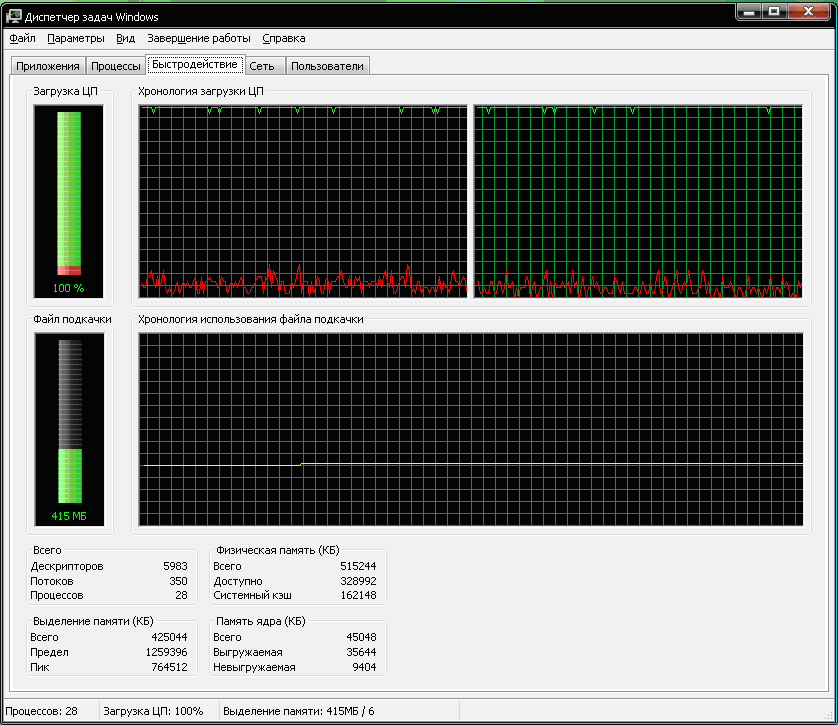
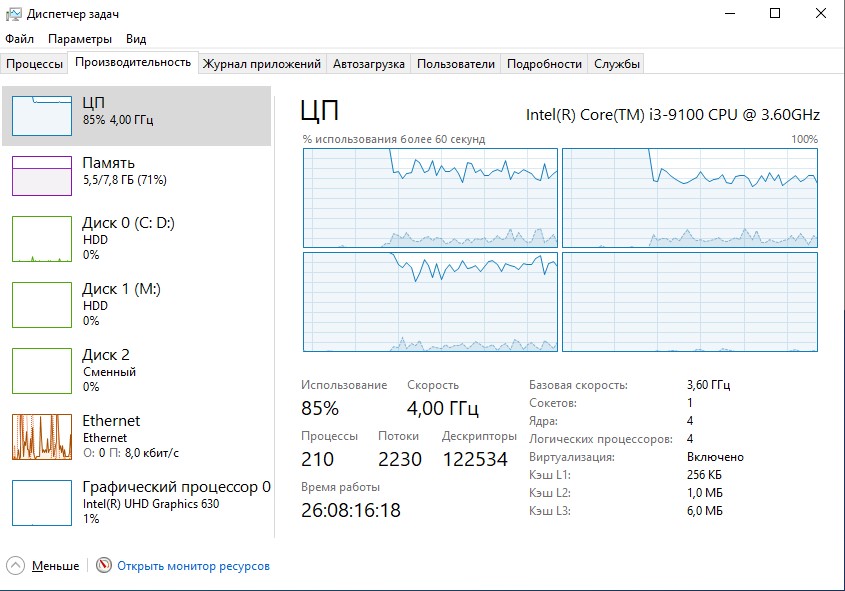
**Описание функциональных характеристик «Монитора реального быстродействия и профилировщика процессоров серии X86\_64 Speedprograf»**

Монитор реального быстродействия и профилировщик процессоров серии X86\_64 Speedprograf предназначен для невозмущающего контроля скорости выполнения операций процессоров серии X86\_64 Intel и AMD, как в режиме ядра (Ring 0 – Kernel Mode), так и в пользовательском режиме, на других уровнях привилегий (Ring 1,2,3 - User Mode). Speedprograf профилирует всю систему целиком (операционную систему, гипервизор, приложения) и накапливает информацию на MSR (Model-Specific Registers) регистрах процессора.

Существующие мониторы производительности под названием быстродействие



и производительность



показывают обыкновенную загрузку процессора (% времени работы на временном промежутке).



Аналога тахометра и спидометра в современных мониторах производительности нет.

Speedprograf исправляет эту несправедливость и показывает и загрузку процессора и реальную скорость исполнения кода (спидометр процессора в % пикового быстродействия).

Если в User Mode код исполняется со скоростью 5-10%, то в Kernel Mode 1-2%!

В США существовала программа развития суперкомпьютеров DARPA HPCS (2001-2010 годы) в которой ставилась задача увеличения производительности суперкомпьютеров на реальных приложениях в 10-40 раз (не за счет увеличения количества ядер, а за счет реализации их пикового быстродействия). Задача увеличения реальной производительности не была выполнена, поскольку уперлась в «стену памяти» и перешла для решения в очередную программу США по суперкомпьютерам UHPCS (2011-2020 годы). Но если в 10-40 раз можно увеличить реальную производительность для суперкомпьютеров, то их реальный текущий КПД по быстродействию 2,5-10%. На сегодняшний день эта задача так и не решена.

Вы бы стали ездить на машине, у которой в паспорте написана максимальная скорость 250 км/час, а она ездит не быстрее 5 км/час? Наверно нет!

Но почему-то все мирятся с такой ситуацией в компьютерных вычислениях?!

По мнению автора существует проблема исполнения кода в программно-аппаратном комплексе, независимо от того где он установлен: в суперкомпьютере, в центре обработке данных, в ПК, в гаджете, в станке.

Проблема исполнения кода делится на 2 части:

1. Низкая скорость исполнения кода, ввиду того, что он однопроходной (не повторяемый или слабоповторяемый).
2. Большие накладные расходы при исполнении кода, которые существенно превышают долю кода, который исполняет целевую функцию программно-аппаратного комплекса (Эту тему автор впервые поднял в материалах международной конференции «Наука и будущее: идеи, которые изменят мир»,14-16 апреля 2004 года, Москва, ГГМ им. В.И.Вернадского РАН, тезисы доклада Суворова Д.А. «Роль организации вычислений в реализации потенциала современных компьютеров»).

Продуктивность программного кода – это доля кода, которая служит выполнению целевой функции, исполняемой программно-аппаратным комплексом. Продуктивность программного кода вычислительных систем 2 – поколения, написанных в машинных кодах составляла 33%(1/3), продуктивность программного кода вычислительных систем 3-го поколения (IBM 360/370, ЕС ЭВМ ряд 1/2), написанных на языках высокого уровня (ALGOL68, FORTRAN, PL1) со стековой организацией памяти вызовов составляла 25%(1/4), продуктивность современного кода систем программирования с ООП (объектно-ориентированное программирование) и стековой организацией памяти составляет 10%(1/10), продуктивность кода ядра современной операционной системы составляет 1%(1/100). Снижение продуктивности приводит к увеличению длины кода, при выполнении одной и той же работы!

Автор предлагает способ повышения производительности, который увеличивает скорость исполнения кода и повышает продуктивность на этапе исполнения.

Практически для кода с продуктивностью 1/3-1/100 можно сократить длину(трассу) с 2,5 до 9 раз при 10-кратном повторении целевого кода, но практически ощутимый прирост эффективности будет происходить только в первых пяти повторениях или от 2 до 5 раз.

Speedprograf специально разработан для объективной оценки скорости исполнения кода и его длины (сколько команд истрачено на выполнение той или иной работы).

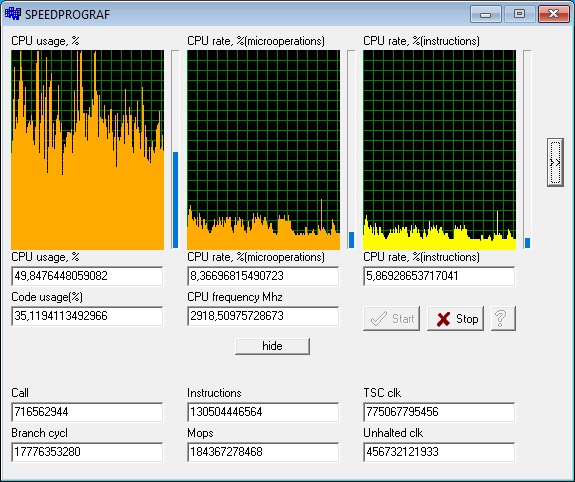
Перед началом профилирования Speedprograf программирует MSR регистры на измерение необходимых характеристик через драйвер msr.sys, который выполняет в режиме ядра привилегированные команды, после окончания профилирования (или периодически) в течение работы Speedprograf периодически считывает значения запрограммированных регистров, так же через драйвер msr.sys.

В наиболее полнофункциональной версии (для процессоров Intel Pentium IV – это наиболее полный профилировщик для процессоров Х86\_64 серии, а не устаревший, который одновременно может регистрировать до 16 программируемых параметров + TSC) Speedprograf выводит следующие параметры:

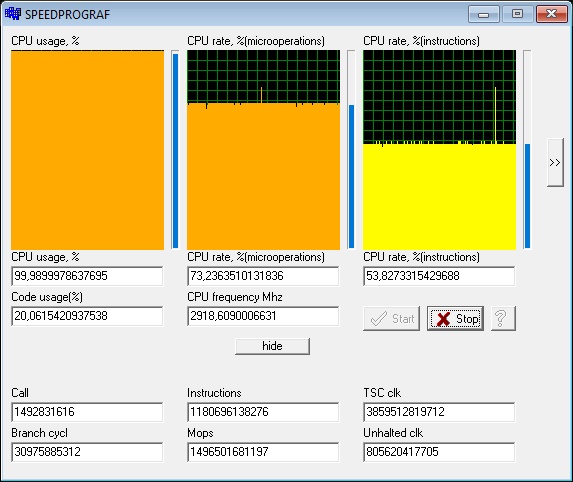
1. Процент реализации пикового быстродействия в инструкциях (CPU rate instruction).
2. Загрузку процессора (CPU usage).
3. Частоту работы процессора (CPU frequency Mhz).
4. Количество исполненных инструкций (Instructions).
5. Количество тактов процессора, прошедших с начала профилирования (TSС clk).
6. Количество тактов процессора, когда он не простаивал (Unhalted clk).
7. Количество вызовов (команд Call).
8. Количество переходов (Branch cycl).
9. Долю команд процессора, обращающихся к данным (Code usage).
10. Процент реализации пикового быстродействия в микрооперациях (CPU rate microoperations).
11. Количество исполненных микроопераций (Mops).

Версия профилировщика для AMD поддерживает до 8 первых параметров (из 11) в предыдущем списке (в AMD 4 программируемых регистра + TSC), а версия профилировщика для Core XX поддерживает только 4 первых параметра (в Core XX 2 программируемых регистра + TSC).

Так выглядит скорость архивации WinRar\_ом

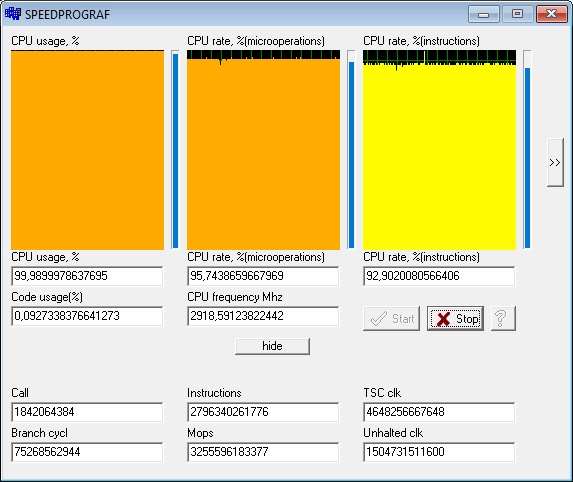


Скорость на уровне 6% от пиковой, при этом доля команд, обрабатывающих данные Code usage составляет 35,1%.

.

А это скорость теста, к которой мы должны стремиться после доработок операционных систем! Здесь пиковое быстродействие 54%, а Code usage = 20%.

Скорость в 9 раз больше, чем на предыдущем рисунке. Есть к чему стремиться!



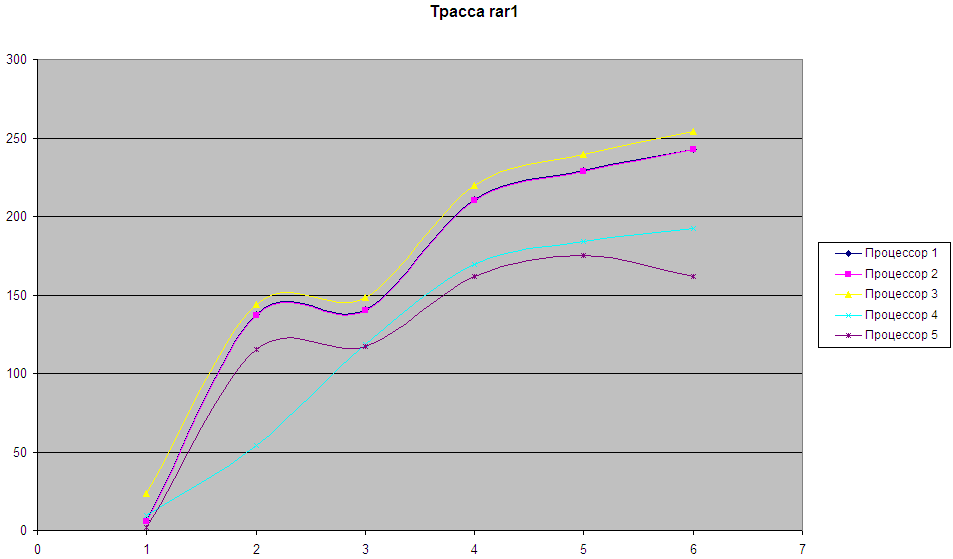
На этом рисунке вообще 92.9% пикового быстродействия!! Если поделить на 6% будет 15 с лишним раз раз!

Однако здесь используется нечестный прием тестировщиков. Тест построен на пустых операциях (NOP)! Code usage = 0,09%. Процессор не обрабатывает данные! При обработке данных он не сможет работать с такой скоростью (это только на тесте).

Полнофункциональный профилировщик позволяет объективно оценивать реальное быстродействие!

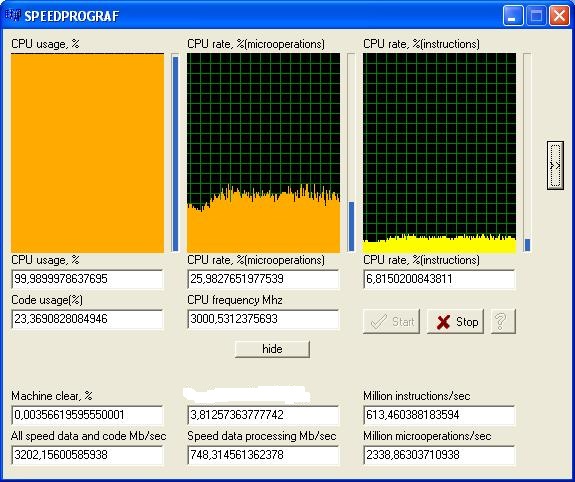
Привожу пример исследования SPEEDPROGAF\_ом однонитевого приложения на 5-ти x86\_64 процессорах. Это архивация WinRar-ом  дистрибутива  Microsoft Office размером 0,55 Гбайт. Состав команд, используемых процессором – целочисленная арифметика. Архивные файлы представляют собой файлы длиной 50-60% от исходного объема дистрибутива. Степень архивации обозначена цифрами; 1 - архив без сжатия, 2 – скоростная, 3 -быстрая, 4 – обычная, 5 – хорошая, 6 – максимальная. Результирующие архивы от работы разных процессоров совпадают по длине до байта, несмотря на различную длину исполненного кода! Что происходит?

Приведем длину трассы (в миллиардах инструкций, х86, исполненных по программе – Retired Instruction)

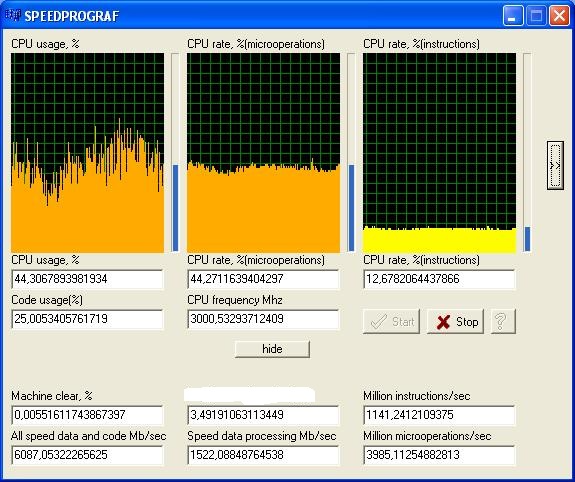


Вот теперь становится понятно, почему «медленный» процессор №4 обогнал самый производительный процессор №1! У него количество выполненных команд во 2-ом режиме архивации в 2 с лишним раза меньше, чем у остальных процессоров! А что, разве одну и ту же работу процессор может выполнить различными путями? Оказывается может! Я не буду приводить другие графики, детализирующие причину таких различий. Моя задача привлечь внимание к существованию проблемы и к ее решению.

SPEEDPROGRAF позволяет достоверно оценить правильные направления оптимизации ядра операционной системы (приложений) – сокращение трассы исполнения (режим трассировки) и рост пиковой производительности. Вот как выглядит 2-ой режим архивации на процессоре №1,

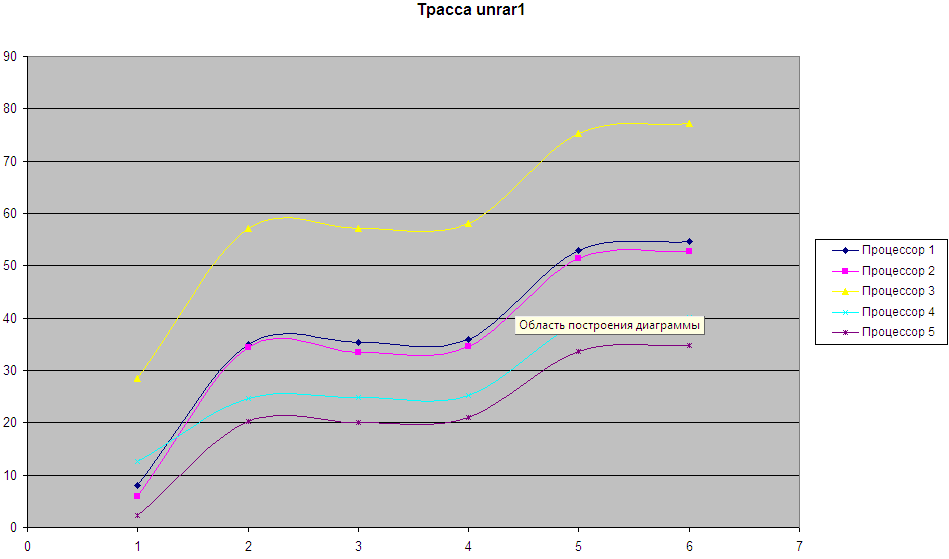


а так, 2-ой режим архивации на процессоре №4.



Загрузка процессора снизилась в 2 раза (трасса короче больше чем в 2 раза), скорость обработки выросла в 2 раза (с 6,81% до 12,68%), а время архивации сократилось в 4 раза (квадратичная зависимость!).

Предлагаемый способ повышения быстродействия спонтанно проявляется на разных платформах.



В подтверждение предыдущей мысли длина трассы разархивации на различных процессорах X86\_64 - кто в лес, кто по дрова?