Определение вычислительной способности процессоров Intel семейств P5 и P6 и сравнение с данными бенчмарков

А.А. Ракитский
ФГОБУ ВПО "СибГУТИ"
e-mail: rakitsky.anton@gmail.com
Б.Я. Рябко
ФГОБУ ВПО "СибГУТИ", ИВТ СО РАН
e-mail: boris@ryabko.net
А.Н. Фионов
ФГОБУ ВПО "СибГУТИ", ИВТ СО РАН
e-mail: fionov@sibsutis.ru

Аннотация

В представленной работе рассматривается применение инновационного метода теоретической оценки производительности компьютера к существующим реальным процессорам. Вычислительная способность процессора зависит от числа различных задач, которые могут быть выполнены за определённое время. Данная характеристика не зависит от каких-либо конкретных подзадач и полностью определяется архитектурой процессора, что позволяет легко вычислить её ещё на стадии проектирования. Применив этот метод к конкретным процессорам Intel семейств P5 и P6, мы сравнили полученные результаты с результатами различных общепризнанных и популярных бенчмарков.

1. Введение

В настоящее время постоянно разрабатываются новые процессоры и вычислительные системы, выпускаются процессоры для мобильных устройств, собираются суперкомпьютеры, и очень остро стоит вопрос оценки их производительности и сравнения таких оценок между собой. На данный момент все применяемые способы оценки производительности вычислительных систем являются эмпирическими и заключаются в том, что оценивается время выполнения конкретного набора задач этими системами. Однако все эти оценки требуют как минимум наличия самой вычислительной системы, а так же их невозможно применить, не имея в наличии рабочей модели. Подход, впервые описанный в работе [1], заключается в том, что при оценке он опирается исключительно на архитектурные особенности самого процессора: на то, сколько времени выполняется та или иная инструкция, на объем кэш-памяти и т.д. Подход основывается на концепции энтропии Шеннона[2], ёмкости дискретного канала без шумов и других его идеях, которые включает в себя теория информации.

Целью работы, которой посвящен доклад, являлась проверка применимости описанного метода для реальных процессоров. Для решения этой задачи необходимо определить вычислительную способность нескольких существующих процессоров, для которых известны и доступны все необходимые характеристики, и сравнить её с результатами тестирования на общепризнанных бенчмарках. Для проверки были выбраны процессоры Intel семейств P5(Pentium и Pentium MMX) и P6(Pentium pro, Pentium II и Pentium III).

2. Вычислительная способность компьютера

В работе [1] представлена упрощённая модель компьютера, в которой он состоит из набора инструкций *I* и некоторого объема памяти *M*. Каждая инструкция компьютера $x \in I$ состоит из имени инструкции и операндов, таких как индексы регистров и адреса ячеек памяти. При этом инструкции, имеющие одинаковое имя, но различные операнды, во множестве *I* будут представлены как различные.

В таком случае вычислительной задачей *P* будем называть некоторую последовательность инструкций $X(P) = x_1 x_2 x_3 \ldots x_n$, где $x_i \in I$. Если задача *P* содержит в себе цикл, который повторяется *n* раз, то последовательность *X* будет содержать в себе тело цикла, повторяющееся *n* раз. Не все последовательности инструкций могут быть допустимы, например, если выполнение последовательности инструкций ведет к какой-нибудь ошибке. Однако на современных компьютерах эти последовательности будут выполняться, и ошибки будут обрабатываться. Поэтому мы не будем подробно рассматривать этот случай.

Время выполнения инструкции *x* обозначается как $\tau(x)$. Тогда время выполнения $\tau(X)$ последовательности инструкций $X(P) = x_1 x_2 x_3 \ldots x_t$ определяется как

$$\tau(X) = \sum_{i=1}^{t} \tau(x_i).$$

Суть заключается в том, что число различных вычислительных задач, время выполнения которых равно *T*, эквивалентно размеру множества всех последовательностей инструкций, чьё время выполнения равно *T*, т.е.

$$\nu(T) = N(T), \tag{1}$$

где $\nu(T)$ - количество разных проблем, чьё время выполнения равно *T*, и

$$N(T) = \left| \{X : \tau(X) = T\} \right| \tag{2}$$

Поэтому,

$$\log \nu(T) = \log N(T) \tag{3}$$

Определение 1. Пусть есть компьютер с набором инструкций *I* и пусть $\tau(x)$ – время выполнения для инструкции $x \in I$. Вычислительная способность $C(I)$ определяется как

$$C(I) = \lim_{T \to \infty} \frac{\log N(T)}{T}. \tag{4}$$

Утверждение 1. Предел (4) существует, если I – конечное множество, времена выполнения $\tau(x)$, $x \in I$, – целые числа и наибольший делитель $\tau(x)$, равен 1.
Доказательство утверждения приводится в работе [1].

Рассмотрим набор инструкций I как алфавит и предположим, что все последовательности букв (инструкций) могут быть выполнены. В таком случае может быть использован метод вычисления пропускной способности канала без потерь, предложенный Шенноном в [2]. Важно отметить, что этот метод может быть использован для нахождения верхней границы вычислительной способности для всех других моделей, потому что для любого компьютера множество допустимых последовательностей инструкций – это подмножество всех слов над «алфавитом» I.

Как и ранее, есть компьютер с набором инструкций I, чьё время выполнения $\tau(x)$, $x \in I$, и все последовательности инструкций являются разрешёнными. Другими словами, если набор I рассматривается как алфавит, то все возможные слова над этим алфавитом

можно рассматривать в качестве допустимых последовательностей инструкций для компьютера. В этом случае, оценить вычислительную способность (4) можно при помощи метода, предложенного Шенноном, который показал, что $C(I)$ равна логарифму от наибольшего действительного решения X_0 следующего уравнения:

$$X^{-\tau(x_1)} + X^{-\tau(x_2)} + \ldots + X^{-\tau(x_S)} = 1, \quad (5)$$

где $I = \{x_1, \ldots, x_S\}$. Иными словами, $C(I) = \log X_0$.

3. Практическое применение метода

В работе [1] рассмотрен пример применения данного метода для простейшего компьютера, рассмотрим тут только основные моменты. Допустим, в данном компьютере есть инструкция, которая в качестве операнда принимает один из 8 регистров, а выполняется за 2 процессорных такта. Тогда в уравнение она войдёт в виде слагаемого $\frac{8}{X^2}$, т.е. это будут 8 инструкций с одинаковым временем работы. Таким образом, зная сколько возможных различных операндов может быть у инструкции и, зная её время работы, мы легко можем получить слагаемое для уравнения. Отдельно стоит заметить про работу с кэш-памятью. Допустим, у нас есть 2х уровневая кэш-память. Обозначим кэш-память 1го уровня как L_1, а 2го уровня – как L_2. Пусть время доступа к кэш-памяти будет равно соответственно $\tau(L_1)$ и $\tau(L_2)$, а время доступа до общей памяти будет $\tau(M)$. Рассмотрим ситуацию, когда инструкция в качестве операнда принимает адрес ячейки памяти. Тогда в первую очередь процессор попытается найти эту ячейку в кэш-памяти первого уровня, если не найдет, то обратится к кэш-памяти второго уровня и только после этого будет искать уже непосредственно в самой оперативной памяти. Таким образом мы получим 3 разных типа инструкций: со временем работы $\tau(L_1)$, со временем работы $\tau(L_1) + \tau(L_2)$ и со временем работы $\tau(L_1) + \tau(L_2) + \tau(M)$. Количество слагаемых будет так же зависеть от того, к какой памяти мы обращаемся.

3.1 Применение метода для компьютеров семейства P5

Для применения метода к процессорам семейства P5, необходимо проанализировать их архитектуру. Во-первых, это были первые суперскалярные процессоры, в которых реализовали 2 конвейера для выполнения инструкций (назовём их u-конвейер и v-конвейер). Следовательно, при описании набора инструкций, необходимо учитывать этот факт. Так как эти конвейеры могут выполнять параллельно только некоторые из инструкций, то для решения данной проблемы предлагается следующий подход. Мы будем рассматривать пары инструкций, которые могут выполняться параллельно, как единую инструкцию и добавим все эти пары в таком виде во множество инструкций процессора. Разделим все целочисленные инструкции на 4 группы:

1) инструкции, которые могут выполняться и на u и на v конвейере;
2) инструкции, которые могут выполняться только на u-конвейере, но при этом параллельно с ними могут выполняться на v-конвейере другие инструкции
3) инструкции, которые могут выполняться только на u-конвейере, и при этом на v-конвейере не должно выполняться других инструкций
4) инструкции, которые могут выполняться на любом конвейере, но могут выполняться в паре, только если выполняются на v-конвейере

В случае операций с плавающей запятой, только некоторые инструкции могут распараллеливаться, и только с инструкцией FXCH.

Операции из блока MMX (который появился в более поздних моделях процессоров этого семейства), так же разбиваются на группы:

1) Все инструкции, которые в качестве операндов используют регистры или память, могут выполняться только на u-конвейере и только параллельно с инструкциями MMX
2) Инструкции сдвига и упаковки могут выполняться на любом конвейере, но не одновременно с операциями такого же вида
3) Инструкции умножения могут выполняться на любом конвейере, но не одновременно с другими инструкциями умножения
4) Все остальные инструкции могут выполняться на любом конвейере и параллельно с любой операцией

Таким образом, составив список всех этих инструкций с описанием операндов и времени выполнения, мы можем при помощи программы получить итоговый набор инструкций. Привести его в докладе не представляется возможным, т.к. этот набор состоит более чем из 20000 слагаемых.

Однако, вычислив уравнение, составленное при помощи этого набора и решив его, мы получим $C(I) \approx 25{,}56$ бит/такт для Pentium и $C(I) \approx 28{,}35$ для Pentium MMX.

3.2 Применение метода для компьютеров семейства P6

Архитектура процессоров семейства P6 радикально отличается от предыдущего семейства. На основе данной архитектуры сконструированы многие современные процессоры, поэтому она требует детального рассмотрения. Условно конвейер этого семейства процессоров можно разбить на несколько основных блоков:

1) блок предсказания переходов
2) блок выборки инструкций
3) декодер инструкций
4) таблица назначения регистров
5) блок переупорядочивания микроопераций
6) станция-резервуар
7) 5 портов, соединяющих с исполнительными блоками
8) блок переупорядочивания памяти
9) блок завершения

Более подробно работа этих блоков разобрана в [3], мы же здесь отметим только основные, необходимые нам моменты. Во-первых, все инструкции разбиваются на простейшие микроинструкции. Блок выборки инструкций может читать за 1 такт блоки размером 16 байт, таким образом, мы можем выделить и одновременно обработать более одной инструкции, в блоке декодирования инструкций, где мы разбиваем их на микрооперации, за 1 такт процессора мы можем получить минимум 2, максимум 6 микроопераций. Т.к. для оценки мы всегда рассматриваем самый лучший случай, то будем считать пропускную способность этого блока за 6 микроопераций. Однако и это не играет роли, т.к. пропускная способность таблицы назначения регистров и блока переупорядочивания микроопераций составляет 3 микрооперации за такт. Следовательно, мы будем считать, что у нас одновременно могут выполняться 3 микрооперации, т.к. блок выполнения состоит из 5 различных блоков, на котором потенциально могло бы обрабатываться одновременно 5 микроопераций. Микрооперации устроены таким образом, что на каждом блоке процессора каждая микрооперация выполняется за 1 такт. Рассмотрим идеальный случай, когда одновременно выполняются 3 микрооперации, тогда временем выполнения одной инструкции будет количество микроопераций, которое она сгенерировала, делённое на 3. После анализа уравнения можно сделать вывод, что нам достаточно определить *C(I)* считая временем выполнения одной инструкции – количество

микроопераций, которое она генерирует (эта информация представлена в [4]), после чего полученный результат умножить на 3. Отдельно стоит отметить, что для Pentium II мы добавляем в набор инструкций, инструкции MMX, а для процессора Pentium III помимо этого ещё и инструкции SSE.

Вычислив уравнение и решив его, мы получим $C(I) \approx 36{,}62$ бит/такт для Pentium Pro, $C(I) \approx 37{,}69$ для Pentium II и $C(I) \approx 42{,}02$ для Pentium III.

4. Анализ и сравнение полученных результатов

Наименование процессора	C(I)	Тактовая частота	Вычислительная способность, Мбит/такт	ICOMP	SPECint95	SPECfp95
Pentium	25,56	150	3834	114	4,13	3,58
Pentium MMX	28,35	200	5670	182	6,37	4,87
Pentium Pro	36,62	200	7324	220	8,59	6,34
Pentium II	37,69	400	15076	440	15,6	12,9
Pentium III	42,02	500	21010	644	21,6	16,2

Так как полученные результаты имеют совершенно различные единицы измерения, то для их сравнения мы рассмотрим их относительное сравнение относительно первого процессора, который мы возьмём за единицу

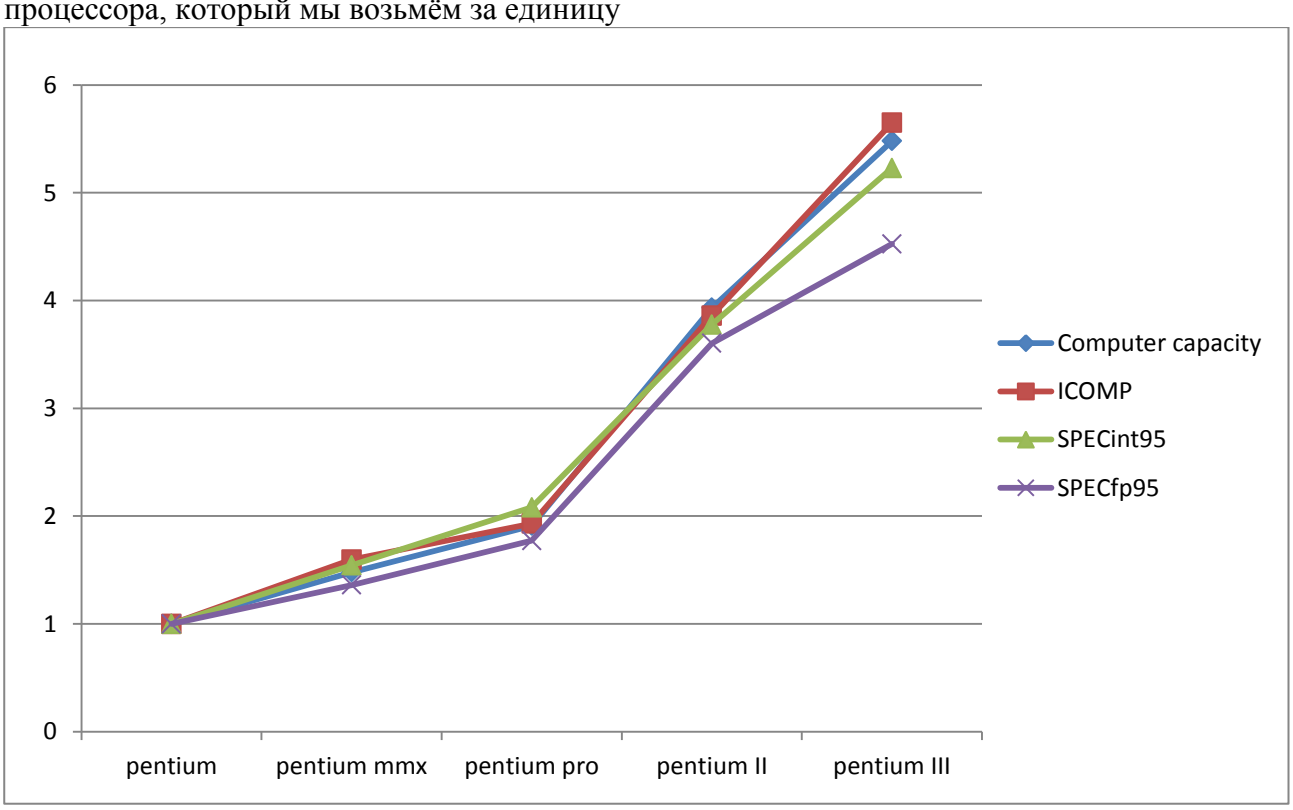

Рисунок 1. График сравнения производительности процессоров

На данной таблице хорошо видно, что вычислительная способность хорошо коррелирует с характеристиками известных бенчмарков, на основе которых чаще всего оцениваются и сравниваются процессоры Intel, что показывает её применимость для реальных процессоров.

Так же можно оценить, как изменялась производительность процессоров при переходе от старого к более новой модели, и сравнить вычислительную способность с данными бенчмарков.

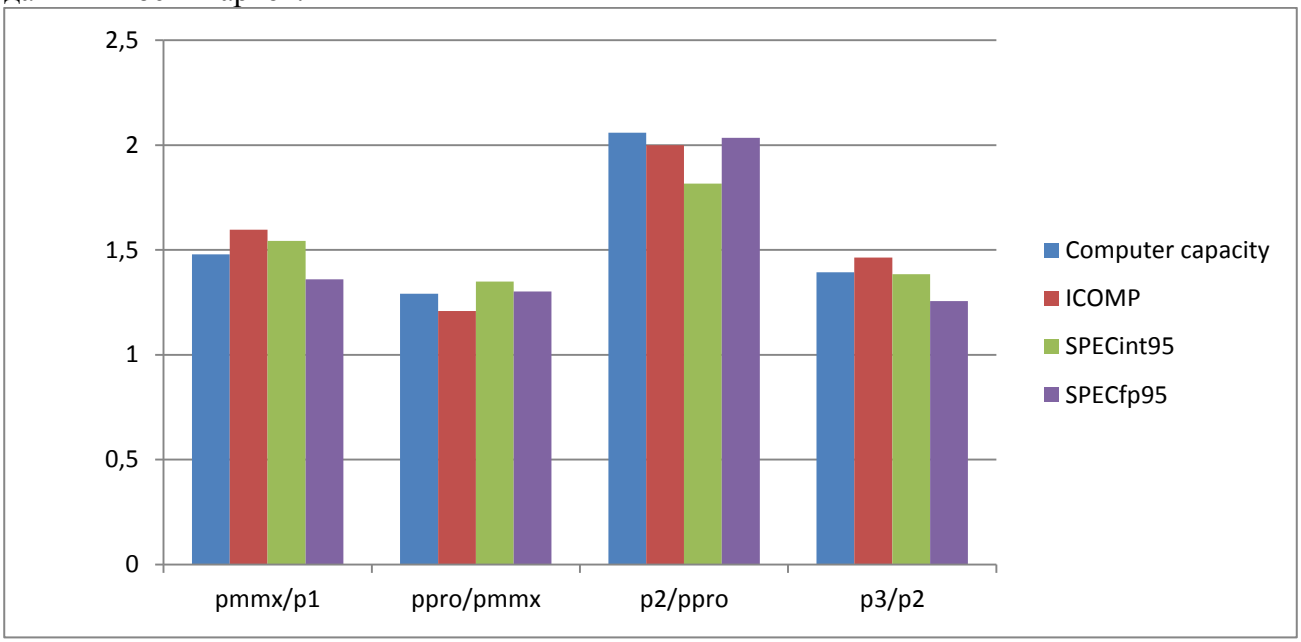

Рисунок 2. Сравнительная характеристика изменения производительности процессоров при переходе к более новой модели

Данный график показывает, что вычислительная способность является объективной и адекватной оценкой производительности процессоров. Так же примечательно то, как себя повела характеристика при переходе от одной архитектуры, к другой, принципиально отличающейся. Этот переход хорошо видно в колонке ppro/pmmx. Характеристика показала себя конкурентоспособной и вполне может применяться для сравнения и оценки более сложных и поздних вычислительных систем, в том числе и суперкомпьютеров.

Литература

1. Boris Ryabko. An information-theoretic approach to estimate the capacity of processing units. // Performance Evaluation, 69 (2012) pp. 267-273 ;
2. C. E. Shannon. A mathematical theory of communication // Bell Sys. Tech. J. , vol. 27, pp. 379-423, pp. 623-656, 1948;
3. A. Fog, “The microarchitecture of Intel, AMD and VIA CPUs An optimization guide for assembly programmers and compiler makers”. Copenhagen University College of Engineering. 2012-02-29.
4. A. Fog, “Lists of instruction latencies, throughputs and microoperation breakdowns for Intel, AMD and VIA CPUs,” Copenhagen University College of Engineering. 2010. http://www.agner.org/optimize/
5. Intel 64 and IA-32 Architectures Software Developers Manual Volume 1: Basic Architecture, Intel Corp., 2011.
6. Intel 64 and IA-32 Architectures Software Developers Manual Volume 2A: Instruction Set Reference, A-M, Intel Corp., 2011.
7. Intel 64 and IA-32 Architectures Software Developers Manual Volume 2B: Instruction Set Reference, N-Z, Intel Corp., 2011.
8. A. S. Tanenbaum, Structured Computer Organization. Prentice Hall, 2005.