УДК 681.3

**Э.И. Ватутин, В.С. Титов**

# ОСОБЕННОСТИ РЕАЛИЗАЦИИ ТЕХНОЛОГИИ HYPER-THREADING В ПРОЦЕССОРАХ INTEL PENTIUM 4 НА ПРИМЕРЕ ВЫПОЛНЕНИЯ КОДА РАЗЛИЧНОГО ТИПА

*Исследованы особенности реализации технологии Hyper-Threading на процессорах Intel Pentium 4 (ядра Northwood и Prescott). Показано, что при правильном использовании технологии можно получить до 50% прибавки в скорости исполнения кода программы. Для сопоставления приведены соотношения скорости выполнения однопоточного и двухпоточного кода на одноядерных и двуядерных процессорах.*

**E.I. Vatutin, V.S. Titov**

# ANALYSIS OF HYPER-THREADING TECHNOLOGY EFFECTIVENESS ON INTEL PENTIUM 4 PROCESSORS EXECUTING DIFFERENT CODE TYPE

*Hyper-Threading Technology characteristics on Intel Pentium 4 processors (Northwood and Prescott cores) is analyzed. Shown that correct technology using can decrease program code execution time up to 50%. Examples of using single-threaded and double-threaded program code on single-core and double-core processors are presented.*

Для повышения степени загрузки исполнительных устройств процессоров Pentium 4 (ядра Northwood, Gallatin и Prescott) фирмой Intel была разработана и реализована технология Hyper-Threading [1]. Благодаря ей один физический процессор видится операционной системой как пара логических процессоров (рис. 1) и способен параллельно выполнять два независимых потока кода.

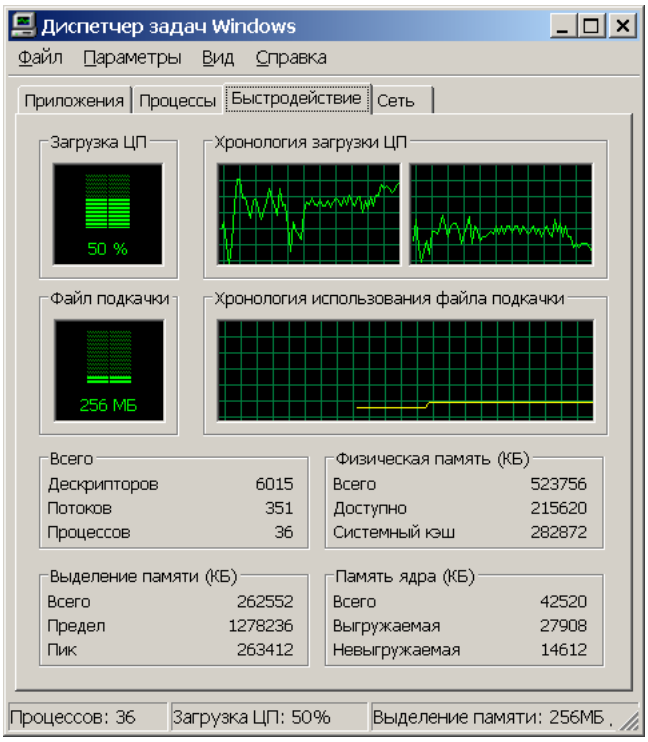


*Рис. 1. Физический процессор включает в своем составе два логических*

Как утверждают официальные руководства и презентации фирмы Intel, при параллельном выполнении нескольких потоков кода технология Hyper-Threading способна обеспечить до 40% прибавки в скорости по сравнению с последовательным исполнением. Синтетические тесты в целом подтверждают указанную информацию, показывая 28–42% прибавки в скорости от параллельного исполнения [2], однако наблюдаются некоторые расхождения с документацией:

1. В документации утверждается, что наибольший выигрыш может быть получен в случае, если различные потоки кода используют различные исполнительные устройства; однако на практике исполнение однотипных потоков, использующих одни и те же исполнительные устройства, приводит к тем же самым 28–42% выигрыша.

2. При интенсивных не выровненных обращениях к памяти в параллельно выполняемых потоках наблюдается около 1% падения производительности на процессорах с ядрами Northwood и порядка **семикратного падения производительности** на процессорах с ядрами Prescott, что по видимому объясняется негативной стороной функционирования плохо документированного механизма replay [3].

Полученные результаты показывают, что для эффективного использования потенциала технологии Hyper-Threading необходимо ее более детальное изучение. В данной статье анализируется выигрыш от использования технологии Hyper-Threading на примере выполнения операции сложения с использованием различных наборов команд (Int ALU, FPU, MMX, SSE), т.е. фактически нагрузки различных исполнительных устройств, различными

(зависимым и независимым) вариантами кода. В данном случае под зависимым кодом понимается код вида

```
add     eax, eax
add     eax, eax
add     eax, eax
add     eax, eax
add     eax, eax
```

где команды выполняются последовательно (не перекрываются во времени), т.к. имеются зависимости RAW (Read After Write) от результата выполнения предыдущей команды. В данном коде теоретически должен наблюдаться наибольший прирост скорости при параллельном выполнении, т.к. исполнительные устройства наименее загружены. Под независимым кодом понимается код вида

```
add     eax, eax
add     ebx, ebx
add     edx, edx
add     esi, esi
add     edi, edi
```

где следующие подряд команды не зависят друг от друга и могут выполняться параллельно во времени. Можно полагать, что данным линейным участком кода (или ему подобным) достигается полная загрузка исполнительных устройств, т.к. в архитектуре процессора Pentium 4 два исполнительных устройства (alu0 port 0 и alu1 port 1) могут исполнять команды целочисленного сложения. Для других исполнительных устройств (FPU, MMX, SSE) возможно параллельное выполнение только одной команды сложения. В некоторых случаях выполнение команд может быть конвейеризовано (FPU), но это не меняет сути. При параллельном выполнении потоков независимого кода теоретически должен наблюдаться наименьший выигрыш от использования технологии Hyper-Threading, т.к. исполнительные устройства полностью загружены.

Для проверки изложенных гипотез была разработана программа (рис. 2), в которой реализованы все возможные варианты кода (буква i соответствует независимому (independent) коду, буква d – зависимому (dependent)).

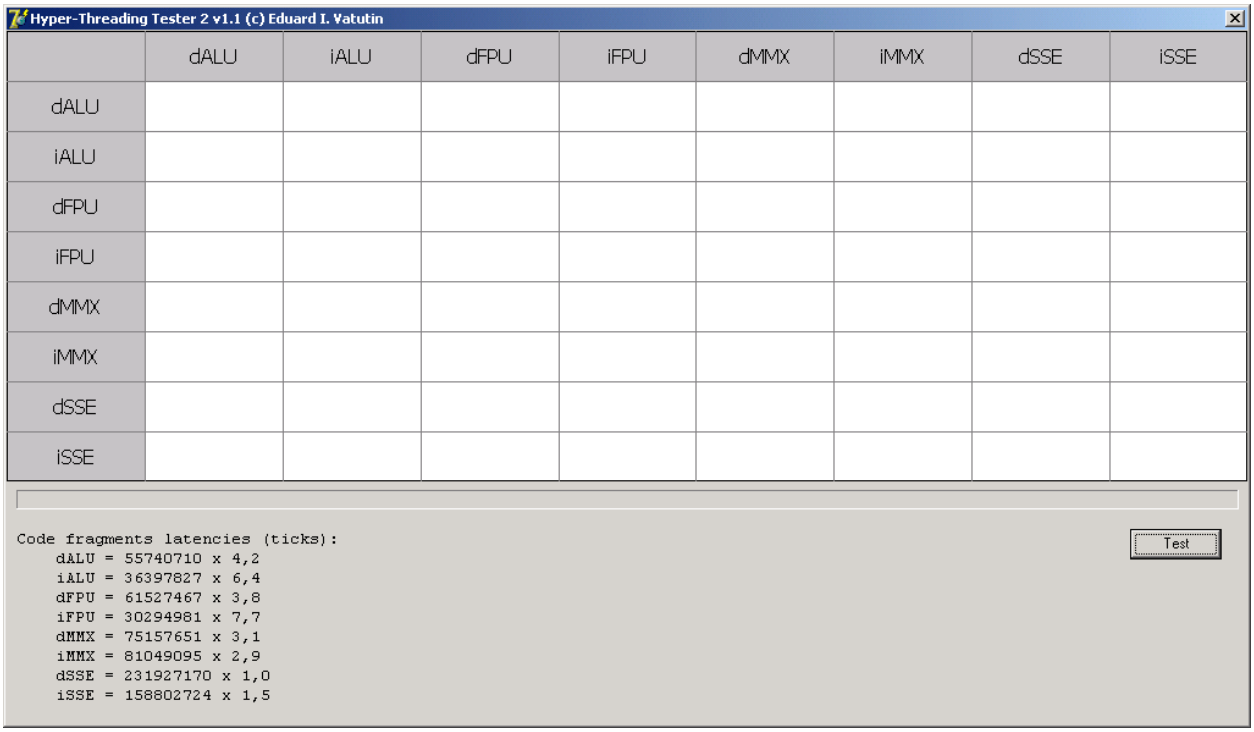


*Рис. 2. Внешний вид программы для тестирования*

Ассемблерный код подпрограмм приведен в таблице.

Таблица. Код подпрограмм, выполняющийся циклически и имитирующий нагрузку различных исполнительных устройств процессора

| **Подпро-грамма** | **ALU** | **FPU** | **MMX** | **SSE** |
|---|---|---|---|---|
| независимый (independent) | add eax, eax<br>add ebx, ebx<br>add edx, edx<br>add esi, esi<br>add edi, edi | fadd st(0), st(2)<br>fxch st(1)<br>fadd st(0), st(3)<br>fxch st(1) | paddw mm1, mm0<br>paddw mm2, mm0<br>paddw mm3, mm0<br>paddw mm4, mm0<br>paddw mm5, mm0<br>paddw mm6, mm0<br>paddw mm7, mm0 | addps xmm1, xmm0<br>addps xmm2, xmm0<br>addps xmm3, xmm0<br>addps xmm4, xmm0<br>addps xmm5, xmm0<br>addps xmm6, xmm0<br>addps xmm7, xmm0 |
| зависимый (dependent) | add eax, eax<br>add eax, eax<br>add eax, eax<br>add eax, eax<br>add eax, eax | fadd st(0), st(1)<br>fadd st(0), st(2) | paddw mm7, mm0<br>paddw mm7, mm1<br>paddw mm7, mm2<br>paddw mm7, mm3<br>paddw mm7, mm4<br>paddw mm7, mm5<br>paddw mm7, mm6 | addps xmm7, xmm0<br>addps xmm7, xmm1<br>addps xmm7, xmm2<br>addps xmm7, xmm3<br>addps xmm7, xmm4<br>addps xmm7, xmm5<br>addps xmm7, xmm6 |

Различные потоки кода запускаются попарно, при этом определяется выигрыш от параллельного исполнения по сравнению с последовательным. Выигрыш отображается как в виде прибавки в скорости в процентах, так и в графическом виде. Число в скобках обозначает количество проведенных тестов, итоговый результат получается путем усреднения результатов отдельных тестов. Для компенсации влияния архитектуры процессора на длительность

исполнения фрагментов кода перед началом тестирования проводится калибровочный прогон тестов (нижняя левая часть формы программы) и корректируется количество итераций выполнения тестовых фрагментов (масштабный коэффициент справа от символа умножения).

Результаты тестирования приведены на рис. 3 и 4.

| | dALU | iALU | dFPU | iFPU | dMMX | iMMX | dSSE | iSSE |
|---|---|---|---|---|---|---|---|---|
| dALU | +2,4% (222) | +1,9% (222) | +18,9% (222) | +21,3% (222) | +22,9% (222) | +26,3% (222) | +26,4% (222) | +18,0% (222) |
| iALU | +1,1% (222) | +2,4% (222) | +19,4% (222) | +22,8% (222) | +23,0% (222) | +25,2% (222) | +25,2% (222) | +16,5% (222) |
| dFPU | +21,3% (222) | +21,8% (222) | +39,0% (222) | +47,2% (222) | +29,8% (222) | +37,9% (222) | +16,0% (222) | +38,2% (222) |
| iFPU | +23,5% (222) | +25,8% (222) | +47,6% (222) | +49,5% (222) | +26,1% (222) | +39,8% (222) | +29,8% (222) | +39,4% (222) |
| dMMX | +6,8% (222) | +24,6% (222) | +29,6% (222) | +25,9% (222) | -0,1% (222) | +24,9% (222) | +24,9% (222) | +24,0% (222) |
| iMMX | +27,3% (222) | +27,3% (222) | +37,8% (222) | +39,2% (222) | +24,8% (222) | +49,7% (222) | +49,7% (222) | +48,6% (222) |
| dSSE | +27,0% (222) | +27,0% (222) | +15,8% (222) | +29,2% (222) | +24,8% (222) | +49,7% (222) | -0,1% (222) | +23,6% (222) |
| iSSE | +24,1% (222) | +23,6% (222) | +39,8% (222) | +39,7% (222) | +24,8% (222) | +49,6% (222) | +24,9% (222) | +48,7% (222) |

*Рис. 3. Результаты выполнения программы на Pentium 4 Northwood 3,06 ГГц (CPUID = 0F29h)*

| | dALU | iALU | dFPU | iFPU | dMMX | iMMX | dSSE | iSSE |
|---|---|---|---|---|---|---|---|---|
| dALU | +4,7% (50) | +24,8% (50) | +27,7% (50) | +24,3% (50) | +30,2% (50) | +31,4% (50) | +31,3% (50) | +30,3% (50) |
| iALU | +24,0% (50) | +39,6% (50) | +36,5% (50) | +37,8% (50) | +40,9% (50) | +40,6% (50) | +40,2% (50) | +39,9% (50) |
| dFPU | +27,7% (50) | +36,8% (50) | +49,3% (50) | +48,4% (50) | +31,5% (50) | +47,1% (50) | +11,1% (50) | +39,3% (50) |
| iFPU | +22,4% (50) | +37,4% (50) | +48,4% (50) | +49,9% (50) | +26,6% (50) | +48,5% (50) | +32,7% (50) | +39,8% (50) |
| dMMX | +32,0% (50) | +41,2% (50) | +31,6% (50) | +29,0% (50) | 0,0% (50) | +23,9% (50) | +15,0% (50) | +29,8% (50) |
| iMMX | +23,5% (50) | +40,7% (50) | +47,1% (50) | +48,5% (50) | +24,9% (50) | +49,9% (50) | +24,8% (50) | +40,2% (50) |
| dSSE | +31,3% (50) | +39,9% (50) | +11,0% (50) | +32,6% (50) | +8,4% (50) | +19,6% (50) | -0,1% (50) | +27,4% (50) |
| iSSE | +28,7% (50) | +39,7% (50) | +37,9% (50) | +39,8% (50) | +29,1% (50) | +39,7% (50) | +29,5% (50) | +49,2% (50) |

*Рис. 4. Результаты выполнения программы на Pentium 4 Prescott 3,20 ГГц (CPUID = 0F34h)*

Анализ полученных результатов позволяет сделать следующие выводы:

1. Выдвинутые выше гипотезы о полной/неполной загрузке исполнительных блоков не подтверждаются результатами эксперимента. Параллельное выполнение потоков независимого кода (iMMX + iMMX, iSSE + iSSE) приводит к 49–50% выигрышу в скорости, в то время как параллельное выполнение потоков зависимого кода (dMMX + dMMX, dSSE + dSSE) не приводит к какому-либо выигрышу: наблюдается проигрыш порядка 0,1%. По видимому это объясняется особенностями функционирования механизма replay [3]. Поскольку код реальных приложений является в большей степени

зависимым, практический выигрыш должен быть значительно ниже максимального теоретического.

2. В целом потоки с независимым кодом получают больший выигрыш в скорости, чем потоки с зависимым кодом. Однако это справедливо не всегда.

3. Параллельное выполнение потоков с кодом для сопроцессора (FPU+FPU) дает хороший выигрыш (39,0–49,9%), что скорее всего объясняется конвейерной организацией этого блока.

4. Выигрыш при исполнении потоков кода на ALU достаточно мал на процессорах с ядром Northwood (1,1–2,4%), в то время как на процессорах с ядром Prescott он более значителен (4,7–39,6%).

5. На процессорах с ядром Prescott в основном наблюдается больший выигрыш в скорости от использования технологии Hyper-Threading по сравнению с процессорами Northwood, что согласуется с данным из официальной документации фирмы Intel. Однако это справедливо не всегда.

Следует также отметить, что выполнение нескольких параллельных потоков кода на компьютере с одним процессором не приводит к значительным издержкам (рис. 5), что обеспечивает возможность эффективного распараллеливания вычислений для процессоров современного поколения без потери эффективности их исполнения на процессорах предыдущих поколений (обеспечение требования обратной совместимости).

| | dALU | iALU | dFPU | iFPU | dMMX | iMMX | dSSE | iSSE |
|---|---|---|---|---|---|---|---|---|
| dALU | -0,1% (37) | -0,6% (37) | 0,0% (37) | -0,3% (37) | -0,3% (37) | +0,1% (37) | +0,2% (37) | -0,6% (37) |
| iALU | +0,3% (37) | -0,4% (37) | -0,1% (37) | -0,1% (37) | -0,1% (37) | +0,0% (37) | +0,0% (37) | -0,3% (37) |
| dFPU | -0,4% (37) | -0,3% (37) | +0,0% (37) | -0,3% (37) | -0,2% (37) | -0,2% (37) | -0,2% (37) | -0,4% (37) |
| iFPU | +0,0% (37) | -0,3% (37) | -0,1% (37) | -0,3% (37) | -0,1% (37) | -0,3% (37) | -0,1% (37) | -0,4% (37) |
| dMMX | -0,3% (37) | -0,2% (37) | -0,1% (37) | -0,7% (37) | -0,3% (37) | -0,3% (37) | -0,2% (37) | -0,2% (37) |
| iMMX | -0,3% (37) | -0,1% (37) | -0,5% (37) | 0,0% (37) | -0,3% (37) | -0,1% (37) | -0,2% (37) | -0,1% (37) |
| dSSE | 0,0% (37) | +0,1% (37) | -0,4% (37) | -0,2% (37) | -0,2% (37) | -0,1% (37) | -0,2% (37) | -0,2% (37) |
| iSSE | +0,1% (37) | -0,3% (37) | -0,3% (37) | -0,1% (37) | -0,1% (37) | +0,0% (37) | -0,7% (37) | -0,3% (37) |

*Рис. 5. Результаты выполнения программы на процессоре Pentium III Coppermine 850 МГц (CPUID = 068Ah)*

Поведение программы на настоящем двуядерном процессоре показано на рис. 6.

| | dALU | iALU | dFPU | iFPU | dMMX | iMMX | dSSE | iSSE |
|---|---|---|---|---|---|---|---|---|
| dALU | +49,8% (10) | +49,4% (10) | +49,7% (10) | +49,7% (10) | +49,6% (10) | +49,7% (10) | +49,7% (10) | +49,5% (10) |
| iALU | +49,4% (10) | +49,6% (10) | +49,5% (10) | +49,4% (10) | +49,6% (10) | +49,5% (10) | +49,5% (10) | +49,6% (10) |
| dFPU | +49,6% (10) | +49,5% (10) | +49,7% (10) | +49,7% (10) | +49,6% (10) | +49,7% (10) | +49,7% (10) | +49,6% (10) |
| iFPU | +49,7% (10) | +49,6% (10) | +49,6% (10) | +49,6% (10) | +49,7% (10) | +49,7% (10) | +49,7% (10) | +49,5% (10) |
| dMMX | +49,6% (10) | +49,2% (10) | +49,7% (10) | +49,7% (10) | +49,6% (10) | +49,7% (10) | +49,6% (10) | +49,5% (10) |
| iMMX | +49,6% (10) | +49,3% (10) | +49,7% (10) | +49,6% (10) | +49,7% (10) | +49,7% (10) | +49,7% (10) | +49,6% (10) |
| dSSE | +49,7% (10) | +49,4% (10) | +49,7% (10) | +49,6% (10) | +49,6% (10) | +49,7% (10) | +49,6% (10) | +49,5% (10) |
| iSSE | +49,7% (10) | +49,5% (10) | +49,7% (10) | +49,6% (10) | +49,5% (10) | +49,7% (10) | +49,5% (10) | +49,7% (10) |

*Рис. 6. Результаты выполнения программы на процессоре Core 2 Duo E6300 (CPUID=06F6h)*

Видно, что эффективность параллельного исполнения во всех случаях составляет величину, близкую к 50% (теоретический максимум), при этом накладные расходы на синхронизацию (отклонения от теоретического максимума) составляют величину не более 1%.

## Библиографический список


1. Intel IA-32 Architecture Optimization Reference Manual. Order number 248966-011. http://developer.intel.com
2. Ватутин Э.И. Использование технологии Hyper-Threading в вычислительных приложениях // Тезисы докладов XXXIII вузовской научно-технической конференции студентов и аспирантов в области научных исследований «Молодежь и XXI век». Ч. 1. Курск: изд-во КурскГТУ, 2005. С. 18–19.
3. Керученько Я., Малич Ю., Левченко В. Replay: неизвестные особенности функционирования ядра Netburst. http://www.fcenter.ru , 2005.