**体系结构 第12章**

**陈彦帆 2018K8009918002**

1. (1)

测单核最大MIPS：考虑4条指令并行计算

#include <stdio.h>

#include <sys/time.h>

#include <stdlib.h>

int main()

{

struct timeval tv1, tv2;

unsigned long time,mintime=-1;

for(int i=0;i<100;i++){

gettimeofday(&tv1,NULL);

asm volatile(

".rept 100000\n\t"

"sub $3, %%ecx\n\t"

"add $0, %%ebx\n\t"

"sal $5, %%eax\n\t"

"or $2, %%edx\n\t"

".endr\n\t"

:

:

);

gettimeofday(&tv2, NULL);

time = (tv2.tv\_sec-tv1.tv\_sec)\*1000000+(tv2.tv\_usec-tv1.tv\_usec);

if(time<mintime)

mintime = time;

}

printf("The cpu core MIPS is %lf \n", 1.0\*400000/mintime);

return 0;

}

测试结果：(i7-8550u)

The cpu core MIPS is 4819.277108

(2) 测最大MFLOPS: 采用8条并行指令

#include <stdio.h>

#include <sys/time.h>

#include <stdlib.h>

int main()

{

struct timeval tv1, tv2;

unsigned long time,mintime=-1;

for(int i=0;i<100;i++){

gettimeofday(&tv1,NULL);

asm volatile(

".rept 100000\n\t"

"addsd %%xmm0,%%xmm0\n\t"

"addsd %%xmm1,%%xmm1\n\t"

"addsd %%xmm2,%%xmm2\n\t"

"addsd %%xmm3,%%xmm3\n\t"

"addsd %%xmm4,%%xmm4\n\t"

"addsd %%xmm5,%%xmm5\n\t"

"addsd %%xmm6,%%xmm6\n\t"

"addsd %%xmm7,%%xmm7\n\t"

".endr\n\t"

:

:

);

gettimeofday(&tv2, NULL);

time = (tv2.tv\_sec-tv1.tv\_sec)\*1000000+(tv2.tv\_usec-tv1.tv\_usec);

if(time<mintime)

mintime = time;

}

printf("The cpu core MFLOPS is %lf \n", 1.0\*800000/mintime);

return 0;

}

结果：(i7-8550u)

The cpu core MFLOPS is 3587.443946

2. (1) 笔记本(i7-8550u)不连接电源测得结果如下：

STREAM version $Revision: 5.10 $

-------------------------------------------------------------

This system uses 8 bytes per array element.

-------------------------------------------------------------

Array size = 10000000 (elements), Offset = 0 (elements)

Memory per array = 76.3 MiB (= 0.1 GiB).

Total memory required = 228.9 MiB (= 0.2 GiB).

Each kernel will be executed 10 times.

The \*best\* time for each kernel (excluding the first iteration)

will be used to compute the reported bandwidth.

-------------------------------------------------------------

Number of Threads requested = 4

Number of Threads counted = 4

-------------------------------------------------------------

Your clock granularity/precision appears to be 1 microseconds.

Each test below will take on the order of 8405 microseconds.

(= 8405 clock ticks)

Increase the size of the arrays if this shows that

you are not getting at least 20 clock ticks per test.

-------------------------------------------------------------

WARNING -- The above is only a rough guideline.

For best results, please be sure you know the

precision of your system timer.

-------------------------------------------------------------

Function Best Rate MB/s Avg time Min time Max time

Copy: 14773.9 0.013792 0.010830 0.018222

Scale: 15343.0 0.012044 0.010428 0.017743

Add: 16427.0 0.015369 0.014610 0.018114

Triad: 16055.8 0.017152 0.014948 0.023534

-------------------------------------------------------------

Solution Validates: avg error less than 1.000000e-13 on all three arrays

-------------------------------------------------------------

(2) 将笔记本(i7-8550u)连接电源测得的结果如下：

-------------------------------------------------------------

STREAM version $Revision: 5.10 $

-------------------------------------------------------------

This system uses 8 bytes per array element.

-------------------------------------------------------------

Array size = 10000000 (elements), Offset = 0 (elements)

Memory per array = 76.3 MiB (= 0.1 GiB).

Total memory required = 228.9 MiB (= 0.2 GiB).

Each kernel will be executed 10 times.

The \*best\* time for each kernel (excluding the first iteration)

will be used to compute the reported bandwidth.

-------------------------------------------------------------

Number of Threads requested = 4

Number of Threads counted = 4

-------------------------------------------------------------

Your clock granularity/precision appears to be 1 microseconds.

Each test below will take on the order of 6844 microseconds.

(= 6844 clock ticks)

Increase the size of the arrays if this shows that

you are not getting at least 20 clock ticks per test.

-------------------------------------------------------------

WARNING -- The above is only a rough guideline.

For best results, please be sure you know the

precision of your system timer.

-------------------------------------------------------------

Function Best Rate MB/s Avg time Min time Max time

Copy: 17841.0 0.009400 0.008968 0.010536

Scale: 17665.8 0.009527 0.009057 0.010635

Add: 19412.8 0.012749 0.012363 0.013077

Triad: 19525.0 0.012999 0.012292 0.015778

-------------------------------------------------------------

Solution Validates: avg error less than 1.000000e-13 on all three arrays

-------------------------------------------------------------

笔记本插电的情况下，内存实际带宽有大概20%的提升。此时，北桥频率和cpu频率都会上升。推测内存带宽提升与北桥频率提升或cpu频率提升都可能有关。

（3）从双精度修改为单精度，笔记本插电测试结果如下：

-------------------------------------------------------------

STREAM version $Revision: 5.10 $

-------------------------------------------------------------

This system uses 4 bytes per array element.

-------------------------------------------------------------

Array size = 10000000 (elements), Offset = 0 (elements)

Memory per array = 38.1 MiB (= 0.0 GiB).

Total memory required = 114.4 MiB (= 0.1 GiB).

Each kernel will be executed 10 times.

The \*best\* time for each kernel (excluding the first iteration)

will be used to compute the reported bandwidth.

-------------------------------------------------------------

Number of Threads requested = 4

Number of Threads counted = 4

-------------------------------------------------------------

Your clock granularity/precision appears to be 1 microseconds.

Each test below will take on the order of 3010 microseconds.

(= 3010 clock ticks)

Increase the size of the arrays if this shows that

you are not getting at least 20 clock ticks per test.

-------------------------------------------------------------

WARNING -- The above is only a rough guideline.

For best results, please be sure you know the

precision of your system timer.

-------------------------------------------------------------

Function Best Rate MB/s Avg time Min time Max time

Copy: 17719.0 0.005040 0.004515 0.005686

Scale: 17730.2 0.005059 0.004512 0.007144

Add: 19151.3 0.006569 0.006266 0.008046

Triad: 19831.2 0.006413 0.006051 0.006643

-------------------------------------------------------------

Solution Validates: avg error less than 1.000000e-06 on all three arrays

把结果与(2)的结果对比，可以看出，单精度测试时，平均时间约为双精度的一半，因为测试的数据减少了一半。而测得的实际带宽几乎不变。

3.

libquantum程序的重要测试对象是SIMD （单指令流多数据流）性能，对微结构的压力主要是对平行处理微元的测试。

猜测icc编译器采用向量化、预取，自动并行化，使其分值高于gcc，达到上百分。

4. 排序10000个数：使用perf stat 命令：

(1)Bubble:

Performance counter stats for './t1':

241.56 msec task-clock # 0.997 CPUs utilized

1 context-switches # 0.004 K/sec

0 cpu-migrations # 0.000 K/sec

55 page-faults # 0.228 K/sec

411,004,941 cycles # 1.701 GHz

950,997,738 instructions # 2.31 insn per cycle

100,203,003 branches # 414.822 M/sec

23,043 branch-misses # 0.02% of all branches

0.242331083 seconds time elapsed

0.242384000 seconds user

0.000000000 seconds sys

(2)Shell:

Performance counter stats for './t1':

2.59 msec task-clock # 0.729 CPUs utilized

0 context-switches # 0.000 K/sec

0 cpu-migrations # 0.000 K/sec

63 page-faults # 0.024 M/sec

2,892,670 cycles # 1.115 GHz

5,301,454 instructions # 1.83 insn per cycle

621,968 branches # 239.807 M/sec

6,906 branch-misses # 1.11% of all branches

0.003559222 seconds time elapsed

0.003565000 seconds user

0.000000000 seconds sys

(3)qsort(Quicksort):

Performance counter stats for './t1':

1.47 msec task-clock # 0.658 CPUs utilized

0 context-switches # 0.000 K/sec

0 cpu-migrations # 0.000 K/sec

72 page-faults # 0.049 M/sec

2,060,899 cycles # 1.400 GHz

3,144,719 instructions # 1.53 insn per cycle

554,192 branches # 376.509 M/sec

8,744 branch-misses # 1.58% of all branches

0.002237886 seconds time elapsed

0.002248000 seconds user

0.000000000 seconds sys

比较三个结果，IPC依次递减，转移失败率依次递增。看出冒泡排序对吞吐率要求更高，希尔排序和快速排序对转移猜测准确率要求更高。

5. Linpack来源https://people.sc.fsu.edu/~jburkardt/c\_src/linpack\_bench/linpack\_bench.c

测试结果如下：

gprof linpack

Flat profile:

Each sample counts as 0.01 seconds.

% cumulative self self total

time seconds seconds calls ms/call ms/call name

83.05 0.39 0.39 501499 0.00 0.00 daxpy

6.39 0.42 0.03 1 30.02 418.79 dgefa

4.26 0.44 0.02 2000000 0.00 0.00 r8\_random

4.26 0.46 0.02 main

2.13 0.47 0.01 2 5.00 15.01 r8mat\_gen

0.00 0.47 0.00 1002000 0.00 0.00 r8\_max

0.00 0.47 0.00 999 0.00 0.00 dscal

0.00 0.47 0.00 999 0.00 0.00 idamax

0.00 0.47 0.00 4 0.00 0.00 cpu\_time

0.00 0.47 0.00 2 0.00 0.00 timestamp

0.00 0.47 0.00 1 0.00 1.56 dgesl

0.00 0.47 0.00 1 0.00 0.00 r8\_epsilon

6.

Memory latencies in nanoseconds - smaller is better

(WARNING - may not be correct, check graphs)

------------------------------------------------------------------------------

Host OS Mhz L1 $ L2 $ Main mem Rand mem Guesses

--------- ------------- ---- ---- -------- -------- -------

ubuntu Linux 5.4.0-4 -1 1.1010 3.3190 18.8 104.9

注：单位为ns。Main mem对应L3 cache。

8.

桌面基准测试所面向的测试对象主要为通用计算机系统。而嵌入式基准测试的对象为嵌入式计算机系统。嵌入式系统通常是面向特定应用的，大多工作在为特定用户群设计的系统中，嵌入式系统中的软件一般都固化在存储器芯片或单片机本身中，而不是存贮于磁盘等载体中。

因此，嵌入式基准测试更注重于机器在某一特定领域上的功能。而桌面基准测试更注重全面性和通用性。

9. ARM CotexA-57

**Table 11-5 Common Event Identification Register 0 bit assignments**

| **Bit** | **Name** | **Event number** | **Value** | **Event implemented if bit set to 1 or not implemented if bit set to 0** |
| --- | --- | --- | --- | --- |
| [31] | - | 0x1F | 0 | Reserved, *RES0*. |
| [30] | CH | 0x1E | 1 | Chain.[a](http://infocenter.arm.com/help/topic/com.arm.doc.ddi0488h/way1382543397991.html" \l "fntarg_a) An odd-numbered counter increments when an overflow occurs on the preceding even-numbered counter. For even-numbered counters, does not count. |
| [29] | BC | 0x1D | 1 | Bus cycle. |
| [28] | TW | 0x1C | 1 | TTBR write, architecturally executed, condition check pass - write to translation table base. |
| [27] | IS | 0x1B | 1 | Instruction speculatively executed. |
| [26] | ME | 0x1A | 1 | Local memory error. |
| [25] | BA | 0x19 | 1 | Bus access. |
| [24] | DC2W | 0x18 | 1 | Level 2 data cache Write-Back. |
| [23] | DC2R | 0x17 | 1 | Level 2 data cache refill. |
| [22] | DC2A | 0x16 | 1 | Level 2 data cache access. |
| [21] | DC1W | 0x15 | 1 | Level 1 data cache Write-Back. |
| [20] | IC1A | 0x14 | 1 | Level 1 instruction cache access. |
| [19] | MA | 0x13 | 1 | Data memory access. |
| [18] | BP | 0x12 | 1 | Predictable branch speculatively executed. |
| [17] | CC | 0x11 | 1 | Cycle. |
| [16] | BM | 0x10 | 1 | Mispredicted or not predicted branch speculatively executed. |
| [15] | UL | 0x0F | 0 | Instruction architecturally executed, condition check pass - unaligned load or store. |
| [14] | BR | 0x0E | 0 | Instruction architecturally executed, condition check pass - procedure return. |
| [13] | BI | 0x0D | 0 | Instruction architecturally executed - immediate branch. |
| [12] | PW | 0x0C | 0 | Instruction architecturally executed, condition check pass - software change of the PC. |
| [11] | CW | 0x0B | 1 | Instruction architecturally executed, condition check pass - write to CONTEXTIDR. |
| [10] | ER | 0x0A | 1 | Instruction architecturally executed, condition check pass - exception return. |
| [9] | ET | 0x09 | 1 | Exception taken. |
| [8] | IA | 0x08 | 1 | Instruction architecturally executed. |
| [7] | ST | 0x07 | 0 | Instruction architecturally executed, condition check pass - store. |
| [6] | LD | 0x06 | 0 | Instruction architecturally executed, condition check pass - load. |
| [5] | DT1R | 0x05 | 1 | Level 1 data TLB refill.This event is implemented. |
| [4] | DC1A | 0x04 | 1 | Level 1 data cache access. |
| [3] | DC1R | 0x03 | 1 | Level 1 data cache refill. |
| [2] | IT1R | 0x02 | 1 | Level 1 instruction TLB refill. |
| [1] | IC1R | 0x01 | 1 | Level 1 instruction cache refill. |
| [0] | SI | 0x00 | 1 | Instruction architecturally executed, condition check pass - software increment. |

10.

性能测量用于理解已经搭建好的系统或者原型系统。而模拟建模的方法使用软件的方式来模拟计算机系统硬件在体系结构层面的功能和性能特性。模拟建模可以在脱离真实系统的情况下测试，而且系统的配置可以容易地改变或重新配置，更加灵活，节省成本。而性能测量的精度更高。

11.

SimPoint首先找到程序执行的相位，然后对能够代表每个相位的部分进行采样和模拟仿真。因为模拟仿真的程序是采样的片段而非原来的整个程序，所以这种采样方法能大大减少模拟建模时间。

12.

让真实机器和模拟器同时运行一系列微测试程序。这些测试程序对处理器核的某个模块进行了测试，同时排除了处理器其他模块的影响，所以特别适合处理器设计时对某个模块的选择和优化。在相同的参数设置下，模拟器和真实机器的微测试程序运行结果越接近，则准确率越高。

13.

测试机器 i7 8550u Ubuntu18.04(虚拟机)

SPEC CPU2000得分如下：（注：分数中空白部分为编译不通过或无法运行，不参与总分计算）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cint |  | 无优化 | O2 | O3 |
| 164 | gzip | 1115.937 | 1957.679 | 2023.939 |
| 175 | vpr | 1395.299 | 2687.943 | 2883.525 |
| 176 | gcc |  | 4479.916 | 4757.141 |
| 181 | mcf | 3270.796 | 3964.936 | 4610.887 |
| 186 | crafty | 2468.356 | 3478.141 | 3730.504 |
| 197 | parser | 1260.129 | 2169.287 | 2361.829 |
| 252 | eon | 584.6977 | 4848.01 | 5997.671 |
| 253 | perlbmk | 2866.788 |  |  |
| 254 | gap | 3946.47 | 3527.612 | 3842.795 |
| 255 | vortex | 2447.246 | 3689.296 | 4714.202 |
| 256 | bzip2 | 2447.246 | 2397.145 | 2845.893 |
| 300 | twolf | 2187.466 | 3448.948 | 3873.694 |
| all |  | 1928.581 | 3206.068 | 3608.542 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cfp |  | 无优化 | O2 | O3 |
| 168 | wupwise |  |  |  |
| 171 | swim |  |  |  |
| 172 | mgrid |  |  |  |
| 173 | applu |  |  |  |
| 177 | mesa | 2281.226 | 3944.957 | 4259.612 |
| 178 | galgel |  |  |  |
| 179 | art | 5929.445 | 11375.67 | 12408.5 |
| 183 | equake | 2277.296 | 6900.745 | 8127.639 |
| 187 | facerec |  |  |  |
| 188 | ammp | 1582.645 | 3241.212 | 2838.115 |
| 189 | lucas |  |  |  |
| 191 | fma3d |  |  |  |
| 200 | sixtrack |  |  |  |
| 301 | apsi |  |  |  |
| all |  | 2642.386 | 5628.668 | 5909.095 |