ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ» ФАКУЛЬТЕТ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Кафедра ВС

Лабораторная работа №1

"Оптимизация ветвлений и циклов"

Выполнил: студент группы ИВ-222 Терешков Р. В.

> Проверил: к.т.н. Курносов М. Г.

Вычисления проводились на процессоре Intel Core i5 - 760 (8M Cache, 2,80 GHz).

Задание 1

| n | T _{blend_map} | T _{blend_map_opt} | s | |
|------|------------------------|----------------------------|------|--|
| 10^5 | 0.633 | 0.540 | 1.17 | |
| 10^6 | 6.696 | 5.757 | 1.16 | |

n — количество элементов в массивах x, y, z;

S — ускорение;

 $T_{\it blend\ map}$ — время выполнения функции blend_map(), [msec];

 $T_{\it blend_map_opt}$ — время выполнения оптимизированной функции blend_map(), [msec].

Intel VTune Amplifier

> Effective time by utilization (red = poor)

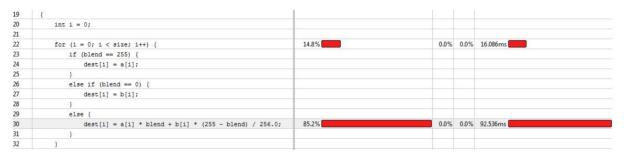


Рис. 1 — blend_map()

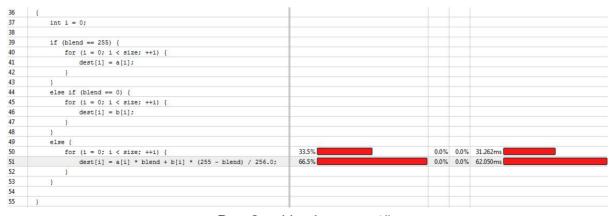


Рис. 2 — blend_map_opt()

Задание 2

| D | _ | 2 | | 4 | | 8 | 8 | | 16 | |
|--------|---------|---------|------|---------|------|--------|------|--------|------|--|
| n | t | t | s | t | s | t | s | t | s | |
| 16 MiB | 48.282 | 38.625 | 1.25 | 27.146 | 1.78 | 24.794 | 1.95 | 23.487 | 2.06 | |
| 64 MiB | 185.236 | 139.514 | 1.33 | 119.821 | 1.55 | 98.687 | 1.88 | 93.474 | 1.98 | |

D — глубина раскрутки цикла;

n — количество элементов (размер) массива v;

t — время выполнения программы, [msec];

s — ускорение.

Intel VTune Amplifier

> Effective time by utilization (red = poor)

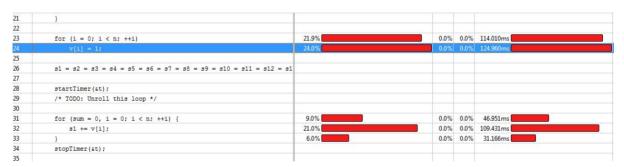


Рис. 3 — до раскрутки

| 23 | for (i = 0; i < n; ++i) | 29.2% | 0.0% | 0.0% | 125.055ms |
|----|---|-------|------|------|-----------|
| 4 | v[i] = 1; | 26.9% | 0.0% | 0.0% | 115.153ms |
| 5 | | | | | |
| 26 | s1 = s2 = s3 = s4 = s5 = s6 = s7 = s8 = s9 = s10 = s11 = s12 = s1 | | | | |
| 27 | | | | | |
| 28 | startTimer(&t); | | | | |
| 29 | /* TODO: Unroll this loop */ | | | | |
| 30 | | | | | |
| 31 | for (sum = 0, i = 0; i < n; i += 16) { | | | | |
| 32 | s1 += v[i]; | 3.7% | 0.0% | 0.0% | 15.628ms |
| 33 | s2 += v[i + 1]; | | | | |
| 34 | s3 += v[i + 2]; | | | | |
| 35 | s4 += v[i + 3]; | | | | |
| 36 | s5 += v[i + 4]; | | | | |
| 37 | s6 += v[i + 5]; | 3.6% | 0.0% | 0.0% | 15.585ms |
| 38 | s7 += v[i + 6]; | 3.3% | 0.0% | 0.0% | 14.141ms |
| 39 | s8 += v[i + 7]; | | | | |
| 40 | s9 += v[i + 8]; | 3.6% | 0.0% | 0.0% | 15.578ms |
| 41 | s10 += v[i + 9]; | 4.0% | 0.0% | 0.0% | 17.105ms |
| 42 | s11 += v[i + 10]; | | | | |
| 43 | s12 += v[i + 11]; | | | | |
| 44 | s13 += v[i + 12]; | 3.7% | 0.0% | 0.0% | 15.672ms |
| 45 | s14 += v[i + 13]; | | | | |
| 46 | s15 += v[i + 14]; | | | | |
| 47 | s16 += v[i + 15]; | | | | |
| 48 |) | | | | |

Рис. 4 — после раскрутки

High Resolution Timing (Windows)

```
hr time.h
#include <windows.h>
typedef struct {
    LARGE INTEGER start;
    LARGE INTEGER stop;
} stopWatch;
void startTimer( stopWatch *timer);
void stopTimer( stopWatch *timer);
double LIToSecs ( LARGE INTEGER * L);
double getElapsedTime( stopWatch *timer);
hr_time.c
#include <windows.h>
#ifndef hr timer
#include "hr time.h"
#define hr timer
#endif
void startTimer( stopWatch *timer) {
    QueryPerformanceCounter(&timer->start);
}
void stopTimer( stopWatch *timer) {
    QueryPerformanceCounter(&timer->stop);
}
double LIToSecs( LARGE INTEGER * L) {
    LARGE INTEGER frequency;
     QueryPerformanceFrequency( &frequency);
     return ((double)L->QuadPart /(double)frequency.QuadPart);
}
double getElapsedTime( stopWatch *timer) {
     LARGE INTEGER time;
     time.QuadPart = timer->stop.QuadPart - timer->start.QuadPart;
    return LIToSecs( &time) ;
}
```

[http://cplus.about.com/od/howtodothingsin1/a/timing.htm]

Вычисления проводились на процессоре Intel Atom N2600 (1M Cache, 1.6 GHz).

Задание 1

| n | T _{blend_map} | T _{blend_map_opt} | s | |
|------|------------------------|----------------------------|------|--|
| 10^5 | 1.933 | 1.716 | 1.13 | |
| 10^6 | 15.285 | 14.07 | 1.09 | |

n — количество элементов в массивах x, y, z;

S — ускорение;

 T_{blend_map} — время выполнения функции blend_map(), [msec]; $T_{blend_map_opt}$ — время выполнения оптимизированной функции blend_map(), [msec].

Linux Perf

blend map():

```
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.005 MB perf.data (~225 samples) ]
```

```
Samples: 122 of event 'branch-misses', Event count (approx.): 28026
        branch [kernel.kallsyms] [k] 0xc11176f0
        branch
                ld-2.19.50
                                   [.] do lookup x
 3.30% branch ld-2.19.so
                                   [.] dl check all versions
 3.15% branch
               ld-2.19.so
                                   [.] dl cache libcmp
        branch
               libc-2.19.so
                                   [.] handle intel
 1.43% branch branch
                                  [.] blend map
                libc-2.19.50
                                       strlen ia32
       branch
                                   [.]
 0.90% branch
               libc-2.19.so
                                   [.] vfprintf
 0.76% branch libc-2.19.so
                                      hack digit.13386
                                   [.]
```

```
/home/statigbloom/Documents/HPCS/hps/branch/branch
blend map
              movl
                    $0x0,-0x4(%ebp)
            ↓ jmp
                     if (blend == 255) {
                         dest[i] = a[i];
                      -0x4(%ebp),%eax
              mov
                      0x0(,%eax,8),%edx
              lea
                      0x8(%ebp), %eax
              mov
              add
                     %edx, %eax
                      -0x4(%ebp),%edx
              mov
              lea
                      0x0(,%edx,8),%ecx
                      0xc(%ebp),%edx
              mov
                      %ecx,%edx
              add
                      (%edx)
              fldl
              fstpl
                     (%eax)
            ↓ jmp
                     cb
                     } else if (blend == 0) {
              cmpl
                     $0x0,0x18(%ebp)
                        dest[i] = b[i];
                      -0x4(%ebp),%eax
              mov
                      0x0(,%eax,8),%edx
              lea
                      0x8(%ebp),%eax
              mov
                     %edx,%eax
-0x4(%ebp),%edx
              add
              mov
                      0x0(,%edx,8),%ecx
              lea
                      0x10(%ebp),%edx
              mov
                     %ecx,%edx
              add
                      (%edx)
              fldl
              fstpl (%eax)
              jmp
                     cb
                     } else {
                         dest[i] = a[i] * blend + b[i] * (255 - blend) / 256.0;
                      -0x4(%ebp),%eax
              mov
              lea
                      0x0(,%eax,8),%edx
```

blend_map_opt():

```
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.004 MB perf.data (~183 samples) ]
```

```
Samples: 97 of event 'branch-misses', Event count (approx.): 24884

97.13% branch [kernel.kallsyms] [k] 0xc111868b

2.34% branch branch [.] blend map opt

0.53% branch libc-2.19.so [.] _IO_file_overflow@@GLIBC_2.1
```

```
/home/statigbloom/Documents/HPCS/hps/branch/branch
blend map opt
                 int i = 0;
                 if (blend == 255) {
                     for (i = 0; i < size; i++) {
               addl
                      $0x1,-0x4(%ebp)
                      -0x4(%ebp),%eax
               mov
                      0x14(%ebp), %eax
               cmp
             t jl
                      1f
                      fe
             ↓ jmp
                         dest[i] = a[i];
                 } else if (blend == 0) {
                     $0x0,0x18(%ebp)
               cmpl
             ↓ jne
                      91
                      for (i = 0; i < size; i++) {
               movl
                      $0x0,-0x4(%ebp)
             ↓ jmp
                          dest[i] = b[i];
                       -0x4(%ebp),%eax
               mov
                      0x8(%ebp), %eax
               mov
               add
                      %edx,%eax
                      -0x4(%ebp),%edx
               mov
                      0x0(,%edx,8),%ecx
               lea
               mov
                      0x10(%ebp),%edx
               add
                      %ecx, %edx
               fldl
                      (%edx)
               fstpl (%eax)
                 if (blend == 255) {
                     for (i = 0; i < size; i++) {
                         dest[i] = a[i];
                 } else if (blend == 0) {
                     for (i = 0; i < size; i++) {
               addl
                      $0x1,-0x4(%ebp)
                      -0x4(%ebp),%eax
               mov
                      0x14(%ebp),%eax
               cmp
```

Задание 2

| D | | 2 | | 4 | | 8 | | 16 | |
|--------|---------|---------|------|---------|------|---|---|----|---|
| n | t | t | s | t | s | t | s | t | s |
| 16 MiB | 141.389 | 149.647 | 0.94 | 152.808 | 0.93 | - | - | - | - |
| 64 MiB | 543.064 | 547.167 | 0.99 | 550.855 | 0.98 | - | - | - | - |

D — глубина раскрутки цикла;

n — количество элементов (размер) массива v;

t — время выполнения программы, [msec];

s — ускорение.