

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;

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

$T_{blend\_map\_opt}$  — время выполнения оптимизированной функции `blend_map()`, [msec].

### Intel VTune Amplifier

> Effective time by utilization (red = poor)

19	{				
20	int i = 0;				
21					
22	for (i = 0; i < size; i++) {	14.8%		0.0%	0.0%
23	if (blend == 255) {				16.086ms
24	dest[i] = a[i];				
25	}				
26	else if (blend == 0) {				
27	dest[i] = b[i];				
28	}				
29	else {				
30	dest[i] = a[i] * blend + b[i] * (255 - blend) / 256.0;	85.2%		0.0%	0.0%
31	}				92.536ms
32	}				

Рис. 1 — `blend_map()`

36	{				
37	int i = 0;				
38					
39	if (blend == 255) {				
40	for (i = 0; i < size; ++i) {				
41	dest[i] = a[i];				
42	}				
43	}				
44	else if (blend == 0) {				
45	for (i = 0; i < size; ++i) {				
46	dest[i] = b[i];				
47	}				
48	}				
49	else {				
50	for (i = 0; i < size; ++i) {	33.5%		0.0%	0.0%
51	dest[i] = a[i] * blend + b[i] * (255 - blend) / 256.0;	66.5%		0.0%	0.0%
52	}				31.262ms
53	}				62.050ms
54					
55	}				

Рис. 2 — `blend_map_opt()`

## Задание 2

<i>D</i>	—	2		4		8		16	
<i>n</i>	<i>t</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>
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)

21	}								
22									
23	for (i = 0; i < n; ++i)	21.9%		0.0%	0.0%	114.010ms			
24	v[i] = 1;	24.0%		0.0%	0.0%	124.960ms			
25									
26	s1 = s2 = s3 = s4 = s5 = s6 = s7 = s8 = s9 = s10 = s11 = s12 = s1								
27									
28	startTimer(st);								
29	/* TODO: Unroll this loop */								
30									
31	for (sum = 0, i = 0; i < n; ++i) {	9.0%		0.0%	0.0%	46.951ms			
32	s1 += v[i];	21.0%		0.0%	0.0%	109.431ms			
33	}	6.0%		0.0%	0.0%	31.166ms			
34	stopTimer(st);								
35									

Рис. 3 — до раскрутки

22									
23	for (i = 0; i < n; ++i)	29.2%		0.0%	0.0%	125.055ms			
24	v[i] = 1;	26.9%		0.0%	0.0%	115.153ms			
25									
26	s1 = s2 = s3 = s4 = s5 = s6 = s7 = s8 = s9 = s10 = s11 = s12 = s1								
27									
28	startTimer(st);								
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>

#ifdef 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/howtotothingsin1/a/timing.htm>]

## II

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	<b>1.13</b>
$10^6$	15.285	14.07	<b>1.09</b>

$n$  — 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

83.07%	branch	[kernel.kallsyms]	[k] 0xc11176f0
3.42%	branch	ld-2.19.so	[.] do_lookup_x
3.30%	branch	ld-2.19.so	[.] _dl_check_all_versions
3.15%	branch	ld-2.19.so	[.] _dl_cache_libcmp
2.90%	branch	libc-2.19.so	[.] handle_intel
1.43%	branch	branch	[.] blend_map
1.07%	branch	libc-2.19.so	[.] __strlen_ia32
0.90%	branch	libc-2.19.so	[.] vfprintf
0.76%	branch	libc-2.19.so	[.] hack_digit.13386

blend\_map /home/statiqbloom/Documents/HPCS/hps/branch/branch

```
    movl    $0x0, -0x4(%ebp)
    ↓ jmp    cf
    if (blend == 255) {
33.33 19:  cmpl    $0xff, 0x18(%ebp)
33.33  ↓ jne    49
        dest[i] = a[i];
        mov    -0x4(%ebp), %eax
        lea    0x0(, %eax, 8), %edx
        mov    0x8(%ebp), %eax
        add    %edx, %eax
        mov    -0x4(%ebp), %edx
        lea    0x0(, %edx, 8), %ecx
        mov    0xc(%ebp), %edx
        add    %ecx, %edx
        fldl   (%edx)
        fstpl  (%eax)
    ↓ jmp    cb
    } else if (blend == 0) {
33.33 49:  cmpl    $0x0, 0x18(%ebp)
    ↓ jne    73
        dest[i] = b[i];
        mov    -0x4(%ebp), %eax
        lea    0x0(, %eax, 8), %edx
        mov    0x8(%ebp), %eax
        add    %edx, %eax
        mov    -0x4(%ebp), %edx
        lea    0x0(, %edx, 8), %ecx
        mov    0x10(%ebp), %edx
        add    %ecx, %edx
        fldl   (%edx)
        fstpl  (%eax)
    ↓ jmp    cb
    } else {
73:  mov    -0x4(%ebp), %eax
    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
```

blend\_map\_opt /home/statiqbloom/Documents/HPCS/hps/branch/branch

```
    int i = 0;  
    if (blend == 255) {  
        for (i = 0; i < size; i++) {  
            addl    $0x1, -0x4(%ebp)  
45:    mov     -0x4(%ebp), %eax  
        cmp     0x14(%ebp), %eax  
        jnl     1f  
        jmp     fe  
            dest[i] = a[i];  
        }  
    } else if (blend == 0) {  
52:    cmpl    $0x0, 0x18(%ebp)  
        jne     91  
        for (i = 0; i < size; i++) {  
            movl    $0x0, -0x4(%ebp)  
        jmp     87  
            dest[i] = b[i];  
61:    mov     -0x4(%ebp), %eax  
100.00    lea     0x0(, %eax, 8), %edx  
            mov     0x8(%ebp), %eax  
            add     %edx, %eax  
            mov     -0x4(%ebp), %edx  
            lea     0x0(, %edx, 8), %ecx  
            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)  
87:    mov     -0x4(%ebp), %eax  
            cmp     0x14(%ebp), %eax
```

## Задание 2

<i>D</i>	—	2		4		8		16	
<i>n</i>	<i>t</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>	<i>t</i>	<i>s</i>
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.