Факультет программной инженерии и компьютерной техники

Архитектура компьютера

Лабораторная работа #8

Сепия на C и ASM

Выполнил: Куприянов А.А P33113

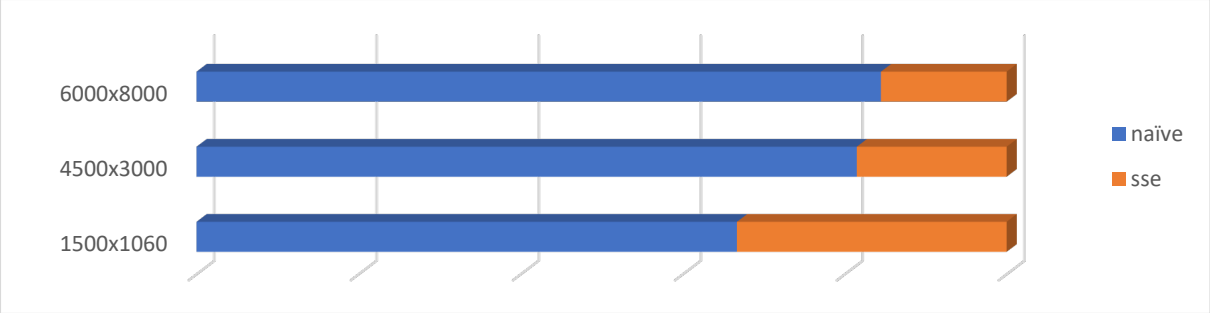
Санкт-Петербург, 2020

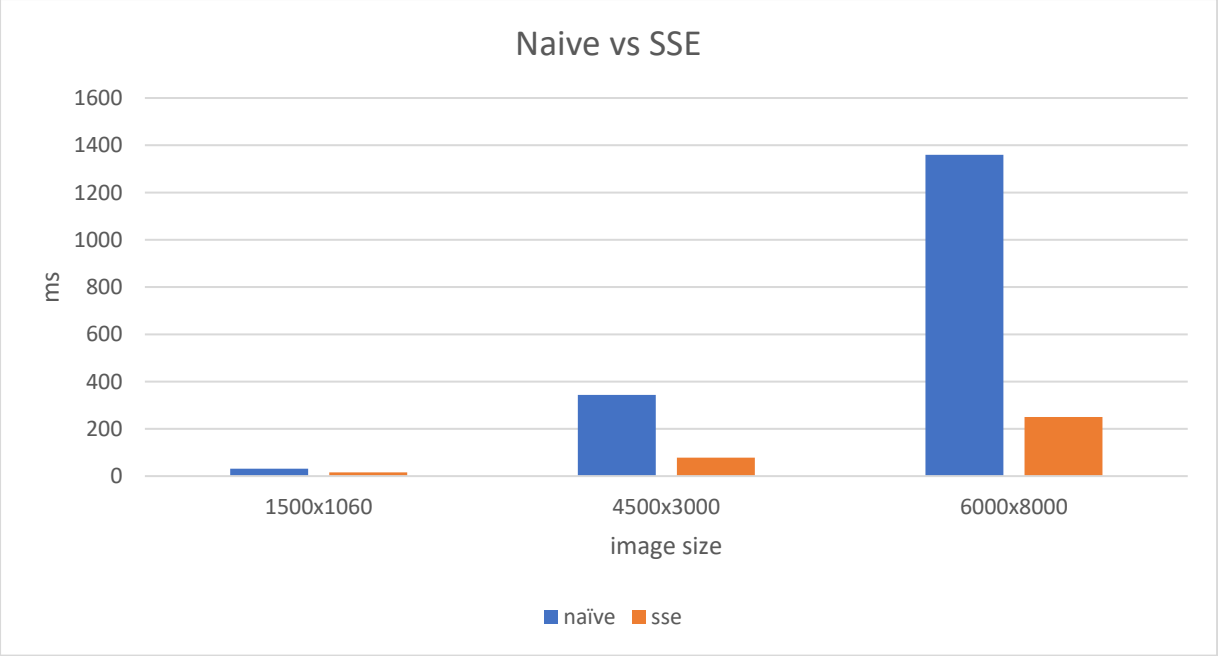
В этой лабораторной работе нужно было сравнить применение сепии на языке C и использованием SSE

```
 1 static unsigned char bounded(uint64_t x) {
 2     if (x < 256) return x;
 3     return 255;
 4 }
 5
 6 /**
 7  * Recolor pixel to sepia-pixel
 8  */
 9 static void sepia_recolor(struct Pixel* pixel) {
10     static const float c[3][3] = {
11             {.272f, .543f, .131f},
12             {.349f, .686f, .168f},
13             {.393f, .769f, .189f}};
14     struct Pixel const old = *pixel;
15     pixel->r = bounded(
16             old.r * c[0][0] + old.g * c[0][1] + old.b * c[0][2]
17     );
18     pixel->g = bounded(
19             old.r * c[1][0] + old.g * c[1][1] + old.b * c[1][2]
20     );
21     pixel->b = bounded(
22             old.r * c[2][0] + old.g * c[2][1] + old.b * c[2][2]
23     );
24 };
```

```
 1 void sse_sepia(struct Image* image) {
 2     float cc[9] = {
                .272f, .543f, .131f,
                .349f, .686f, .168f,
                .393f, .769f, .189f
            };
 3     size_t i;
 4     size_t full_size = image->width * image->height;
 5     size_t sse_num = full_size / 4;
 6     if (sse_num)
 7         // process pixels with sse asm
 8         sepia_asm_recolor(cc, image->imageData, sse_num);
 9     for (i = 4 * sse_num; i < full_size; ++i)
10         // process last few pixels that that didn't fill the last chunk
11         sepia_recolor(&image->imageData[i]);
12 }
```

Streaming SIMD Extensions (SSE) показывают лучший результат в тестах, чем при иммплементации того же алгоритма на си.





Чтобы понять почему мы получаем такие результаты, рассмотрим скомпилированный объектный файл image_filter.c, который применяет сепию на картинку.

Имплементация на Си:

```
00000000000001e3 <sepia>:
 1e3:   55                      push   rbp
 1e4:   48 89 e5                mov    rbp,rsp
 1e7:   48 83 ec 18             sub    rsp,0x18
 1eb:   48 89 7d e8             mov    QWORD PTR [rbp-0x18],rdi
 1ef:   c7 45 f4 00 00 00 00    mov    DWORD PTR [rbp-0xc],0x0
 1f6:   eb 2e                   jmp    226 <sepia+0x43>
 1f8:   48 8b 45 e8             mov    rax,QWORD PTR [rbp-0x18]
 1fc:   48 8b 48 10             mov    rcx,QWORD PTR [rax+0x10]
 200:   8b 45 f4                mov    eax,DWORD PTR [rbp-0xc]
 203:   48 63 d0                movsxd rdx,eax
 206:   48 89 d0                mov    rax,rdx
 209:   48 01 c0                add    rax,rax
 20c:   48 01 d0                add    rax,rdx
 20f:   48 01 c8                add    rax,rcx
 212:   48 89 45 f8             mov    QWORD PTR [rbp-0x8],rax
 216:   48 8b 45 f8             mov    rax,QWORD PTR [rbp-0x8]
 21a:   48 89 c7                mov    rdi,rax
 21d:   e8 fd fd ff ff          call   1f <sepia_recolor>
 222:   83 45 f4 01             add    DWORD PTR [rbp-0xc],0x1
 226:   8b 45 f4                mov    eax,DWORD PTR [rbp-0xc]
 229:   48 63 d0                movsxd rdx,eax
 22c:   48 8b 45 e8             mov    rax,QWORD PTR [rbp-0x18]
 230:   48 8b 48 08             mov    rcx,QWORD PTR [rax+0x8]
 234:   48 8b 45 e8             mov    rax,QWORD PTR [rbp-0x18]
 238:   48 8b 00                mov    rax,QWORD PTR [rax]
 23b:   48 0f af c1             imul   rax,rcx
 23f:   48 39 c2                cmp    rdx,rax
 242:   72 b4                   jb     1f8 <sepia+0x15>
 244:   90                      nop
 245:   c9                      leave
 246:   c3                      ret
```

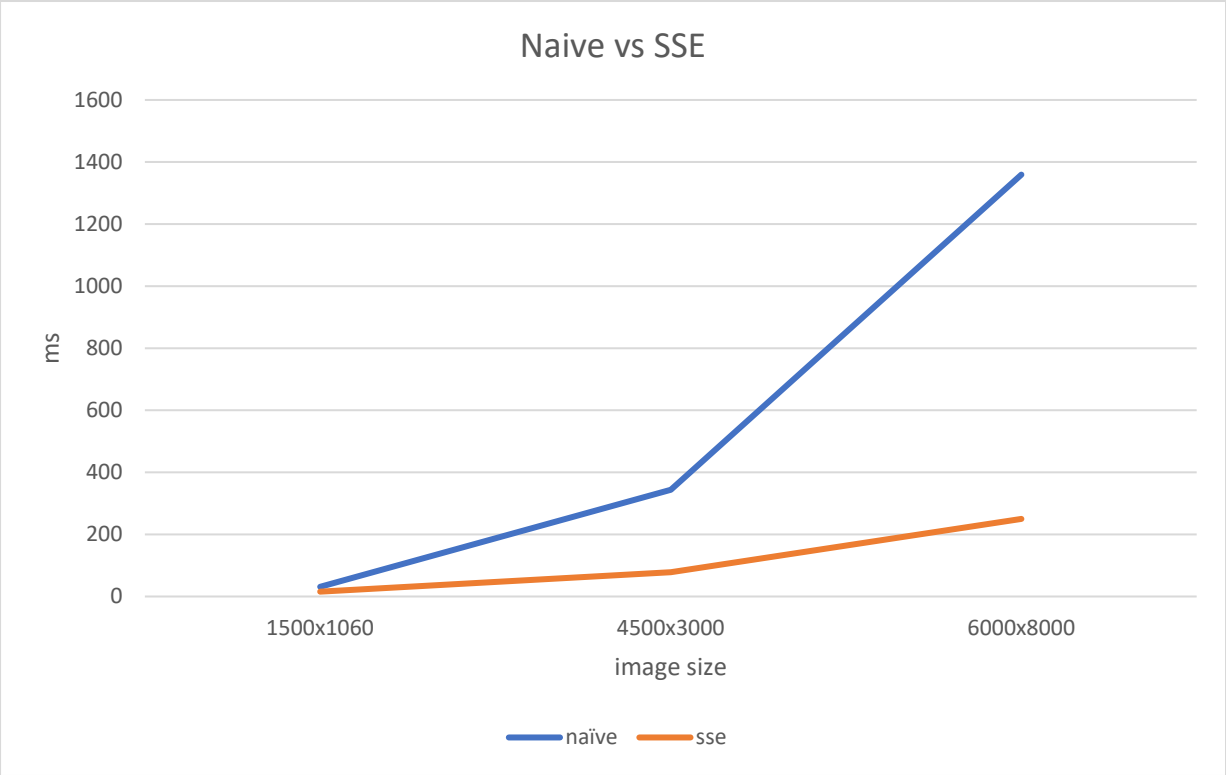
Имплментация на ассемблере (через SSE инструкции):

```
0000000000000247 <sse_sepia>:
 247:   55                      push   rbp
 248:   48 89 e5                mov    rbp,rsp
 24b:   48 83 ec 60             sub    rsp,0x60
 24f:   48 89 7d a8             mov    QWORD PTR [rbp-0x58],rdi
```

```
253:   64 48 8b 04 25 28 00    mov    rax,QWORD PTR fs:0x28
25a:   00 00
25c:   48 89 45 f8             mov    QWORD PTR [rbp-0x8],rax
260:   31 c0                   xor    eax,eax
262:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 26a <sse_sepia+0x23>
269:   00
26a:   f3 0f 11 45 d0          movss  DWORD PTR [rbp-0x30],xmm0
26f:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 277 <sse_sepia+0x30>
276:   00
277:   f3 0f 11 45 d4          movss  DWORD PTR [rbp-0x2c],xmm0
27c:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 284 <sse_sepia+0x3d>
283:   00
284:   f3 0f 11 45 d8          movss  DWORD PTR [rbp-0x28],xmm0
289:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 291 <sse_sepia+0x4a>
290:   00
291:   f3 0f 11 45 dc          movss  DWORD PTR [rbp-0x24],xmm0
296:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 29e <sse_sepia+0x57>
29d:   00
29e:   f3 0f 11 45 e0          movss  DWORD PTR [rbp-0x20],xmm0
2a3:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 2ab <sse_sepia+0x64>
2aa:   00
2ab:   f3 0f 11 45 e4          movss  DWORD PTR [rbp-0x1c],xmm0
2b0:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 2b8 <sse_sepia+0x71>
2b7:   00
2b8:   f3 0f 11 45 e8          movss  DWORD PTR [rbp-0x18],xmm0
2bd:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 2c5 <sse_sepia+0x7e>
2c4:   00
2c5:   f3 0f 11 45 ec          movss  DWORD PTR [rbp-0x14],xmm0
2ca:   f3 0f 10 05 00 00 00    movss  xmm0,DWORD PTR [rip+0x0]        # 2d2 <sse_sepia+0x8b>
2d1:   00
2d2:   f3 0f 11 45 f0          movss  DWORD PTR [rbp-0x10],xmm0
2d7:   48 8b 45 a8             mov    rax,QWORD PTR [rbp-0x58]
2db:   48 8b 10                mov    rdx,QWORD PTR [rax]
2de:   48 8b 45 a8             mov    rax,QWORD PTR [rbp-0x58]
2e2:   48 8b 40 08             mov    rax,QWORD PTR [rax+0x8]
2e6:   48 0f af c2             imul   rax,rdx
2ea:   48 89 45 c0             mov    QWORD PTR [rbp-0x40],rax
2ee:   48 8b 45 c0             mov    rax,QWORD PTR [rbp-0x40]
2f2:   48 c1 e8 02             shr    rax,0x2
2f6:   48 89 45 c8             mov    QWORD PTR [rbp-0x38],rax
2fa:   48 83 7d c8 00          cmp    QWORD PTR [rbp-0x38],0x0
2ff:   74 1b                   je     31c <sse_sepia+0xd5>
301:   48 8b 45 a8             mov    rax,QWORD PTR [rbp-0x58]
305:   48 8b 48 10             mov    rcx,QWORD PTR [rax+0x10]
309:   48 8b 55 c8             mov    rdx,QWORD PTR [rbp-0x38]
30d:   48 8d 45 d0             lea    rax,[rbp-0x30]
311:   48 89 ce                mov    rsi,rcx
314:   48 89 c7                mov    rdi,rax
317:   e8 00 00 00 00          call   31c <sse_sepia+0xd5>
31c:   48 8b 45 c8             mov    rax,QWORD PTR [rbp-0x38]
320:   48 c1 e0 02             shl    rax,0x2
324:   48 89 45 b8             mov    QWORD PTR [rbp-0x48],rax
328:   eb 25                   jmp    34f <sse_sepia+0x108>
32a:   48 8b 45 a8             mov    rax,QWORD PTR [rbp-0x58]
32e:   48 8b 48 10             mov    rcx,QWORD PTR [rax+0x10]
332:   48 8b 55 b8             mov    rdx,QWORD PTR [rbp-0x48]
336:   48 89 d0                mov    rax,rdx
339:   48 01 c0                add    rax,rax
33c:   48 01 d0                add    rax,rdx
33f:   48 01 c8                add    rax,rcx
342:   48 89 c7                mov    rdi,rax
345:   e8 d5 fc ff ff          call   1f <sepia_recolor>
34a:   48 83 45 b8 01          add    QWORD PTR [rbp-0x48],0x1
34f:   48 8b 45 b8             mov    rax,QWORD PTR [rbp-0x48]
353:   48 3b 45 c0             cmp    rax,QWORD PTR [rbp-0x40]
357:   72 d1                   jb     32a <sse_sepia+0xe3>
359:   90                      nop
35a:   48 8b 45 f8             mov    rax,QWORD PTR [rbp-0x8]
35e:   64 48 33 04 25 28 00    xor    rax,QWORD PTR fs:0x28
365:   00 00
367:   74 05                   je     36e <sse_sepia+0x127>
369:   e8 00 00 00 00          call   36e <sse_sepia+0x127>
36e:   c9                      leave
36f:   c3                      ret
```

Как мы видим, второй вариант занимает больше места, но все равно он выполняется быстрее, так как мы используем SSE инструкции, которые поддерживает параллельное выполненение в рамках одной инструкции

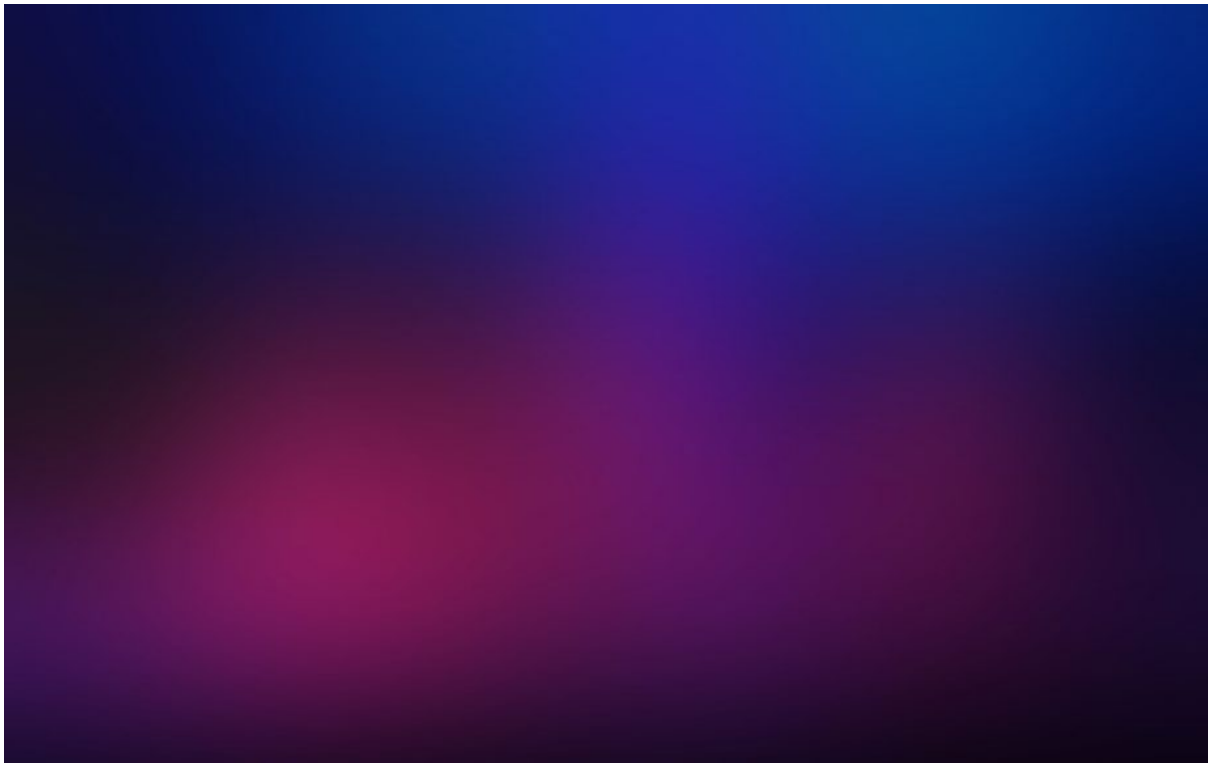
Также по графикам мы видим, что чем больше пикселей нужно обработать, тем больше разрыв между обычном си и SSE инструкциями

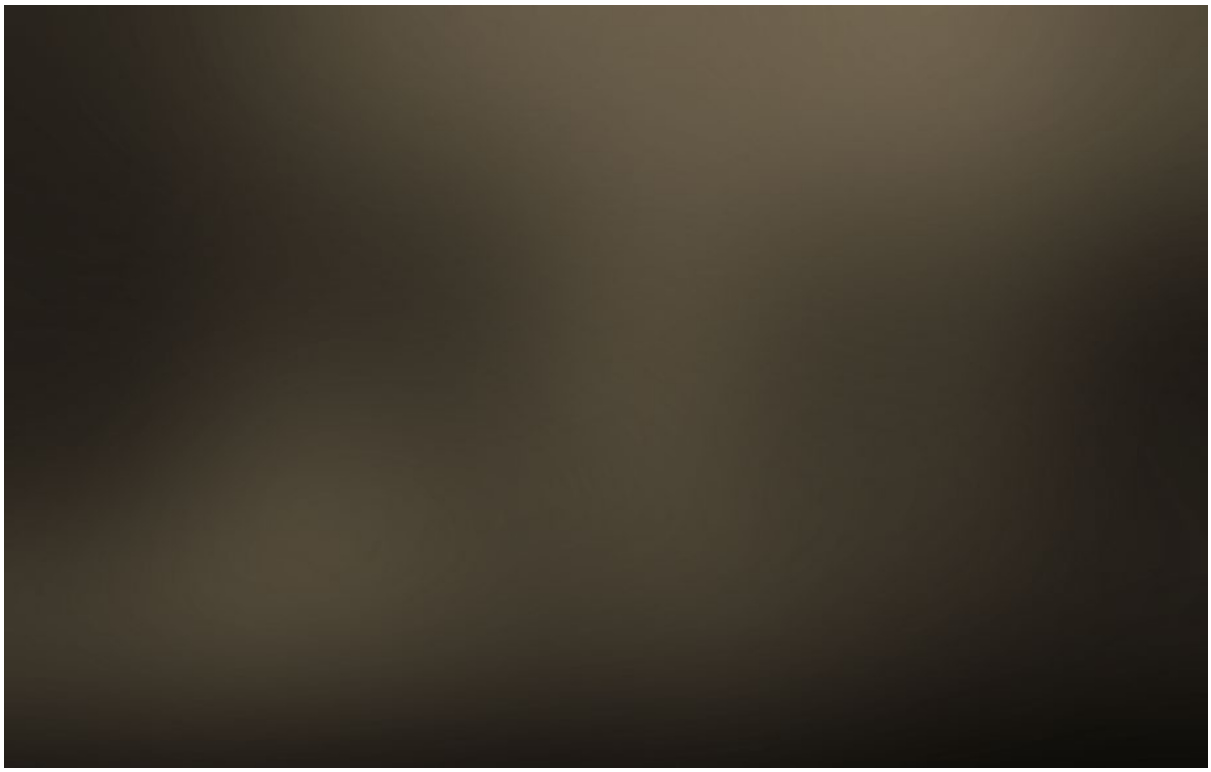


Пример обработки изображений:

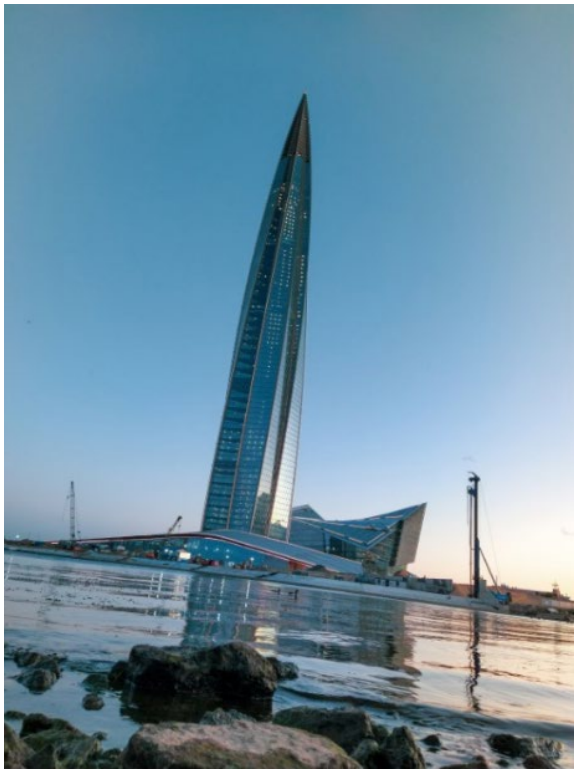
1500x1060

4500x3000





6000x8000



Выходные значения для реализации на языке Си и SSE-инструкции не отличались

Вывод

В этой лабораторной работе я ознакомился с SSE-инструкциями, которые позволяют ускорить время выполнения функции через параллельное исполнение одной инструкции.

На данном примере я сравнил скорость работы с и без них. Во всех случаях SSE был в 4-5 раз быстрее