ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине «Параллельные вычисления»

Выполнил Ковалев Даниил СКБ171, вариант 10 МИЭМ НИУ ВШЭ

Содержание

1	Обі	цее	3
	1.1	Описание вычислительной системы	3
	1.2	Методика измерений	
2	Зад	ача 1.1	3
	2.1	Постановка задачи	3
	2.2	Теоретическая часть	
	2.3		4
	2.4		6
3	Зад	ача 1.2	6
	3.1	Постановка задачи	6
	3.2	Теоретическая часть	
	3.3	Результаты	
	3.4	Выводы	
4	Прі	иложение 12	2
	4.1	Исходные коды	2
	4.2	Ассемблерные листинги	
		4.2.1 Intel ICC	
		4.2.2 Intel LLVM-based compiler	
		4.2.3 GNU GCC	

1 Общее

1.1 Описание вычислительной системы

- Процессор: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
- Количество ядер: 6
- Операционная система: виртуальная машина с Arch Linux
- Оперативная память: 12 ГБ
- Компилятор 1: Intel ICC: icpc (ICC) 2021.4.0 20210910
- Компилятор 2: Intel LLVM-based compiler: Intel(R) oneAPI DPC++/C++ Compiler 2021.4.0 (2021.4.0.20210924)
- Компилятор 3: GNU GCC: g++ (GCC) 11.1.0

1.2 Методика измерений

Получение значения счетчика тактов процессора производились с помощью инструкции rdtsc. Получение значения реального времени производилось с помощью библиотеки std::chrono. Код, реализующий эти операции, см. в приложении.

Для большей репрезентативности и улучшения воспроизводимости результатов частота процессора была заблокирована на номинальном уровне $2.20~\Gamma\Gamma$ ц, а использование технологии Turbo Boost — выключено.

В таблицах с результатами помимо числа тактов и наносекунд, потраченных на выполнение той или иной операции, приведено значение рассчитанной по этим числам частоты процессора.

2 Задача 1.1

Оценка максимальной производительности микропроцессора на заданных операциях (целочисленное деление, векторное целочисленное деление). Код, выполняющий поставленную задачу, см. в приложении.

2.1 Постановка задачи

• Написать программу, выполняющую многократно (в цикле) заданную операцию.

- Замерить время выполнения цикла. По результатам замера получить оценку производительности микропроцессора на заданной операции (в тактах процессора):
 - используя последовательность зависимых операций («латентность»),
 - используя последовательность независимых операций («темп выдачи результатов»).

2.2 Теоретическая часть

Использованная скалярная операция — операция деления двух знаковых 64битных целых чисел (std::int64_t). Соответствует ассемблерной инструкции idiv.

Использованная векторная операция — _mm256_div_epi64 из расширения AVX. Доступна с библиотекой libsvml, поставляющейся с компиляторами Intel. Про-изводит четыре 64-битных целочисленных деления в рамках одной операции.

2.3 Результаты

Intel ICC

Операция	Такты	Наносекунды	ГГц
Независимая скалярная	28.4806	12.8986	2.2080
Зависимая скалярная	42.3657	19.1871	2.2080
Независимая векторная	22.7078	10.2842	2.2080
Зависимая векторная	32.4164	14.6812	2.2080

Intel LLVM-based compiler

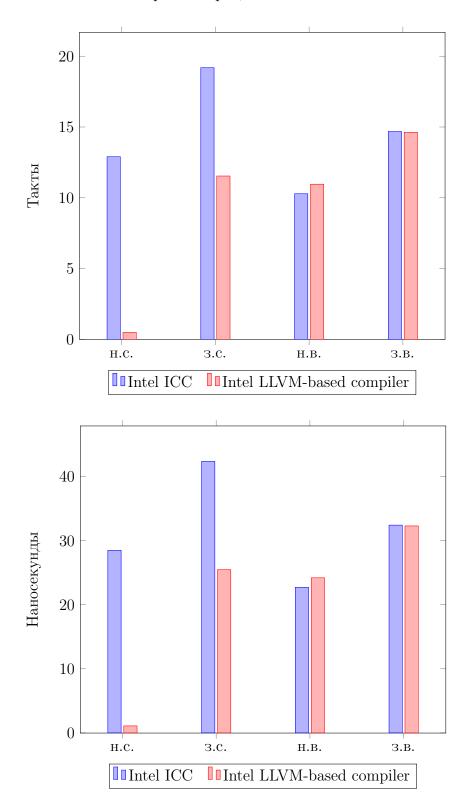
Операция	Такты	Наносекунды	ГГц
Независимая скалярная	1.0880	0.49274	2.2081
Зависимая скалярная	45.6729	20.6851	2.2080
Независимая векторная	24.1993	10.9597	2.2080
Зависимая векторная	32.2940	14.6258	2.2040

GNU GCC — результатов нет, т.к. векторная операция доступна только в библиотеке libsvml, поставляющейся с компиляторами Intel.

В диаграммах используются следующие обозначения:

• н.с. — независимая скалярная операция

- з.с. зависимая скалярная операция
- н.в. независимая векторная операция
- з.в. зависимая векторная операция



2.4 Выводы

- На всех проведенных тестах фактическая частота процессора совпадает с номинальной 2.20 ГГц.
- Аномально малое время выполнения независимой скалярной операции в программе, скомпилированной Intel LLVM-based compiler, можно объяснить при помощи анализа соответствующего ассемблерного листинга (см. приложение). Несмотря на указание не оптимизировать выполнение операции внутри цикла, компилятор вынес ее за его рамки, оставив в теле цикла лишь изменение счетчика. Таким образом, измеренный показатель в 1.0880 такта на операцию итерацию это стоимость изменение счетчика цикла и короткого jump-a.
- За исключением описанной и объясненной выше аномалии, все результаты соответствуют теоретическим предсказаниям: при прочих равных векторные операции быстрее скалярных, независимые быстрее зависимых.

3 Задача 1.2

Освоение векторизации и распараллеливания программ на системе с общей памятью. Производилась оптимизация программы, выполняющей численное интегрирование таблично заданной функции. Код, выполняющий поставленную задачу, см. в приложении.

3.1 Постановка задачи

Оптимизировать заданную программу автоматически или полуавтоматически с помощью компилятора, а также при желании с помощью intrinsics. Сравнить времена работы следующих вариантов программы:

- исходная программа, без оптимизации
- исходная программа, оптимизированная только с помощью ключей компилятора,
- программа, векторизованная полуавтоматически (с помощью директив и ключей компилятора и незначительной правки кода),
- программа, векторизованная и распараллеленная полуавтоматически,
- ..

Во всех случаях использовать ключи оптимизации, дающие наименьшее время работы программы, и наиболее позднее (эффективное) из доступных векторных расширений. По возможности обеспечить использование команд выровненного чтения и записи векторов. В программе с ручной векторизацией минимизировать количество обращений к памяти. На каждом этапе ручной оптимизации проверить, что время работы программы уменьшилось.

3.2 Теоретическая часть

В рамках решения этой задачи использовались следующие флаги и директивы компилятора.

- Флаг -00 не оптимизировать код.
- Флаг -03 использовать 3-й (максимальный) уровень оптимизации.
- Флаг -march=native компилировать код под машину, на которой происходит сборка. Включает возможность использовать специфичные для конкретного процессора инструкции.
- Директива #pragma omp parallel разделить выполнение итераций цикла между потоками.
- Директива **#pragma omp simd** векторизовать цикл.

Было проведены замеры кода со следующими наборами флаг и директив:

- «Наивная» реализация без оптимизаций флаг -00
- «Наивная» реализация с оптимизацией флаг -03.
- «Наивная» реализация с оптимизацией под конкретную машину флаги -03 -march=native.
- Векторизованная реализация флаги -03 -march=native и директива #pragma omp simd.
- Распараллеленная реализация флаги -03 -march=native и директива #pragma omp parallel.

3.3 Результаты

Intel ICC

Интегрирование	Такты	Наносекунды	ГГц
«Наивное» -00	339824176.06	153925526.02	2.2077
«Наивное» -03	11229922.87	5106233.29	2.1993
«Наивное» -03 -march=native	10845158.04	4932022.54	2.1989
Векторизованное	10828490.51	4924846.01	2.1987
Параллельное (1 п-к)	18179023.56	8253084.30	2.2027
Параллельное (2 п-ка)	9589733.79	4361244.32	2.1989
Параллельное (3 п-ка)	6985018.73	3181948.56	2.1952
Параллельное (4 п-ка)	6216703.93	2834112.84	2.1935
Параллельное (5 п-в)	5939075.67	2708988.63	2.1924
Параллельное (6 п-в)	6061977.57	2766919.99	2.1909

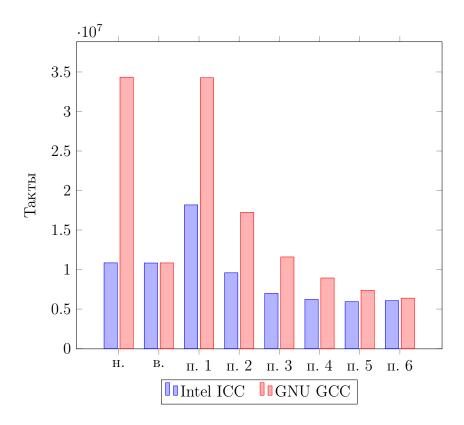
Intel LLVM-based compiler — результатов нет из-за проблем с подключением OpenMP.

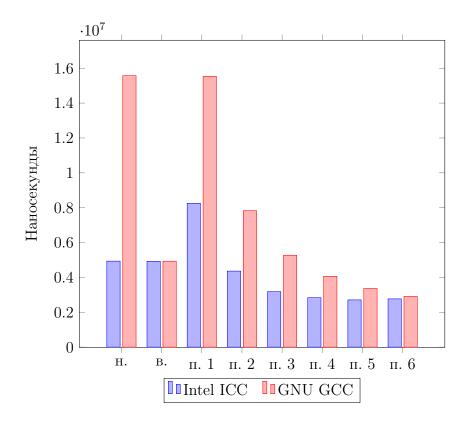
GNU GCC

Интегрирование	Такты	Наносекунды	ГГц
«Наивное» -00	296990216.49	134525747.74	2.2077
«Наивное» -03	34226829.07	15521588.21	2.2051
«Наивное» -03 -march=native	34339403.18	15572950.02	2.2051
Векторизованное	10838969.24	4929447.89	2.1988
Параллельное (1 п-к)	34272412.32	15540118.80	2.2054
Параллельное (2 п-ка)	17236436.64	7825217.52	2.2027
Параллельное (3 п-ка)	11596463.97	5271108.65	2.2000
Параллельное (4 п-ка)	8933118.66	4065504.48	2.1973
Параллельное (5 п-в)	7384450.33	3364648.08	2.1947
Параллельное (6 п-в)	6384912.18	2913110.82	2.1918

В диаграммах используются следующие обозначения:

- н. «наивная» реализация интегрирования, флаги -03 -march=native
- в. векторизованная при помощи OpenMP реализация интегрирования
- \bullet п. x распараллеленная на x потоков при помощи OpenMP реализация интегрирования





Ниже приведены графики зависимости числа тактов процессора, потраченных на выполнение распараллеленной при помощи OpenMP операции интегрирования, от числа потоков. Используются следующие обозначения:

• Теор. 1 — график функции

Такты на «наивное» интегрирование (Intel ICC)

x

• Теор. 2 — график функции

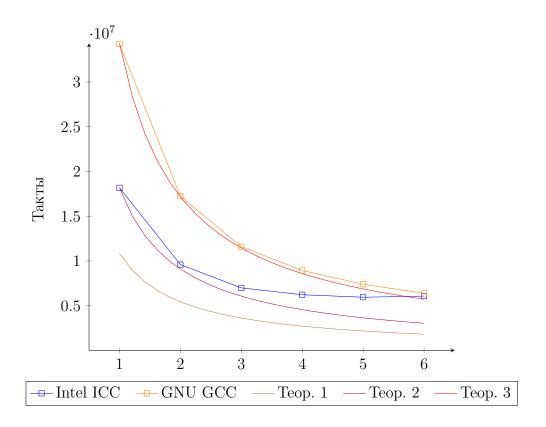
Такты на параллельное интегрирование в 1 поток (Intel ICC)

x

• Теор. 3 — график функции

Такты на «наивное» интегрирование (GNU GCC)

x



Ниже приведены графики зависимости времени (в наносекундах), потраченного на выполнение распараллеленной при помощи OpenMP операции интегрирования, от числа потоков. Используются следующие обозначения:

• Теор. 1 — график функции

Время на «наивное» интегрирование (Intel ICC)

x

• Теор. 2 — график функции

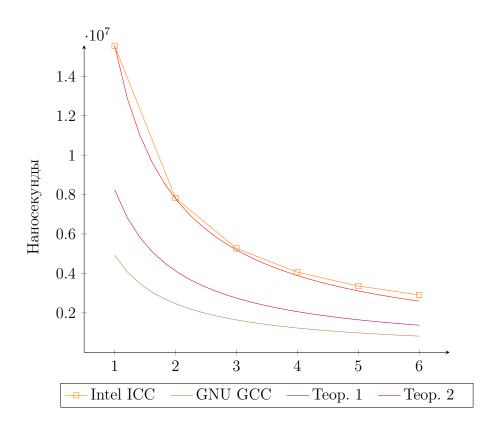
Время на параллельное интегрирование в 1 поток (Intel ICC)

x

• Теор. 3 — график функции

Время на «наивное» интегрирование (GNU GCC)

x



3.4 Выводы

- На всех проведенных тестах фактическая частота процессора совпадает с номинальной 2.20 ГГц.
- Зависимость времени выполнения распараллеленной операции интегрирования от числа потоков близка к обратной пропорциональности.
- Совпадение результатов векторизованной и оптимизированной «наивной» реализаций при компиляции с помощью Intel ICC можно объяснить, проанализировав соответствующие ассемблерные листинги (см. приложение). Выясняется, что компилятор провел векторизацию самостоятельно без явного указания соответствующей директивы.
- Для программы, скомпилированной Intel ICC, параллельное выполнение в 1 поток практически в 2 раза медленнее оптимизированной «наивной» реализации. Анализ соответствующих ассемблерных листингов дает следующее объяснение: при компиляции оптимизированной «наивной» реализации были использованы векторные инструкции, в то время как в распараллеленной реализации используются скалярные.
- За исключением описанных и объясненных выше аномалий, все результаты соответствуют теоретическим предсказаниям: при увеличении уровня оптимизации «наивная» реализация начинает работать быстрее, при прочих равных векторизованное и распараллеленное интегрирование быстрее «наивного», а увеличение числа потоков приводит к обратно пропорцио-

- нальному уменьшению времени работы.
- Код, скомпилированный Intel ICC, на всех тестах (кроме выполнения неоптимизированной «наивной» реализации) показывает более хорошие результаты, чем код, скомпилированный GNU GCC.

4 Приложение

4.1 Исходные коды

CMakeLists.txt

```
cmake minimum required(VERSION 3.0 FATAL ERROR)
  project(task1 LANGUAGES CXX)
   set(CMAKE CXX STANDARD 20)
   set(CMAKE CXX STANDARD REQUIRED ON)
   add library(tools INTERFACE tools.hpp)
   option(BUILD TASK 1 1 "Build task 1.1 (requires Intel compiler)" ON)
10
   option(BUILD_TASK_1_2 "Build task 1.2 (requires OpenMP)"
11
   if(BUILD TASK 1 1)
13
       if(NOT(CMAKE_CXX_COMPILER_ID STREQUAL "Intel" OR CMAKE_CXX_COMPILER_ID
14

    STREQUAL "IntelLLVM")
)
           message(WARNING "Compiler is not ICPC or Intel LLVM, you might face
15
           → problems when compiling")
       endif()
16
       add executable(task1.1 task1.1.cpp)
17
       target link libraries(task1.1 tools)
   endif()
19
20
  if(BUILD TASK 1 2)
21
       find package(OpenMP REQUIRED)
       add executable(task1.2 task1.2.cpp)
23
       target_link_libraries(task1.2 tools)
24
       target_link_libraries(task1.2 OpenMP::OpenMP_CXX)
   endif()
```

tools.hpp

```
#ifndef PARALLEL_COMPUTING_TASK1_HPP_
define PARALLEL_COMPUTING_TASK1_HPP_

#include <chrono>
finclude <cstdint>
finclude <cstdlib>
```

```
namespace my
8
9
10
   template <typename T>
11
   inline __attribute__((always_inline)) void do_not_optimize(T& value)
^{12}
13
   #if defined(__clang__)
14
       asm volatile("" : "+r,m"(value) : : "memory");
15
16
       asm volatile("" : "+m,r"(value) : : "memory");
   #endif
18
19
20
   inline attribute ((always inline)) std::uint64_t ticks()
22
   {
       std::uint64_t tsc;
23
       asm volatile("mfence; "
                                         // memory barrier
^{24}
                     "rdtsc; "
                                          // read of tsc
25
                     "shl $32,%rdx; " // shift higher 32 bits stored in rdx up
26
                     "or %%rdx,%%rax"
                                         // and or onto rax
27
                     : "=a"(tsc)
28
                                         // output to tsc
29
                     : "%rcx", "%rdx", "memory");
30
       return tsc;
31
32
33
   struct BenchmarkResult
34
35
36
       double ticks;
       double nanoseconds;
37
   };
38
39
   class Timer
40
   {
41
   public:
42
       Timer(BenchmarkResult& result, std::size_t iterations_count = 1)
43
            : m_result(result)
44
            , m_iterations_count(iterations_count)
45
       {
46
           m_time_before = std::chrono::high_resolution_clock::now();
           m_ticks_before = ticks();
48
       }
49
50
       ~Timer()
51
       {
           try
53
            {
54
                using namespace std::chrono;
                std::uint64_t ticks after = ticks();
56
                high_resolution_clock::time_point time_after =
57
       high_resolution_clock::now();
```

```
m result.ticks = (ticks after - m ticks before) /
58
       static_cast<double>(m_iterations_count);
               m result.nanoseconds = duration cast<nanoseconds>(time after -
59
       m time before).count() / static cast<double>(m iterations count);
           }
60
           catch (...)
61
62
               std::exit(EXIT FAILURE);
63
64
       }
65
   private:
67
       BenchmarkResult& m result;
68
       std::size_t m iterations count;
69
       std::uint64_t m ticks before;
70
       std::chrono::high resolution clock::time point m time before;
71
   };
72
73
74
   } // namespace my
75
   #endif // PARALLEL_COMPUTING_TASK1_HPP
```

task1.1.cpp

```
#include "tools.hpp"
2
   #include <immintrin.h>
3
4
   #include <cstdint>
5
  #include <cstdlib>
6
  #include <cstring>
  #include <functional>
  #include <iomanip>
  #include <iostream>
   #include <limits>
12
   #include <string>
   #include <string view>
13
   static_assert (sizeof(long long) == sizeof(std::int64_t), "long long and

    std::int64 t do not match");
   static_assert (sizeof( m256i) % sizeof(std::int64_t) == 0, "Size of m256i is
   → not multiple of size of std::int64 t");
   static constexpr std::size_t SCALARS IN VECTOR = sizeof( m256i) /
17

    sizeof(std::int64_t);

18
   static constexpr std::size_t ITERATIONS_COUNT = 10'000'000'000;
19
20
  my::BenchmarkResult independent_scalar_operation(std::int64_t n1, std::int64_t
21
22
       asm volatile("# independent scalar operation enter");
23
       my::BenchmarkResult result;
24
```

```
{
25
           my::Timer timer(result, ITERATIONS COUNT);
26
           for (std::size t i = 0; i < ITERATIONS COUNT; ++i)</pre>
28
                std::int64_t n3 = n1 / n2;
29
                my::do_not_optimize(n3);
30
            }
31
32
       asm volatile("# independent scalar operation exit");
33
       return result;
34
35
36
   my::BenchmarkResult dependent scalar operation(std::int64 t n1, std::int64 t
37
       n2)
   {
38
       asm volatile("# dependent scalar operation enter");
39
       my::BenchmarkResult result;
40
           my::Timer timer(result, ITERATIONS_COUNT);
42
           for (std::size_t i = 0; i < ITERATIONS_COUNT; ++i)</pre>
43
44
            {
                n1 /= n2;
45
46
           my::do_not_optimize(n1);
47
48
       asm volatile("# dependent_scalar_operation exit");
       return result;
50
51
52
   void put_scalar_into_vector(std::int64_t scalar, __m256i& vector)
53
   {
54
       std::int64_t array[SCALARS IN VECTOR];
55
       for (std::size_t i = 0; i < SCALARS IN VECTOR; ++i)</pre>
56
            array[i] = scalar;
57
       std::memcpy(&vector, array, sizeof(vector));
58
59
60
   my::BenchmarkResult independent_vector_operation(std::int64_t n1, std::int64_t
61
   {
62
       asm volatile("# independent_vector_operation enter\n");
63
64
        m256i v1, v2;
65
       put_scalar_into_vector(n1, v1);
66
       put_scalar_into_vector(n2, v2);
67
68
       static constexpr std::size_t vector iterations count = ITERATIONS COUNT /
69
       SCALARS IN VECTOR;
       my::BenchmarkResult result;
70
       {
71
           my::Timer timer(result, ITERATIONS_COUNT);
72
           for (std::size_t i = 0; i < vector_iterations_count; ++i)</pre>
73
            {
```

```
m256i v3 = mm256 div epi64(v1, v2);
75
                 my::do_not_optimize(v3);
76
            }
77
78
        asm volatile("# independent_vector_operation exit");
79
        return result;
80
81
82
   my::BenchmarkResult dependent_vector_operation(std::int64_t n1, std::int64_t
83
    {
84
        asm volatile("# dependent vector operation enter\n");
85
86
         m256i v1, v2;
87
        put scalar into vector(n1, v1);
88
        put scalar into vector(n2, v2);
89
90
        static constexpr std::size_t vector_iterations_count = ITERATIONS_COUNT /
91
        SCALARS_IN_VECTOR;
        my::BenchmarkResult result;
92
93
            my::Timer timer(result, ITERATIONS_COUNT);
94
            for (std::size t i = 0; i < vector iterations count; ++i)</pre>
95
            {
96
                 v1 = _mm256_div_epi64(v1, v2);
97
            }
            my::do not optimize(v1);
99
100
        asm volatile("# dependent_vector_operation exit");
101
        return result;
103
104
   struct Params
105
106
        std::int64 t n1;
107
        std::int64_t n2;
108
109
   };
110
    Params parse_cmd_line(int argc, char* argv[])
111
112
        if (argc != 3)
113
114
            throw std::invalid_argument("Error: you must pass 2 integers as
115
       CLI-arguments");
116
117
        auto int from string = [](const std::string& s) -> std::int64_t
118
        {
119
            try
            {
121
                 std::size_t pos;
122
                 int n = std::stoll(s, &pos);
123
                 if (pos != s.size())
```

```
{
125
                    throw std::invalid argument("stoll");
126
               }
127
               return n;
128
            }
129
            catch (const std::invalid argument& e)
130
131
            {
               throw std::invalid_argument("Error: cannot convert \"" + s + "\" to
132
       std::int64 t: \"" + e.what() + "\"");
           }
133
           catch (const std::out of range& e)
134
135
               throw std::out_of_range("Error: value \"" + s + "\" is out of range
136
       of type std::int64 t: \"" + e.what() + "\"");
137
       };
138
139
       return Params{ .n1 = int_from_string(std::string(argv[1])),
140
                       .n2 = int_from_string(std::string(argv[2])) };
141
142
143
   void print_table_row(std::string_view label, my::BenchmarkResult result)
144
145
       std::cout << "| " << label << " | "
146
                 << std::setw(12) << result.ticks << " | "
147
                 << std::setw(9) << result.nanoseconds << " | " << std::endl</pre>
148
                 << "+-----+" <<
149
       std::endl;
   }
150
151
   int main(int argc, char* argv[]) try
152
153
       Params params = parse cmd line(argc, argv);
154
155
       my::BenchmarkResult independent scalar result =
156
       independent scalar operation(params.n1, params.n2);
       my::BenchmarkResult dependent_scalar_result =
157
       dependent_scalar_operation(params.n1, params.n2);
       my::BenchmarkResult independent vector result =
158
       independent_vector_operation(params.n1, params.n2);
       my::BenchmarkResult dependent_vector_result =
159
       dependent_vector_operation(params.n1, params.n2);
160
       std::cout << "+------" <<
161
       std::endl
                                           | ticks / iter | ns / iter | " <<
                             operation
162
       std::endl
163
       std::endl;
       print_table_row("independent scalar", independent_scalar_result);
164
       print_table_row(" dependent scalar ", dependent_scalar_result);
165
       print_table_row("independent vector", independent_vector_result);
166
       print_table_row(" dependent vector ", dependent_vector_result);
```

```
return EXIT_SUCCESS;
return EXIT_SUCCESS;
return Exit_success;
return exit_exception e)
{
    std::cerr << e.what() << "\nAborting\n";
    return EXIT_FAILURE;
}</pre>
```

task1.2.cpp

```
#include "tools.hpp"
2
  #include <immintrin.h>
  #include <cstdint>
  #include <cstdlib>
   #include <cstring>
   #include <functional>
   #include <iomanip>
  #include <iostream>
  #include <limits>
11
  #include <string>
  #include <string_view>
13
   static_assert (sizeof(long long) == sizeof(std::int64_t), "long long and

    std::int64 t do not match");
   static_assert (sizeof(__m256i) % sizeof(std::int64_t) == 0, "Size of __m256i is
   → not multiple of size of std::int64 t");
   static constexpr std::size_t SCALARS IN VECTOR = sizeof( m256i) /
17

    sizeof(std::int64_t);

18
   static constexpr std::size_t ITERATIONS_COUNT = 10'000'000'000;
19
20
   my::BenchmarkResult independent_scalar_operation(std::int64_t n1, std::int64_t
21
   \hookrightarrow n2)
   {
22
       asm volatile("# independent scalar operation enter");
23
       my::BenchmarkResult result;
24
           my::Timer timer(result, ITERATIONS COUNT);
26
           for (std::size_t i = 0; i < ITERATIONS COUNT; ++i)</pre>
27
           {
28
               std::int64_t n3 = n1 / n2;
29
30
               my::do not optimize(n3);
           }
31
32
       asm volatile("# independent scalar operation exit");
       return result;
34
35
36
  my::BenchmarkResult dependent_scalar_operation(std::int64_t n1, std::int64_t
```

```
38
   {
       asm volatile("# dependent scalar operation enter");
39
       my::BenchmarkResult result;
40
       {
41
           my::Timer timer(result, ITERATIONS_COUNT);
42
            for (std::size_t i = 0; i < ITERATIONS_COUNT; ++i)</pre>
43
            {
44
                n1 /= n2;
45
            }
46
           my::do_not_optimize(n1);
47
       asm volatile("# dependent scalar operation exit");
49
       return result:
50
51
   }
52
   void put scalar into vector(std::int64_t scalar, m256i& vector)
53
54
       std::int64_t array[SCALARS_IN_VECTOR];
55
       for (std::size_t i = 0; i < SCALARS_IN_VECTOR; ++i)</pre>
56
            array[i] = scalar;
57
       std::memcpy(&vector, array, sizeof(vector));
58
59
   }
60
   my::BenchmarkResult independent vector operation(std::int64 t n1, std::int64 t
61
       n2)
   {
62
       asm volatile("# independent vector operation enter\n");
63
64
        __m256i v1, v2;
65
       put_scalar_into_vector(n1, v1);
66
       put scalar into vector(n2, v2);
67
68
       static constexpr std::size_t vector iterations count = ITERATIONS COUNT /
69
       SCALARS IN VECTOR;
       my::BenchmarkResult result;
70
71
           my::Timer timer(result, ITERATIONS_COUNT);
72
            for (std::size_t i = 0; i < vector_iterations_count; ++i)</pre>
73
            {
74
                  _{m256i} v3 = _{mm256} div_{epi64}(v1, v2);
75
                my::do_not_optimize(v3);
76
            }
77
       }
78
       asm volatile("# independent vector operation exit");
79
       return result;
81
82
   my::BenchmarkResult dependent vector operation(std::int64_t n1, std::int64_t
83
       n2)
   {
84
       asm volatile("# dependent vector operation enter\n");
85
86
       __m256i v1, v2;
```

```
put scalar into vector(n1, v1);
88
        put_scalar_into_vector(n2, v2);
89
        static constexpr std::size t vector iterations count = ITERATIONS COUNT /
91
        SCALARS_IN_VECTOR;
        my::BenchmarkResult result;
92
93
            my::Timer timer(result, ITERATIONS COUNT);
94
            for (std::size_t i = 0; i < vector_iterations_count; ++i)</pre>
95
96
                 v1 = _mm256_div_epi64(v1, v2);
            }
98
            my::do not optimize(v1);
99
100
        asm volatile("# dependent vector operation exit");
101
        return result;
102
103
105
    struct Params
106
        std::int64_t n1;
107
108
        std::int64_t n2;
    };
109
110
   Params parse_cmd_line(int argc, char* argv[])
111
112
113
        if (argc != 3)
114
            throw std::invalid_argument("Error: you must pass 2 integers as
115
        CLI-arguments");
        }
116
117
        auto int from string = [](const std::string& s) -> std::int64_t
118
119
        {
            try
120
            {
121
                 std::size_t pos;
122
                 int n = std::stoll(s, &pos);
123
                 if (pos != s.size())
124
                 {
125
                     throw std::invalid_argument("stoll");
126
127
                 return n;
128
129
            catch (const std::invalid argument& e)
130
131
                 throw std::invalid argument("Error: cannot convert \"" + s + "\" to
132
        std::int64 t: \"" + e.what() + "\"");
133
            catch (const std::out of range& e)
134
135
                 throw std::out_of_range("Error: value \"" + s + "\" is out of range
136
        of type std::int64_t: \"" + e.what() + "\"");
```

```
137
           }
       };
138
139
       return Params{ .n1 = int from string(std::string(argv[1])),
140
                      .n2 = int from string(std::string(argv[2])) };
141
142
143
   void print table row(std::string view label, my::BenchmarkResult result)
144
145
       std::cout << "| " << label << " | "
146
                 << std::setw(12) << result.ticks << " | "
                 << std::setw(9) << result.nanoseconds << " | " << std::endl</pre>
148
                 << "+-----+" <<
149
       std::endl;
150
151
   int main(int argc, char* argv[]) try
152
153
154
       Params params = parse_cmd_line(argc, argv);
155
       my::BenchmarkResult independent_scalar_result =
156
       independent_scalar_operation(params.n1, params.n2);
       my::BenchmarkResult dependent scalar result =
157
       dependent scalar operation(params.n1, params.n2);
       my::BenchmarkResult independent_vector_result =
158
       independent_vector_operation(params.n1, params.n2);
       my::BenchmarkResult dependent vector result =
159
       dependent vector operation(params.n1, params.n2);
160
       std::cout << "+-----+" <<
161
       std::endl
                                        | ticks / iter | ns / iter |" <<
                           operation
162
       std::endl
                 << "+-----+" <<
163
       std::endl;
       print_table_row("independent scalar", independent_scalar_result);
164
       print_table_row(" dependent scalar ", dependent_scalar_result);
165
       print_table_row("independent vector", independent_vector_result);
166
       print_table_row(" dependent vector ", dependent_vector_result);
167
168
       return EXIT_SUCCESS;
169
170
   catch (const std::exception& e)
171
172
       std::cerr << e.what() << "\nAborting\n";</pre>
173
       return EXIT FAILURE;
174
   }
175
```

4.2 Ассемблерные листинги

4.2.1 Intel ICC

Скалярное деление (независимые операции)

```
# Begin ASM
   # # independent_scalar_operation enter
2
   # End ASM
                                     # L0E
   ..B1.153:
                                     # Preds ..B1.3
                                     # Execution count [1.00e+00]
6
                      $0, 56(%rsp)
                                                                        #156.93
           movq
7
                                     # L0E
8
   ..B1.4:
                                     # Preds ..B1.153
9
                                     # Execution count [1.00e+00]
10
       __tag_value_main.15:
11
           std::chrono::_V2::system_clock::now() noexcept
12
                      ZNSt6chrono3 V212system clock3nowEv
                                                                        #156.93
13
        tag_value_main.16:
14
                                     # LOE rax
16
   ..B1.5:
                                     # Preds ..B1.4
                                     # Execution count [1.00e+00]
17
                                                                        #156.93
                      %rax, 56(%rsp)
           movq
18
                      $0x174876e800, %r12
                                                                        #156.93
           movq
20
           mfence
                                                                        #156.93
            rdtsc
                                                                        #156.93
21
                      $32, %rdx
                                                                        #156.93
           shlq
22
                                                                        #156.93
                      %rdx, %rax
           orq
                      %rax, 96(%rsp)
           movq
                                                                        #156.93[spill]
24
           xorl
                      %ecx, %ecx
                                                                        #156.93
25
                                     # L0E rcx r12
26
   ..B1.6:
                                     # Preds ..B1.6 ..B1.5
                                     # Execution count [5.56e+00]
28
                                                                        #156.82
           movq
                      160(%rsp), %rax
29
                                                                        #156.93
           incq
                      %rcx
30
            cqto
                                                                        #156.93
31
            idivq
                      168(%rsp)
                                                                        #156.93
32
           movq
                      %rax, (%rsp)
                                                                        #156.93
33
                      %rcx, %r12
                                                                        #156.93
34
            cmpq
                                     # Prob 82%
           ja
                      ..B1.6
                                                                        #156.93
35
                                     # L0E rcx r12
36
                                     # Preds ..B1.6
   ..B1.7:
37
                                     # Execution count [1.00e+00]
           mfence
                                                                        #156.93
39
            rdtsc
                                                                        #156.93
40
                      $32, %rdx
                                                                        #156.93
           shlq
41
                      %rdx, %rax
                                                                        #156.93
42
           orq
43
           movq
                      %rax, %rbx
                                                                        #156.93
        tag value main.18:
44
           std::chrono:: V2::system clock::now() noexcept
45
                      ZNSt6chrono3 V212system clock3nowEv
                                                                       #156.93
46
           call
```

```
..___tag_value_main.19:
                                     # LOE rax rbx r12
   ..B1.155:
                                     # Preds ..B1.7
49
                                     # Execution count [1.00e+00]
50
                      %rax, 64(%rsp)
                                                                       #156.93
           movq
51
                                     # L0E rbx r12
52
   ..B1.8:
                                     # Preds ..B1.155
53
                                     # Execution count [1.00e+00]
54
                                                                       #156.93
           movq
                      64(%rsp), %rax
55
                      %xmm0, %xmm0, %xmm0
                                                                       #156.93
           vxorpd
56
                      56(%rsp), %rax
                                                                       #156.93
           subq
           vcvtsi2sdg %rax, %xmm0, %xmm0
                                                                       #156.93
58
           vdivsd
                      .L 2il0floatpacket.8(%rip), %xmm0, %xmm1
                                                                       #156.93
59
           vmovsd
                      %xmm1, 104(%rsp)
                                                                       #156.93[spill]
60
                                     # L0E rbx r12
   ..B1.159:
                                     # Preds ..B1.8
62
                                     # Execution count [1.00e+00]
63
   # Begin ASM
   # # independent_scalar_operation exit
   # End ASM
```

Скалярное деление (зависимые операции)

```
# Begin ASM
   # # dependent scalar operation enter
   # End ASM
3
                                     # L0E rbx r12
                                     # Preds ..B1.157
   ..B1.156:
5
                                     # Execution count [1.00e+00]
6
                      $0, 24(%rsp)
                                                                       #157.89
           movq
7
                                     # L0E rbx r12
   ..B1.9:
                                     # Preds ..B1.156
9
                                     # Execution count [1.00e+00]
10
        tag_value_main.21:
11
           std::chrono::_V2::system_clock::now() noexcept
12
                      _ZNSt6chrono3_V212system_clock3nowEv
                                                                       #157.89
13
        _tag_value_main.22:
14
                                     # LOE rax rbx r12
15
                                     # Preds ..B1.9
   ..B1.10:
                                     # Execution count [1.00e+00]
17
                                                                       #157.89
           mova
                      %rax, 24(%rsp)
18
                      %esi, %esi
                                                                       #157.89
           xorl
19
                                                                       #157.89
           mfence
20
           rdtsc
                                                                       #157.89
21
                      $32, %rdx
                                                                       #157.89
           shlq
22
                      %rdx, %rax
23
           orq
                                                                       #157.89
24
           movq
                      %rax, 112(%rsp)
                                                                       #157.89[spill]
           movq
                      168(%rsp), %rcx
                                                                       #157.89
25
                      32(%rsp), %rax
                                                                       #157.89
           movq
26
                                     # LOE rax rcx rbx rsi r12
                                     # Preds ..B1.11 ..B1.10
   ..B1.11:
28
                                     # Execution count [5.56e+00]
29
```

```
#157.89
30
            cqto
            incq
                      %rsi
                                                                        #157.89
31
                                                                        #157.89
            idivq
                      %rcx
32
                      %rsi, %r12
            cmpq
                                                                        #157.89
33
                                     # Prob 82%
            ja
                       ..B1.11
                                                                        #157.89
34
                                      # LOE rax rcx rbx rsi r12
35
   ..B1.12:
                                     # Preds ..B1.11
36
                                     # Execution count [1.00e+00]
37
                                                                        #157.89
           movq
                      %rax, 32(%rsp)
38
                                                                        #157.89
           mfence
39
                                                                        #157.89
            rdtsc
                      $32, %rdx
            shlq
                                                                        #157.89
41
                      %rdx, %rax
                                                                        #157.89
            orq
42
                                                                        #157.89
                      %rax, %r14
           movq
43
        tag_value_main.24:
44
           std::chrono:: V2::system clock::now() noexcept
45
            call
                       _ZNSt6chrono3_V212system_clock3nowEv
                                                                        #157.89
46
        _tag_value_main.25:
47
                                     # LOE rax rbx r14
48
   ..B1.161:
                                     # Preds ..B1.12
49
                                     # Execution count [1.00e+00]
50
                                                                        #157.89
51
           movq
                      %rax, 32(%rsp)
                                      # L0E rbx r14
52
   ..B1.13:
                                     # Preds ..B1.161
53
                                     # Execution count [1.00e+00]
54
                      32(%rsp), %rax
                                                                        #157.89
55
           movq
            vxorpd
                      %xmm0, %xmm0, %xmm0
                                                                        #157.89
56
            subq
                      24(%rsp), %rax
                                                                        #157.89
57
            vcvtsi2sdq %rax, %xmm0, %xmm0
                                                                        #157.89
58
                      .L 2il0floatpacket.8(%rip), %xmm0, %xmm1
            vdivsd
                                                                        #157.89
            vmovsd
                      %xmm1, 120(%rsp)
                                                                        #157.89[spill]
60
                                     # LOE rbx r14
61
   ..B1.163:
                                     # Preds ..B1.13
62
                                     # Execution count [1.00e+00]
   # Begin ASM
64
   # # dependent_scalar_operation exit
65
   # End ASM
```

Векторное деление (независимые операции)

```
# Begin ASM
   # # independent vector operation enter
2
   # End ASM
3
                                     # LOE rbx r14
4
   ..B1.162:
                                     # Preds ..B1.163
5
                                     # Execution count [1.00e+00]
6
7
           vpbroadcastq 160(%rsp), %ymm0
                                                                       #158.82
           vmovups
                     %ymm0, 64(%rsp)
                                                                       #158.93
8
                                     # L0E rbx r14
9
   ..B1.14:
                                     # Preds ..B1.162
10
                                     # Execution count [1.00e+00]
11
           vpbroadcastq 168(%rsp), %ymm0
                                                                      #158.93
12
```

```
%ymm0, 32(%rsp)
                                                                        #158.93
13
           vmovups
                                     # LOE rbx r14
14
                                     # Preds ..B1.14
   ..B1.15:
15
                                     # Execution count [1.00e+00]
16
                      $0, 24(%rsp)
                                                                        #158.93
           movq
17
                                     # L0E rbx r14
18
   ..B1.16:
                                     # Preds ..B1.15
19
                                     # Execution count [1.00e+00]
20
           vzeroupper
                                                                        #158.93
21
        tag_value_main.27:
22
            std::chrono::_V2::system_clock::now() noexcept
23
24
                      ZNSt6chrono3 V212system clock3nowEv
                                                                        #158.93
        tag value main.28:
25
                                     # LOE rax rbx r14
26
                                     # Preds ..B1.16
   ..B1.17:
                                     # Execution count [1.00e+00]
28
                      %rax, 24(%rsp)
           movq
                                                                        #158.93
29
           xorl
                      %esi, %esi
                                                                        #158.93
30
                                                                        #158.93
31
           mfence
            rdtsc
                                                                        #158.93
32
                      $32, %rdx
                                                                        #158.93
            shlq
33
                      %rdx, %rax
            orq
                                                                        #158.93
34
           movq
                      $0x5d21dba00, %r15
                                                                        #158.93
35
                      %rax, %r13
                                                                        #158.93
           movq
36
                                     # LOE rbx rsi r13 r14 r15
37
                                     # Preds ..B1.165 ..B1.17
   ..B1.18:
                                     # Execution count [5.56e+00]
39
            vmovdqu
                      64(%rsp), %ymm0
                                                                        #158.93
40
            vmovdqu
                      32(%rsp), %ymm1
                                                                        #158.93
41
        tag value main.29:
            call
                      * svml i64div4@GOTPCREL(%rip)
                                                                        #158.93
43
       _tag_value_main.30:
44
                                     # LOE rbx rsi r13 r14 r15 ymm0
45
   ..B1.165:
                                     # Preds ..B1.18
                                     # Execution count [5.56e+00]
47
                                                                        #158.93
           vmovdqu
                      %ymm0, 128(%rsp)
48
                                                                        #158.93
           incq
                      %rsi
49
            cmpq
                      %rsi, %r15
                                                                        #158.93
50
                      ..B1.18
                                     # Prob 82%
                                                                        #158.93
           ja
51
                                     # LOE rbx rsi r13 r14 r15
52
                                     # Preds ..B1.165
   ..B1.19:
53
                                     # Execution count [1.00e+00]
54
           mfence
                                                                        #158.93
55
            rdtsc
                                                                        #158.93
56
                      $32, %rdx
                                                                        #158.93
            shlq
                      %rdx, %rax
                                                                        #158.93
            orq
58
                      %rax, %r12
                                                                        #158.93
           movq
59
           vzeroupper
                                                                        #158.93
60
        tag value main.31:
61
           std::chrono:: V2::system clock::now() noexcept
62
                      _ZNSt6chrono3_V212system_clock3nowEv
                                                                        #158.93
           call
63
        tag_value_main.32:
64
                                     # LOE rax rbx r12 r13 r14 r15
```

```
..B1.166:
                                     # Preds ..B1.19
66
67
                                     # Execution count [1.00e+00]
                                                                       #158.93
           movq
                      %rax, 64(%rsp)
68
                                     # LOE rbx r12 r13 r14 r15
69
   ..B1.20:
                                     # Preds ..B1.166
70
                                     # Execution count [1.00e+00]
71
                      64(%rsp), %rax
                                                                       #158.93
           movq
72
           vxorpd
                      %xmm0, %xmm0, %xmm0
                                                                       #158.93
73
                      24(%rsp), %rax
                                                                       #158.93
           subq
74
           vcvtsi2sdq %rax, %xmm0, %xmm0
                                                                       #158.93
75
                      .L_2il0floatpacket.8(%rip), %xmm0, %xmm1
           vdivsd
                                                                       #158.93
           vmovsd
                      %xmm1, 128(%rsp)
                                                                       #158.93[spill]
77
                                     # LOE rbx r12 r13 r14 r15
78
   ..B1.168:
                                     # Preds ..B1.20
79
                                     # Execution count [1.00e+00]
   # Begin ASM
81
   # # independent vector operation exit
82
   # End ASM
```

Векторное деление (зависимые операции)

```
# Begin ASM
   # # dependent vector operation enter
   # End ASM
                                     # LOE rbx r12 r13 r14 r15
   ..B1.167:
                                     # Preds ..B1.168
5
                                     # Execution count [1.00e+00]
6
           vpbroadcastq 160(%rsp), %ymm0
                                                                       #159.78
7
           vmovups %ymm0, 32(%rsp)
                                                                       #159.89
8
                                     # LOE rbx r12 r13 r14 r15
9
   ..B1.21:
                                     # Preds ..B1.167
10
                                     # Execution count [1.00e+00]
11
                                                                       #159.89
           vpbroadcastq 168(%rsp),
                                    %ymm0
12
                                                                       #159.89
           vmovups
                      %ymm0, (%rsp)
13
                                     # LOE rbx r12 r13 r14 r15
15
   ..B1.22:
                                     # Preds ..B1.21
                                     # Execution count [1.00e+00]
16
           movq
                      $0, 88(%rsp)
                                                                       #159.89
17
                                     # LOE rbx r12 r13 r14 r15
   ..B1.23:
                                     # Preds ..B1.22
19
                                     # Execution count [1.00e+00]
20
           vzeroupper
                                                                       #159.89
21
        tag value main.34:
22
           std::chrono:: V2::system clock::now() noexcept
23
                      _ZNSt6chrono3_V212system_clock3nowEv
           call
                                                                       #159.89
24
        _tag_value_main.35:
25
                                     # LOE rax rbx r12 r13 r14 r15
   ..B1.24:
                                     # Preds ..B1.23
27
                                     # Execution count [1.00e+00]
28
                      %rax, 88(%rsp)
                                                                       #159.89
           movq
29
                      %esi, %esi
           xorl
                                                                       #159.89
30
           mfence
                                                                       #159.89
31
```

```
rdtsc
                                                                        #159.89
32
            shla
                      $32, %rdx
                                                                        #159.89
33
                      %rdx, %rax
                                                                        #159.89
            orq
                      %rax, 136(%rsp)
           movq
                                                                        #159.89[spill]
35
            vmovdqu
                      32(%rsp), %ymm0
                                                                        #159.89
36
           vmovdqu
                      (%rsp), %ymm8
                                                                        #159.89
37
                                     # LOE rbx rsi r12 r13 r14 r15 ymm0 ymm8
38
   ..B1.25:
                                     # Preds ..B1.170 ..B1.24
39
                                     # Execution count [5.56e+00]
40
                                                                        #159.89
            vmovdqa
                      %ymm8, %ymm1
41
        tag value main.37:
                      * svml i64div4@GOTPCREL(%rip)
            call
                                                                        #159.89
43
        tag value main.38:
44
                                     # LOE rbx rsi r12 r13 r14 r15 ymm0 ymm8
45
                                     # Preds ..B1.25
   ..B1.170:
                                     # Execution count [5.56e+00]
47
            incq
                      %rsi
                                                                        #159.89
48
                      %rsi, %r15
                                                                        #159.89
            cmpq
49
                       ..B1.25
                                     # Prob 82%
                                                                        #159.89
50
            jа
                                     # LOE rbx rsi r12 r13 r14 r15 ymm0 ymm8
51
   ..B1.26:
                                     # Preds ..B1.170
52
                                     # Execution count [1.00e+00]
53
            vmovdqu
                      %ymm0, 32(%rsp)
                                                                        #159.89
54
                                                                        #159.89
           mfence
55
            rdtsc
                                                                        #159.89
56
                      $32, %rdx
                                                                        #159.89
57
            shlq
                      %rdx, %rax
            orq
                                                                        #159.89
58
           movq
                      %rax, %r15
                                                                        #159.89
59
           vzeroupper
                                                                        #159.89
60
        tag_value_main.39:
            std::chrono:: V2::system clock::now() noexcept
62
                      ZNSt6chrono3 V212system clock3nowEv
                                                                        #159.89
63
       tag value main.40:
64
                                     # LOE rax rbx r12 r13 r14 r15
65
                                     # Preds ..B1.26
   ..B1.171:
66
                                     # Execution count [1.00e+00]
67
                                                                        #159.89
                      %rax, (%rsp)
68
           movq
                                     # LOE rbx r12 r13 r14 r15
   ..B1.27:
                                     # Preds ..B1.171
70
                                     # Execution count [1.00e+00]
71
           movq
                      (%rsp), %rax
                                                                        #159.89
72
                      %xmm0, %xmm0, %xmm0
            vxorpd
                                                                        #159.89
73
                      88(%rsp), %rax
                                                                        #159.89
            subq
74
            vcvtsi2sdq %rax, %xmm0, %xmm0
                                                                        #159.89
75
                      .L 2il0floatpacket.8(%rip), %xmm0, %xmm1
           vdivsd
                                                                        #159.89
76
                      %xmm1, 16(%rsp)
                                                                        #159.89[spill]
            vmovsd
77
                                     # LOE rbx r12 r13 r14 r15
78
   ..B1.173:
                                     # Preds ..B1.27
79
                                     # Execution count [1.00e+00]
   # Begin ASM
81
   # # dependent vector operation exit
82
   # End ASM
```

«Наивное» интегрирование

```
# Begin ASM
   # # integrate dummy enter
   # End ASM
                                      # LOE rbx rbp r12 r13 r14 r15
   ..B17.32:
                                      # Preds ..B17.33
5
                                      # Execution count [1.00e+00]
6
           movq
                       (%r12), %rdx
                                                                         #21.25
           movq
                      8(%r12), %rax
                                                                        #21.25
8
                      %xmm0, %xmm0, %xmm0
                                                                        #20.16
            vxorpd
9
                      %xmm0, 8(%rsp)
                                                                        #20.16[spill]
            vmovsd
10
                      %rax, %rdx
                                                                        #21.25
            cmpq
11
            jе
                       ..B17.24
                                      # Prob 50%
                                                                         #21.25
12
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
13
   ..B17.2:
                                      # Preds ..B17.32
14
                                      # Execution count [9.00e-01]
15
            subq
                      %rdx, %rax
                                                                         #21.25
16
                      $7, %rax
                                                                        #21.25
            addq
17
                      $3, %rax
                                                                        #21.25
            shrq
18
                      $16, %rax
                                                                        #21.25
19
            cmpq
            jb
                       ..B17.25
                                      # Prob 10%
                                                                        #21.25
20
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
21
   ..B17.3:
                                      # Preds ..B17.2
22
                                      # Execution count [9.00e-01]
23
                       $29, %rax
24
            cmpq
                                                                        #21.25
                       ..B17.28
                                      # Prob 10%
                                                                        #21.25
            jЬ
25
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
26
   ..B17.4:
                                      # Preds ..B17.3
27
                                      # Execution count [9.00e-01]
28
                      %rdx, %rdi
                                                                         #21.25
           movq
29
            andq
                      $31, %rdi
                                                                        #21.25
30
            jе
                       ..B17.11
                                      # Prob 50%
                                                                        #21.25
31
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
32
   ..B17.5:
                                      # Preds ..B17.4
33
                                      # Execution count [9.00e-01]
34
35
            testq
                       $7, %rdi
                                                                         #21.25
                       ..B17.25
                                      # Prob 10%
                                                                        #21.25
            jne
36
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
37
                                      # Preds ..B17.5
   ..B17.6:
                                      # Execution count [4.50e-01]
39
                      %rdi
                                                                        #21.25
            negg
40
                      $32, %rdi
                                                                        #21.25
            addq
41
                                                                        #21.25
                       $3, %rdi
            shrq
42
            lea
                      16(%rdi), %rcx
                                                                        #21.25
43
                      %rcx, %rax
                                                                        #21.25
            cmpq
44
                       ..B17.25
                                      # Prob 10%
                                                                         #21.25
45
            jЬ
46
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
   ..B17.7:
                                      # Preds ..B17.6
47
                                      # Execution count [9.00e-01]
48
                      %rax, %rcx
                                                                        #21.25
           movq
49
                      %esi, %esi
                                                                        #21.25
            xorl
50
                      %rdi, %rcx
                                                                        #21.25
            subq
51
```

```
andq
                      $15, %rcx
                                                                        #21.25
52
            negq
                      %rcx
                                                                        #21.25
53
                      %rax, %rcx
                                                                        #21.25
            addq
                      %rdi, %rdi
                                                                        #21.25
            testq
55
                       ..B17.12
                                     # Prob 9%
                                                                        #21.25
            jе
56
                                     # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
57

→ xmm0

   ..B17.8:
                                     # Preds ..B17.7
58
                                      # Execution count [9.00e-01]
59
                      %xmm0, %xmm1
            vmovapd
60
                       (%rsp), %xmm2
            vmovsd
                                                                        #[spill]
62
                                     # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
                                      → xmm1 xmm2
   ..B17.9:
                                     # Preds ..B17.9 ..B17.8
63
                                     # Execution count [5.00e+00]
64
            vmulsd
                      (%rdx,%rsi,8), %xmm2, %xmm0
                                                                        #23.9
65
            incq
                      %rsi
                                                                        #21.25
66
            vaddsd
                      %xmm1, %xmm0, %xmm1
                                                                        #23.9
67
                      %rdi, %rsi
                                                                        #21.25
68
            cmpq
            jb
                       ..B17.9
                                     # Prob 82%
                                                                        #21.25
69
                                     # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
70
                                      \hookrightarrow xmm1 xmm2
   ..B17.10:
                                     # Preds ..B17.9
71
                                     # Execution count [9.00e-01]
72
                                                                        #[spill]
                      %xmm1, 8(%rsp)
           vmovsd
73
                                     # Prob 100%
                       ..B17.12
            jmp
                                     # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
75
   ..B17.11:
                                     # Preds ..B17.4
76
                                     # Execution count [4.05e-01]
77
           movq
                      %rax, %rcx
                                                                        #21.25
            andq
                      $15, %rcx
                                                                        #21.25
79
                                                                        #21.25
           negq
                      %rcx
80
                                                                        #21.25
           addq
                      %rax, %rcx
81
                                     # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
                                     # Preds ..B17.11 ..B17.10 ..B17.7 ..B17.28
   ..B17.12:
83
                                     # Execution count [9.00e-01]
84
                      8(%rsp), %xmm1
                                                                        #20.16[spill]
            vmovsd
85
                      %ymm3, %ymm3, %ymm3
            vxorpd
                                                                        #20.16
86
            vxorpd
                      %xmm0, %xmm0, %xmm0
                                                                        #20.16
87
            vmovdqa
                      %ymm3, %ymm2
                                                                        #20.16
88
                      %xmm1, %xmm0, %xmm4
            vmovsd
                                                                        #20.16
            vbroadcastsd (%rsp), %ymm0
                                                                        #17.8[spill]
90
                      %ymm2, %ymm1
            vmovdqa
                                                                        #20.16
91
            vmovaps
                      %xmm4, %xmm4
                                                                        #20.16
92
                                     # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0

→ ymm1 ymm2 ymm3 ymm4

   ..B17.13:
                                     # Preds ..B17.13 ..B17.12
94
                                     # Execution count [5.00e+00]
95
            vfmadd231pd (%rdx,%rdi,8), %ymm0, %ymm4
                                                                        #23.9
96
            vfmadd231pd 32(%rdx,%rdi,8), %ymm0, %ymm3
                                                                        #23.9
97
            vfmadd231pd 64(%rdx,%rdi,8), %ymm0, %ymm2
                                                                        #23.9
98
            vfmadd231pd 96(%rdx,%rdi,8), %ymm0, %ymm1
                                                                        #23.9
99
            addq
                      $16, %rdi
                                                                        #21.25
```

```
%rcx, %rdi
                                                                         #21.25
101
            cmpq
            ib
                        ..B17.13
                                      # Prob 82%
                                                                         #21.25
102
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
103

→ ymm1 ymm2 ymm3 ymm4

                                      # Preds ..B17.13
    ..B17.14:
104
                                      # Execution count [9.00e-01]
105
                       %ymm3, %ymm4, %ymm0
                                                                         #20.16
            vaddpd
106
            vaddpd
                       %ymm1, %ymm2, %ymm1
                                                                         #20.16
107
                       %ymm1, %ymm0, %ymm2
                                                                         #20.16
            vaddpd
108
            vextractf128 $1, %ymm2, %xmm3
                                                                         #20.16
109
                       %xmm3, %xmm2, %xmm4
                                                                         #20.16
110
            vaddpd
            vunpckhpd %xmm4, %xmm4, %xmm5
                                                                         #20.16
111
            vaddsd
                       %xmm5, %xmm4, %xmm6
                                                                         #20.16
112
                       %xmm6, 8(%rsp)
            vmovsd
                                                                         #20.16[spill]
113
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
114
    ..B17.15:
                                      # Preds ..B17.14 ..B17.25
115
                                      # Execution count [1.00e+00]
116
            lea
                       1(%rcx), %rsi
                                                                         #21.25
117
118
            cmpq
                       %rax, %rsi
                                                                         #21.25
            ja
                       ..B17.24
                                      # Prob 50%
                                                                         #21.25
119
                                       # LOE rax rdx rcx rbx rbp r13 r14 r15
120
    ..B17.16:
                                      # Preds ..B17.15
121
                                      # Execution count [9.00e-01]
122
                                                                         #21.25
            subg
                       %rcx, %rax
123
            \mathsf{cmpq}
                       $4, %rax
                                                                         #21.25
124
                        ..B17.27
                                      # Prob 10%
                                                                         #21.25
125
            jb
126
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
    ..B17.17:
                                      # Preds ..B17.16
127
                                      # Execution count [9.00e-01]
128
            vmovsd
                       8(%rsp), %xmm1
                                                                         #20.16[spill]
129
            movq
                       %rax, %rsi
                                                                         #21.25
130
                       %xmm0, %xmm0, %xmm0
                                                                         #20.16
            vxorpd
131
                       (%rdx,%rcx,8), %rdi
                                                                         #21.25
            lea
132
                                                                         #20.16
                       %xmm1, %xmm0, %xmm1
            vmovsd
133
            andq
                       $-4, %rsi
                                                                         #21.25
134
            vbroadcastsd (%rsp), %ymm0
                                                                         #17.8[spill]
135
                       %r8d, %r8d
            xorl
                                                                         #21.25
136
            vmovaps
                       %xmm1, %xmm1
                                                                         #20.16
137
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
138
                                       → r15 ymm0 ymm1
    ..B17.18:
                                      # Preds ..B17.18 ..B17.17
139
                                      # Execution count [5.00e+00]
140
            vmulpd
                       (%rdi,%r8,8), %ymm0, %ymm2
                                                                         #23.9
141
            addq
                       $4, %r8
                                                                         #21.25
142
                                                                         #23.9
            vaddpd
                       %ymm1, %ymm2, %ymm1
143
                       %rsi, %r8
                                                                         #21.25
            cmpq
144
                       ..B17.18
                                      # Prob 82%
                                                                         #21.25
            jb
145
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
146
                                       ..B17.19:
                                      # Preds ..B17.18
147
                                      # Execution count [9.00e-01]
148
            vextractf128 $1, %ymm1, %xmm0
                                                                         #20.16
149
            vaddpd
                      %xmm0, %xmm1, %xmm2
                                                                         #20.16
```

```
vunpckhpd %xmm2, %xmm2, %xmm3
                                                                          #20.16
151
                       %xmm3, %xmm2, %xmm4
             vaddsd
                                                                          #20.16
152
                       %xmm4, 8(%rsp)
             vmovsd
                                                                          #20.16[spill]
153
                                       # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
154
    ..B17.20:
                                       # Preds ..B17.19 ..B17.27
155
                                       # Execution count [1.00e+00]
156
                                                                          #21.25
             cmpq
                       %rax, %rsi
157
                                       # Prob 9%
            jae
                        ..B17.24
                                                                          #21.25
158
                                       # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
159
    ..B17.21:
                                       # Preds ..B17.20
160
                                       # Execution count [9.00e-01]
161
             vmovsd
                       8(%rsp), %xmm1
                                                                          #21.25[spill]
162
             lea
                        (%rdx,%rcx,8), %rdx
                                                                          #21.25
163
                        (%rsp), %xmm2
                                                                          #21.25[spill]
             vmovsd
164
                                       # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
165
    ..B17.22:
                                       # Preds ..B17.22 ..B17.21
166
                                       # Execution count [5.00e+00]
167
                        (%rdx,%rsi,8), %xmm2, %xmm0
             vmulsd
                                                                          #23.9
168
169
             incq
                       %rsi
                                                                          #21.25
             vaddsd
                       %xmm1, %xmm0, %xmm1
                                                                          #23.9
170
                       %rax, %rsi
                                                                          #21.25
             cmpq
171
                                       # Prob 82%
                                                                          #21.25
172
             jЬ
                        ..B17.22
                                       # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
173
    ..B17.23:
                                       # Preds ..B17.22
174
                                       # Execution count [9.00e-01]
175
             vmovsd
                                                                          #[spill]
176
                       %xmm1, 8(%rsp)
                                       # LOE rbx rbp r13 r14 r15
177
    ..B17.24:
                                       # Preds ..B17.23 ..B17.20 ..B17.15 ..B17.32
178
                                       # Execution count [1.00e+00]
179
    # Begin ASM
180
    # # integrate dummy exit
181
    # End ASM
182
```

Векторизованное интегрирование

```
# Begin ASM
   # # integrate omp simd enter
   # End ASM
3
                                     # LOE rbx rbp r12 r13 r14 r15
4
   ..B19.33:
                                     # Preds ..B19.34
5
                                     # Execution count [1.00e+00]
6
                                                                        #47.24
           movq
                      (%r12), %rdx
7
                                                                        #47.24
           movq
                      8(%r12), %rax
8
                      %xmm0, %xmm0, %xmm0
                                                                        #45.16
           vxorpd
9
                                                                        #47.24
                      %rdx, %rax
10
           subq
                      %xmm0, 8(%rsp)
                                                                        #45.16[spill]
11
           vmovsd
12
           sarq
                      $3, %rax
                                                                        #47.24
           jе
                      ..B19.25
                                     # Prob 50%
13
                                     # LOE rax rdx rbx rbp r13 r14 r15 xmm0
14
   ..B19.2:
                                     # Preds ..B19.33
15
                                     # Execution count [5.00e-01]
16
                      ..B19.25
                                     # Prob 50%
                                                                        #47.5
17
           jе
```

```
# LOE rax rdx rbx rbp r13 r14 r15 xmm0
18
   ..B19.3:
                                      # Preds ..B19.2
19
                                      # Execution count [4.50e-01]
20
            cmpq
                       $16, %rax
                                                                         #47.5
21
            jb
                       ..B19.26
                                      # Prob 10%
                                                                         #47.5
22
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
^{23}
   ..B19.4:
                                      # Preds ..B19.3
24
                                      # Execution count [4.50e-01]
25
                       $29, %rax
                                                                         #47.5
            cmpq
26
                       ..B19.29
                                      # Prob 10%
                                                                         #47.5
            jb
27
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
29
   ..B19.5:
                                      # Preds ..B19.4
                                      # Execution count [4.50e-01]
30
                                                                         #47.5
           movq
                       %rdx, %rdi
31
                       $31, %rdi
                                                                         #47.5
            andq
                       ..B19.12
                                      # Prob 50%
                                                                         #47.5
33
            je
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
34
   ..B19.6:
                                      # Preds ..B19.5
35
                                      # Execution count [4.50e-01]
36
            testq
                       $7, %rdi
                                                                         #47.5
37
                       ..B19.26
                                      # Prob 10%
                                                                         #47.5
            jne
38
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
39
   ..B19.7:
                                      # Preds ..B19.6
40
                                      # Execution count [2.25e-01]
41
                       %rdi
                                                                         #47.5
            negq
42
                       $32, %rdi
                                                                         #47.5
            addq
                       $3, %rdi
                                                                         #47.5
            shrq
            lea
                       16(%rdi), %rcx
                                                                         #47.5
45
                       %rcx, %rax
                                                                         #47.5
            cmpq
46
                       ..B19.26
                                      # Prob 10%
                                                                         #47.5
47
            jЬ
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
48
   ..B19.8:
                                      # Preds ..B19.7
49
                                      # Execution count [4.50e-01]
50
                       %rax, %rcx
           movq
                                                                         #47.5
51
                       %esi, %esi
            xorl
                                                                         #47.5
52
                       %rdi, %rcx
                                                                         #47.5
            subq
53
                                                                         #47.5
            andq
                       $15, %rcx
54
            negq
                       %rcx
                                                                         #47.5
55
                       %rax, %rcx
                                                                         #47.5
            addq
56
            testq
                       %rdi, %rdi
                                                                         #47.5
57
                                      # Prob 9%
                       ..B19.13
                                                                         #47.5
            jе
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15

→ xmm0

   ..B19.9:
                                      # Preds ..B19.8
60
                                      # Execution count [4.50e-01]
61
            vmovapd
                       %xmm0, %xmm1
            vmovsd
                       (%rsp), %xmm2
                                                                         #[spill]
63
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
64
                                      \hookrightarrow xmm1 xmm2
   ..B19.10:
                                      # Preds ..B19.10 ..B19.9
65
                                      # Execution count [2.50e+00]
66
            vmulsd
                       (%rdx,%rsi,8), %xmm2, %xmm0
                                                                         #49.9
67
                                                                         #47.5
            incq
                       %rsi
```

```
vaddsd
                       %xmm1, %xmm0, %xmm1
                                                                         #49.9
69
            cmpq
                       %rdi, %rsi
                                                                         #47.5
70
                        ..B19.10
                                                                         #47.5
            jb
                                      # Prob 82%
71
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
72
                                       → xmm1 xmm2
    ..B19.11:
                                      # Preds ..B19.10
73
                                       # Execution count [4.50e-01]
74
            vmovsd
                       %xmm1, 8(%rsp)
                                                                         #[spill]
75
                       ..B19.13
                                      # Prob 100%
            jmp
76
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
77
    ..B19.12:
                                      # Preds ..B19.5
78
                                      # Execution count [2.03e-01]
79
            movq
                       %rax, %rcx
                                                                         #47.5
80
                                                                         #47.5
                       $15, %rcx
            andq
81
                                                                         #47.5
            negq
                       %rcx
82
                       %rax, %rcx
                                                                         #47.5
            addq
83
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
84
    ..B19.13:
                                      # Preds ..B19.12 ..B19.11 ..B19.8 ..B19.29
85
                                      # Execution count [4.50e-01]
86
            vmovsd
                       8(%rsp), %xmm1
                                                                         #45.16[spill]
87
            vxorpd
                       %ymm3, %ymm3, %ymm3
                                                                         #45.16
88
                       %xmm0, %xmm0, %xmm0
                                                                         #45.16
            vxorpd
            vmovdqa
                       %ymm3, %ymm2
                                                                         #45.16
90
                       %xmm1, %xmm0, %xmm4
                                                                         #45.16
            vmovsd
91
            vbroadcastsd (%rsp), %ymm0
                                                                         #42.8[spill]
92
                       %ymm2, %ymm1
                                                                         #45.16
            vmovdqa
                       %xmm4, %xmm4
94
            vmovaps
                                                                         #45.16
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
95

→ ymm1 ymm2 ymm3 ymm4

    ..B19.14:
                                      # Preds ..B19.14 ..B19.13
                                      # Execution count [2.50e+00]
97
                                                                         #49.9
            vfmadd231pd (%rdx,%rdi,8), %ymm0, %ymm4
98
            vfmadd231pd 32(%rdx,%rdi,8), %ymm0, %ymm3
                                                                         #49.9
99
            vfmadd231pd 64(%rdx,%rdi,8), %ymm0, %ymm2
                                                                         #49.9
100
            vfmadd231pd 96(%rdx,%rdi,8), %ymm0, %ymm1
                                                                         #49.9
101
            addq
                       $16, %rdi
                                                                         #47.5
102
                       %rcx, %rdi
                                                                         #47.5
            cmpq
103
            jb
                       ..B19.14
                                      # Prob 82%
                                                                         #47.5
104
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
105

→ ymm1 ymm2 ymm3 ymm4

    ..B19.15:
                                      # Preds ..B19.14
106
                                      # Execution count [4.50e-01]
107
            vaddpd
                       %ymm3, %ymm4, %ymm0
                                                                         #45.16
108
            vaddpd
                       %ymm1, %ymm2, %ymm1
                                                                         #45.16
109
                       %ymm1, %ymm0, %ymm2
            vaddpd
                                                                         #45.16
110
            vextractf128 $1, %ymm2, %xmm3
                                                                         #45.16
111
            vaddpd
                       %xmm3, %xmm2, %xmm4
                                                                         #45.16
112
            vunpckhpd %xmm4, %xmm4, %xmm5
                                                                         #45.16
113
                       %xmm5, %xmm4, %xmm6
            vaddsd
                                                                         #45.16
114
                       %xmm6, 8(%rsp)
            vmovsd
                                                                         #45.16[spill]
115
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
116
    ..B19.16:
                                      # Preds ..B19.15 ..B19.26
117
                                      # Execution count [5.00e-01]
118
```

```
lea
                       1(%rcx), %rsi
                                                                         #47.5
119
            cmpq
                       %rax, %rsi
                                                                         #47.5
120
                       ..B19.25
                                      # Prob 50%
                                                                         #47.5
121
            ja
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
122
    ..B19.17:
                                      # Preds ..B19.16
123
                                      # Execution count [4.50e-01]
124
            subq
                       %rcx, %rax
                                                                         #47.5
125
            cmpq
                       $4, %rax
                                                                         #47.5
126
                       ..B19.28
                                      # Prob 10%
                                                                         #47.5
            jЬ
127
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
128
    ..B19.18:
                                      # Preds ..B19.17
129
                                      # Execution count [4.50e-01]
130
            vmovsd
                       8(%rsp), %xmm1
                                                                         #45.16[spill]
131
                                                                         #47.5
            movq
                       %rax, %rsi
132
                                                                         #45.16
            vxorpd
                       %xmm0, %xmm0, %xmm0
133
                       (%rdx,%rcx,8), %rdi
                                                                         #49.26
            lea
134
            vmovsd
                       %xmm1, %xmm0, %xmm1
                                                                         #45.16
135
                       $-4, %rsi
                                                                         #47.5
            andq
136
137
            vbroadcastsd (%rsp), %ymm0
                                                                         #42.8[spill]
            xorl
                       %r8d, %r8d
                                                                         #47.5
138
                       %xmm1, %xmm1
                                                                         #45.16
            vmovaps
139
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
140
                                       ..B19.19:
                                      # Preds ..B19.19 ..B19.18
141
                                      # Execution count [2.50e+00]
142
            vmulpd
                       (%rdi,%r8,8), %ymm0, %ymm2
                                                                         #49.9
143
144
            addq
                       $4, %r8
                                                                         #47.5
            vaddpd
                       %ymm1, %ymm2,
                                      %ymm1
                                                                         #49.9
145
                       %rsi, %r8
                                                                         #47.5
            cmpq
146
                                                                         #47.5
                       ..B19.19
                                      # Prob 82%
            jЬ
147
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
148
                                       ..B19.20:
                                      # Preds ..B19.19
149
                                      # Execution count [4.50e-01]
150
            vextractf128 $1, %ymm1, %xmm0
                                                                         #45.16
151
            vaddpd
                       %xmm0, %xmm1, %xmm2
                                                                         #45.16
152
            vunpckhpd %xmm2, %xmm2, %xmm3
                                                                         #45.16
153
            vaddsd
                       %xmm3, %xmm2, %xmm4
                                                                         #45.16
154
            vmovsd
                       %xmm4, 8(%rsp)
                                                                         #45.16[spill]
155
                                      # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
156
    ..B19.21:
                                      # Preds ..B19.20 ..B19.28
157
                                      # Execution count [4.50e-01]
158
                       %rax, %rsi
                                                                         #47.5
            cmpq
159
            jae
                       ..B19.25
                                      # Prob 9%
                                                                         #47.5
160
                                      # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
161
    ..B19.22:
                                      # Preds ..B19.21
162
                                      # Execution count [4.50e-01]
163
            vmovsd
                       8(%rsp), %xmm1
                                                                         #49.26[spill]
164
            l ea
                       (%rdx,%rcx,8), %rdx
                                                                         #49.26
165
            vmovsd
                       (%rsp), %xmm2
                                                                         #49.26[spill]
166
                                      # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
167
    ..B19.23:
                                      # Preds ..B19.23 ..B19.22
168
                                      # Execution count [2.50e+00]
169
```

```
vmulsd
                       (%rdx,%rsi,8), %xmm2, %xmm0
                                                                          #49.9
170
                       %rsi
                                                                          #47.5
            incq
171
                       %xmm1, %xmm0, %xmm1
            vaddsd
                                                                          #49.9
172
                       %rax, %rsi
                                                                          #47.5
            cmpq
173
            jb
                       ..B19.23
                                       # Prob 82%
                                                                          #47.5
174
                                       # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
175
    ..B19.24:
                                       # Preds ..B19.23
176
                                       # Execution count [4.50e-01]
177
            vmovsd
                       %xmm1, 8(%rsp)
                                                                          #[spill]
178
                                       # LOE rbx rbp r13 r14 r15
179
                                       # Preds ..B19.24 ..B19.21 ..B19.16 ..B19.2
    ..B19.25:
180
      ..B19.33
181
                                       # Execution count [1.00e+00]
182
    # Begin ASM
183
   # # integrate omp simd exit
184
   # End ASM
185
```

Параллельное интегрирование

```
# Begin ASM
   # # integrate_omp_parallel enter
   # End ASM
                                     # L0E
   ..B18.35:
                                     # Preds ..B18.36
5
                                     # Execution count [1.00e+00]
6
           movl
                      $.2.486_2_kmpc_loc_struct_pack.18, %edi
                                                                       #33.5
7
           movq
                      $0, 72(%rsp)
                                                                       #32.16
8
                        _kmpc_global_thread_num
                                                                       #33.5
           call
9
                                     # LOE eax
10
   ..B18.34:
                                     # Preds ..B18.35
11
                                     # Execution count [1.00e+00]
12
           movl
                      %eax, 80(%rsp)
                                                                       #33.5
13
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
                                                                       #33.5
           movl
14
15
           xorl
                      %eax, %eax
                                                                       #33.5
                    Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.385:
        tag_value
16
            call
                        _kmpc_ok_to_fork
17
                    Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.386:
        tag_value
18
                                     # LOE eax
   ..B18.2:
                                     # Preds ..B18.34
20
                                     # Execution count [1.00e+00]
21
            testl
                      %eax, %eax
                                                                       #33.5
22
                                     # Prob 50%
                                                                       #33.5
            jе
                      ..B18.4
23
                                     # L0E
24
   ..B18.3:
                                     # Preds ..B18.2
25
                                     # Execution count [0.00e+00]
26
27
            addq
                      $-16, %rsp
                                                                       #33.5
            .cfi_def_cfa_offset 128
28
           movl
29
               $L__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_33__par_region0_2.40,
                %edx #33.5
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
           movl
                                                                       #33.5
30
```

```
16(%rsp), %rax
                                                                        #33.5
31
            lea
           movl
                      $4, %esi
                                                                        #33.5
32
            lea
                      72(%rax), %rcx
                                                                        #33.5
            lea
                      56(%rax), %r8
                                                                        #33.5
34
           movq
                      %rax, (%rsp)
                                                                        #33.5
35
                                                                        #33.5
            lea
                      64(%rax), %r9
36
            xorl
                      %eax, %eax
                                                                        #33.5
37
                    Z22integrate omp parallelRKSt6vectorIdSaIdEEd.388:
38
         tag value
            call
                         kmpc_fork_call
39
        tag_value_
                    _Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.389:
40
                                     # L0E
41
   ..B18.37:
                                     # Preds ..B18.3
42
                                     # Execution count [0.00e+00]
43
           addq
                                                                        #33 5
                      $16, %rsp
44
            .cfi def cfa offset 112
45
                      ..B18.7
                                     # Prob 100%
                                                                        #33.5
            jmp
46
                                     # L0E
47
   ..B18.4:
                                     # Preds ..B18.2
48
                                     # Execution count [0.00e+00]
49
           movl
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
                                                                        #33.5
50
                      %eax, %eax
                                                                        #33.5
           xorl
51
           movl
                      80(%rsp), %esi
                                                                        #33.5
52
        tag value
                    Z22integrate omp parallelRKSt6vectorIdSaIdEEd.391:
53
                         kmpc serialized parallel
54
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.392:
55
                                     # L0E
56
                                     # Preds ..B18.4
57
   ..B18.5:
                                     # Execution count [0.00e+00]
58
           movl
59
                    _kmpv_zero_Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_0, %esi
                #33.5
                      80(%rsp), %rdi
            lea
                                                                        #33.5
60
                      -8(%rdi), %rdx
                                                                        #33.5
            lea
61
                      -16(%rdx), %rcx
                                                                        #33.5
            lea
62
            lea
                       -8(%rdx), %r8
                                                                        #33.5
63
            lea
                       (%rsp), %r9
                                                                        #33.5
64
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.393:
65
            call
               L__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_33__par_region0_2.40

→ #33.5

        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.394:
67
                                     # L0E
68
   ..B18.6:
                                     # Preds ..B18.5
69
                                     # Execution count [0.00e+00]
70
           movl
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
                                                                        #33.5
71
           xorl
                      %eax, %eax
                                                                        #33.5
72
           movl
                      80(%rsp), %esi
73
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.395:
74
                         kmpc end serialized parallel
            call
        tag value Z22integrate omp parallelRKSt6vectorIdSaIdEEd.396:
76
                                     # L0E
77
   ..B18.7:
                                     # Preds ..B18.37 ..B18.6
78
                                     # Execution count [1.00e+00]
79
```

```
80  # Begin ASM
81  # # integrate_omp_parallel exit
82  # End ASM
```

4.2.2 Intel LLVM-based compiler

Скалярное деление (независимые операции)

```
#APP
            # independent_scalar_operation enter
2
            #NO APP
3
                           _ZNSt6chrono3_V212system_clock3nowEv
            callq
4
                          %rax, %r14
            movq
5
            #APP
6
            mfence
            rdtsc
8
            shlq
                          $32, %rdx
9
                         %rdx, %rax
            orq
10
            #NO APP
11
                          %rax, %r15
            movq
12
            movq
                         %rbx, %rax
13
                         %r12, %rax
14
            orq
                          $32, %rax
            shrq
15
                        .LBB0 1
16
            jе
   # %bb.2:
17
            movq
                         %rbx, %rax
18
            cqto
19
            idivq
                           %r12
20
                         .LBB0_3
21
            jmp
    .LBB0 1:
22
            movl
                          %ebx, %eax
23
            xorl
                          %edx, %edx
24
            divl
                          %r12d
^{25}
                                                # kill: def $eax killed $eax def $rax
26
27
    .LBB0_3:
            movabsq
                             $10000000000, %rcx
                                                                 \# imm = 0 \times 174876E800
28
            .p2align
                              4, 0x90
29
                                                # =>This Inner Loop Header: Depth=1
    .LBB0 4:
            movq
                          %rax, (%rsp)
31
            #APP
32
            #NO APP
33
                          %rcx
34
            decq
                         .LBB0 4
            jne
35
   # %bb.5:
36
            #APP
37
38
            mfence
            rdtsc
39
            shlq
                          $32, %rdx
40
                         %rdx, %rax
41
            orq
            #NO APP
42
            movq
                         %rax, %rbx
43
```

```
callq
                            ZNSt6chrono3 V212system clock3nowEv
44
            vmovq
                           %rbx, %xmm0
45
                           %rax, %xmm1
46
            vmovq
                                                             \# \times mm0 = \times mm0[0], \times mm1[0]
            vpunpcklqdq
                                  %xmm1, %xmm0, %xmm0
47
                           %r14, %xmm1
            vmovq
48
                           %r15, %xmm2
49
            vmovq
                                  %xmm1, %xmm2, %xmm1
                                                             \# xmm1 = xmm2[0], xmm1[0]
            vpunpcklqdq
50
            vpsubq
                            %xmm1, %xmm0, %xmm0
51
                           %xmm1, %xmm1, %xmm1
            vpxor
52
                              $3, %xmm0, %xmm1, %xmm1
                                                                  # xmm1 =
            vpblendw
53
             \rightarrow xmm0[0,1],xmm1[2,3,4,5,6,7]
            vpor
                          .LCPIO 0(%rip), %xmm1, %xmm1
54
            vpsrlq
                            $32, %xmm0, %xmm2
55
                          .LCPI0 1(%rip), %xmm2, %xmm2
            vpor
56
                            .LCPI0 2(%rip), %xmm2, %xmm2
            vsubpd
57
            vaddpd
                            %xmm2, %xmm1, %xmm1
58
            vpextrq
                             $1, %xmm0, %rax
59
                               %rax, %xmm3, %xmm0
            vcvtsi2sd
60
                              %xmm0, %xmm0
                                                                  \# \times mm0 = \times mm0[0,0]
61
            vmovddup
            vblendpd
                              $1, %xmm1, %xmm0, %xmm0
                                                                  # xmm0 =
62
             \rightarrow xmm1[0],xmm0[1]
63
            vdivpd
                            .LCPI0_3(%rip), %xmm0, %xmm0
            #APP
64
            # independent scalar operation exit
65
            #NO APP
66
```

Скалярное деление (зависимые операции)

```
1
            # dependent_scalar_operation enter
2
            #NO APP
3
            callq
                          ZNSt6chrono3 V212system clock3nowEv
4
           movq
                         %rax, %r14
5
                            $-10000000000, %rsi
                                                              # imm =
           movabsq
6
            → 0xFFFFFE8B7891800
7
            #APP
           mfence
8
            rdtsc
9
                         $32, %rdx
            shlq
10
                        %rdx, %rax
            orq
11
            #NO APP
12
                        %rax, %r15
           movq
13
                        LBB1 1
            jmp
14
            .p2align
                             4, 0x90
15
   .LBB1 24:
                                                  in Loop: Header=BB1 1 Depth=1
16
17
            cqto
18
            idivq
                          %rbx
           movq
                        %rax, %r12
19
                         $8, %rsi
            addq
20
                       .LBB1 26
21
            jе
   .LBB1 1:
                                              # =>This Inner Loop Header: Depth=1
22
                        %r12, %rax
23
           movq
```

```
%rbx, %rax
24
            orq
                          $32, %rax
            shrq
25
                        .LBB1_2
26
            jе
   # %bb.3:
                                                     in Loop: Header=BB1 1 Depth=1
27
            movq
                         %r12, %rax
28
            cqto
29
            idivq
                           %rbx
30
            movq
                         %rax, %rcx
31
                        %rbx, %rcx
            orq
32
                          $32, %rcx
            shrq
33
                        .LBB1_5
            jе
                                                     in Loop: Header=BB1 1 Depth=1
   .LBB1 6:
35
            cqto
36
                           %rbx
            idivq
37
            movq
                         %rax, %rcx
38
                        %rbx, %rcx
            orq
39
                          $32, %rcx
            shrq
40
                        .LBB1_8
41
            jе
   .LBB1_9:
                                                     in Loop: Header=BB1_1 Depth=1
42
            cqto
43
                           %rbx
            idivq
44
45
            movq
                         %rax, %rcx
            orq
                        %rbx, %rcx
46
                          $32, %rcx
            shrq
47
                        .LBB1_11
            jе
48
                                                     in Loop: Header=BB1_1 Depth=1
   .LBB1_12:
49
50
            cqto
            idivq
                           %rbx
51
                         %rax, %rcx
            movq
52
            orq
                        %rbx, %rcx
            shrq
                         $32, %rcx
54
                        .LBB1 14
            jе
55
   .LBB1 15:
                                                     in Loop: Header=BB1 1 Depth=1
56
            cqto
57
            idivq
                           %rbx
58
                         %rax, %rcx
            movq
59
                        %rbx, %rcx
60
            orq
61
            shrq
                          $32, %rcx
                        .LBB1_17
            jе
62
                                                     in Loop: Header=BB1_1 Depth=1
   .LBB1_18:
63
64
            cqto
            idivq
                           %rbx
65
                         %rax, %rcx
            movq
66
                        %rbx, %rcx
            orq
67
                          $32, %rcx
68
            shrq
                        .LBB1 20
            jе
69
   .LBB1 21:
                                                     in Loop: Header=BB1 1 Depth=1
70
            cqto
71
            idivq
                           %rbx
72
            movq
                         %rax, %rcx
73
                        %rbx, %rcx
            orq
74
                         $32, %rcx
            shrq
75
                         .LBB1_24
            jne
```

```
.LBB1 23
77
             jmp
             .p2align
                              4, 0×90
78
    .LBB1 2:
                                                    in Loop: Header=BB1 1 Depth=1
79
                          %r12d, %eax
            movl
80
                          %edx, %edx
             xorl
81
            divl
                          %ebx
82
                                                # kill: def $eax killed $eax def $rax
83
            movq
                          %rax, %rcx
84
                         %rbx, %rcx
             orq
85
             shrq
                          $32, %rcx
86
                         .LBB1 6
             jne
                                                     in Loop: Header=BB1 1 Depth=1
    .LBB1 5:
88
                                                # kill: def $eax killed $eax killed
89

    $rax

             xorl
                          %edx, %edx
             divl
                          %ebx
91
                                                # kill: def $eax killed $eax def $rax
92
                          %rax, %rcx
            movq
93
94
             orq
                         %rbx, %rcx
             shrq
                          $32, %rcx
95
                         .LBB1_9
96
             jne
                                                     in Loop: Header=BB1_1 Depth=1
    .LBB1_8:
97
                                                # kill: def $eax killed $eax killed
98

    $rax

                          %edx, %edx
            xorl
99
             divl
                          %ebx
100
                                                # kill: def $eax killed $eax def $rax
101
            movq
                          %rax, %rcx
102
             orq
                         %rbx, %rcx
103
                          $32, %rcx
104
             shrq
             jne
                         .LBB1 12
105
    .LBB1 11:
                                                     in Loop: Header=BB1 1 Depth=1
106
                                                # kill: def $eax killed $eax killed
107

    $rax

                          %edx, %edx
             xorl
108
             divl
                          %ebx
109
                                                # kill: def $eax killed $eax def $rax
110
111
            movq
                          %rax, %rcx
                         %rbx, %rcx
            orq
112
                          $32, %rcx
             shrq
113
                         .LBB1_15
114
             jne
                                                    in Loop: Header=BB1_1 Depth=1
    .LBB1_14:
115
                                                # kill: def $eax killed $eax killed
116

    $rax

             xorl
                          %edx, %edx
117
             divl
                          %ebx
118
                                                # kill: def $eax killed $eax def $rax
119
                          %rax, %rcx
            movq
120
                         %rbx, %rcx
121
             orq
                          $32, %rcx
             shrq
122
                         .LBB1 18
             jne
123
124 .LBB1 17:
                                                    in Loop: Header=BB1 1 Depth=1
```

```
# kill: def $eax killed $eax killed
125

    $rax

                           %edx, %edx
             xorl
126
             divl
                           %ebx
127
                                                  # kill: def $eax killed $eax def $rax
128
             movq
                           %rax, %rcx
129
                          %rbx, %rcx
             orq
130
             shrq
                           $32, %rcx
131
                          .LBB1 21
             jne
132
    .LBB1 20:
                                                      in Loop: Header=BB1 1 Depth=1
133
                                                  # kill: def $eax killed $eax killed

    $rax

             xorl
                           %edx, %edx
135
             divl
                           %ebx
136
                                                  # kill: def $eax killed $eax def $rax
137
             movq
                           %rax, %rcx
138
                          %rbx, %rcx
             orq
139
                           $32, %rcx
             shrq
140
                          .LBB1_24
141
             jne
    .LBB1 23:
                                                       in Loop: Header=BB1 1 Depth=1
142
                                                  # kill: def $eax killed $eax killed
143

    $rax

             xorl
                           %edx, %edx
144
             divl
                           %ebx
145
                           %eax, %r12d
             movl
146
147
                           $8, %rsi
             addq
                          .LBB1 1
148
             jne
    .LBB1 26:
149
                           %r12, (%rsp)
             movq
150
151
             #APP
             #NO APP
152
             #APP
153
             mfence
154
             rdtsc
155
                           $32, %rdx
             shlq
156
                          %rdx, %rax
             orq
157
             #NO_APP
158
             movq
                           %rax, %rbx
159
             callq
                             _ZNSt6chrono3_V212system_clock3nowEv
160
                            %rbx, %xmm0
             vmovq
161
                            %rax, %xmm1
162
             vmovq
                                                               \# \times mm0 = \times mm0[0], \times mm1[0]
             vpunpcklqdq
                                   %xmm1, %xmm0, %xmm0
163
                            %r14, %xmm1
             vmovq
164
             vmovq
                            %r15, %xmm2
165
             vpunpcklqdq
                                                               \# xmm1 = xmm2[0], xmm1[0]
                                   %xmm1, %xmm2, %xmm1
166
                             %xmm1, %xmm0, %xmm0
             vpsubq
167
                            %xmm1, %xmm1, %xmm1
             vpxor
168
                               $3, %xmm0, %xmm1, %xmm1
             vpblendw
                                                                    \# \times mm1 =
169
              \rightarrow xmm0[0,1],xmm1[2,3,4,5,6,7]
             vpor
                           .LCPI1 0(%rip), %xmm1, %xmm1
170
                             $32, %xmm0, %xmm2
             vpsrlq
171
                           .LCPI1_1(%rip), %xmm2, %xmm2
             vpor
172
                             .LCPI1_2(%rip), %xmm2, %xmm2
             vsubpd
```

```
vaddpd
                             %xmm2, %xmm1, %xmm1
174
             vpextrq
                               $1, %xmm0, %rax
175
                                 %rax, %xmm3, %xmm0
             vcvtsi2sd
176
                                %xmm0, %xmm0
                                                                     \# \times mm0 = \times mm0[0,0]
             vmovddup
177
                                $1, %xmm1, %xmm0, %xmm0
                                                                     # xmm0 =
             vblendpd
178
              \rightarrow xmm1[0],xmm0[1]
                             .LCPI1_3(%rip), %xmm0, %xmm0
             vdivpd
179
             #APP
180
             # dependent_scalar_operation exit
181
             #NO APP
182
```

Векторное деление (независимые операции)

```
#APP
1
            # independent_vector_operation enter
2
3
            #NO_APP
            vmovq
                          %rdi, %xmm0
4
                                  %xmm0, %ymm0
            vpbroadcastq
5
                                                                # 32-byte Spill
            vmovdqa
                            %ymm0, 64(%rsp)
6
                          %rsi, %xmm0
            vmovq
            vpbroadcastq
                                  %xmm0, %ymm0
8
            vmovdqa
                            %ymm0, 32(%rsp)
                                                                # 32-byte Spill
9
            vzeroupper
10
                           ZNSt6chrono3 V212system clock3nowEv
            callq
11
                         %rax, %r14
12
            movq
                            $25000000000, %rsi
                                                                # imm = 0 \times 5D21DBA00
            movabsq
13
            #APP
14
            mfence
15
            rdtsc
16
            shlq
                         $32, %rdx
17
                        %rdx, %rax
18
            orq
            #NO APP
19
            movq
                         %rax, %r15
20
                         __svml_i64div4@GOTPCREL(%rip), %rdi
            movq
^{21}
22
            .p2align
                             4, 0x90
23
   .LBB3 1:
                                               # =>This Inner Loop Header: Depth=1
                            64(%rsp), %ymm0
                                                                # 32-byte Reload
            vmovdqa
24
                                                                # 32-byte Reload
            vmovaps
                            32(%rsp), %ymm1
25
                          *%rdi
26
            callq
            vmovdga
                            %ymm0, (%rsp)
27
            #APP
28
            #NO APP
29
                         %rsi
            decq
30
                        .LBB3 1
            jne
31
   # %bb.2:
32
            #APP
33
34
            mfence
            rdtsc
35
            shlq
                         $32, %rdx
36
                        %rdx, %rax
37
            orq
            #NO APP
38
                         %rax, %rbx
39
            movq
```

```
vzeroupper
40
            callq
                            _ZNSt6chrono3_V212system_clock3nowEv
41
                           %rbx, %xmm0
42
            vmovq
            vmovq
                           %rax, %xmm1
43
                                                              \# \times mm0 = \times mm0[0], \times mm1[0]
            vpunpcklqdq
                                  %xmm1, %xmm0, %xmm0
44
                           %r14, %xmm1
^{45}
            vmovq
            vmovq
                           %r15, %xmm2
46
            vpunpcklqdq
                                  %xmm1, %xmm2, %xmm1
                                                              \# xmm1 = xmm2[0], xmm1[0]
47
            vpsubq
                            %xmm1, %xmm0, %xmm0
48
                           %xmm1, %xmm1, %xmm1
            vpxor
49
                               $3, %xmm0, %xmm1, %xmm1
                                                                   \# \times mm1 =
50
            vpblendw
             \rightarrow xmm0[0,1],xmm1[2,3,4,5,6,7]
                          .LCPI3 0(%rip), %xmm1, %xmm1
            vpor
51
                            $32, %xmm0, %xmm2
            vpsrlq
52
            vpor
                          .LCPI3 1(%rip), %xmm2, %xmm2
53
                            .LCPI3 2(%rip), %xmm2, %xmm2
            vsubpd
54
            vaddpd
                            %xmm2, %xmm1, %xmm1
55
                             $1, %xmm0, %rax
56
            vpextrq
57
            vcvtsi2sd
                                %rax, %xmm3, %xmm0
            vmovddup
                               %xmm0, %xmm0
                                                                   \# \times mm0 = \times mm0[0,0]
58
                               $1, %xmm1, %xmm0, %xmm0
                                                                   # xmm0 =
            vblendpd
59
             \rightarrow xmm1[0],xmm0[1]
            vdivpd
                            .LCPI3 3(%rip), %xmm0, %xmm0
60
            #APP
61
            # independent_vector_operation exit
62
            #NO APP
```

Векторное деление (зависимые операции)

```
#APP
1
2
            # dependent_vector_operation enter
            #NO APP
3
                          %rdi, %xmm0
            vmovq
4
            vpbroadcastq
                                  %xmm0, %ymm0
5
                                                               # 32-byte Spill
6
            vmovdqa
                            %ymm0, 32(%rsp)
                            %ymm0, (%rsp)
7
            vmovdqa
            vmovq
                          %rsi, %xmm0
8
            vpbroadcastq
                                  %xmm0, %ymm0
9
                            %ymm0, 64(%rsp)
                                                               # 32-byte Spill
            vmovdqa
            vzeroupper
11
                          ZNSt6chrono3 V212system clock3nowEv
            callq
12
                            32(%rsp), %ymm0
                                                               # 32-byte Reload
13
            vmovaps
            movq
                         %rax, %r14
14
                            $25000000000, %rsi
                                                               \# imm = 0 \times 5D21DBA00
            movabsq
15
            #APP
16
17
            mfence
18
            rdtsc
            shlq
                         $32, %rdx
19
                        %rdx, %rax
            orq
20
            #NO APP
21
            movq
                         %rax, %r15
22
                         __svml_i64div4@GOTPCREL(%rip), %rdi
            movq
23
```

```
.p2align
                              4, 0x90
24
   .LBB4 1:
                                                 # =>This Inner Loop Header: Depth=1
25
                             64(%rsp), %ymm1
                                                                  # 32-byte Reload
26
            vmovaps
            callq
                           *%rdi
27
            decq
                          %rsi
28
            jne
                         .LBB4_1
29
   # %bb.2:
30
            vmovaps
                             %ymm0, (%rsp)
31
            #APP
32
            #NO APP
33
            #APP
            mfence
35
            rdtsc
36
            shlq
                          $32, %rdx
37
                         %rdx, %rax
            orq
38
            #NO APP
39
            movq
                          %rax, %rbx
40
            vzeroupper
41
                            _ZNSt6chrono3_V212system_clock3nowEv
42
            callq
            vmovq
                           %rbx, %xmm0
43
                           %rax, %xmm1
44
            vmovq
                                  %xmm1, %xmm0, %xmm0
                                                             \# \times mm0 = \times mm0[0], \times mm1[0]
45
            vpunpcklqdq
            vmovq
                           %r14, %xmm1
46
            vmovq
                           %r15, %xmm2
47
                                  %xmm1, %xmm2, %xmm1
                                                             \# xmm1 = xmm2[0], xmm1[0]
            vpunpcklqdq
48
                            %xmm1, %xmm0, %xmm0
            vpsubq
50
            vpxor
                           %xmm1, %xmm1, %xmm1
            vpblendw
                              $3, %xmm0, %xmm1, %xmm1
                                                                   \# \times mm1 =
51
             \rightarrow xmm0[0,1],xmm1[2,3,4,5,6,7]
                          .LCPI4_0(%rip), %xmm1, %xmm1
            vpor
            vpsrlq
                            $32, %xmm0, %xmm2
53
                          .LCPI4 1(%rip), %xmm2, %xmm2
            vpor
54
                            .LCPI4 2(%rip), %xmm2, %xmm2
            vsubpd
55
                            %xmm2, %xmm1, %xmm1
            vaddpd
56
            vpextrq
                             $1, %xmm0, %rax
57
                               %rax, %xmm3, %xmm0
            vcvtsi2sd
58
                              %xmm0, %xmm0
                                                                   \# \times mm0 = \times mm0[0,0]
            vmovddup
59
                                                                   # xmm0 =
            vblendpd
                              $1, %xmm1, %xmm0, %xmm0
             \rightarrow xmm1[0],xmm0[1]
            vdivpd
                            .LCPI4_3(%rip), %xmm0, %xmm0
61
            #APP
62
            # dependent_vector_operation exit
63
            #NO APP
64
```

4.2.3 GNU GCC

«Наивное» интегрирование

```
# Begin ASM
# # integrate_dummy enter
# End ASM
```

```
# LOE rbx rbp r12 r13 r14 r15
   ..B17.32:
                                      # Preds ..B17.33
5
                                      # Execution count [1.00e+00]
6
           movq
                       (%r12), %rdx
                                                                         #21.25
7
           movq
                                                                         #21.25
                       8(%r12), %rax
8
                       %xmm0, %xmm0, %xmm0
                                                                         #20.16
            vxorpd
9
                       %xmm0, 8(%rsp)
                                                                         #20.16[spill]
            vmovsd
10
            cmpq
                       %rax, %rdx
                                                                         #21.25
11
                       ..B17.24
                                      # Prob 50%
                                                                         #21.25
            jе
12
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
13
                                      # Preds ..B17.32
   ..B17.2:
                                      # Execution count [9.00e-01]
15
            subq
                       %rdx, %rax
                                                                         #21.25
16
                       $7, %rax
                                                                         #21.25
            addq
17
                       $3, %rax
                                                                         #21.25
            shrq
18
                       $16, %rax
                                                                         #21.25
            cmpq
19
            jЬ
                       ..B17.25
                                      # Prob 10%
                                                                         #21.25
20
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
^{21}
   ..B17.3:
                                      # Preds ..B17.2
22
                                      # Execution count [9.00e-01]
23
                       $29, %rax
                                                                         #21.25
            cmpq
24
                       ..B17.28
                                      # Prob 10%
                                                                         #21.25
25
            jЬ
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
26
   ..B17.4:
                                      # Preds ..B17.3
27
                                      # Execution count [9.00e-01]
28
                       %rdx, %rdi
                                                                         #21.25
           movq
                       $31, %rdi
                                                                         #21.25
30
            andq
            jе
                       ..B17.11
                                      # Prob 50%
                                                                         #21.25
31
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
32
                                      # Preds ..B17.4
   ..B17.5:
                                      # Execution count [9.00e-01]
34
                       $7, %rdi
            testq
                                                                         #21.25
35
                       ..B17.25
                                      # Prob 10%
                                                                         #21.25
            jne
36
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
37
   ..B17.6:
                                      # Preds ..B17.5
38
                                      # Execution count [4.50e-01]
39
                       %rdi
                                                                         #21.25
40
           negq
            addq
                       $32, %rdi
                                                                         #21.25
41
                       $3, %rdi
                                                                         #21.25
            shrq
42
            lea
                       16(%rdi), %rcx
                                                                         #21.25
43
                       %rcx, %rax
                                                                         #21.25
            cmpq
44
                                      # Prob 10%
            jЬ
                       ..B17.25
                                                                         #21.25
45
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
46
   ..B17.7:
                                      # Preds ..B17.6
47
                                      # Execution count [9.00e-01]
                                                                         #21.25
           movq
                       %rax, %rcx
            xorl
                       %esi, %esi
                                                                         #21.25
50
            subq
                       %rdi, %rcx
                                                                         #21.25
51
                                                                         #21.25
            andq
                       $15, %rcx
52
                                                                         #21.25
            negq
                       %rcx
53
                                                                         #21.25
            addq
                       %rax, %rcx
54
                       %rdi, %rdi
                                                                         #21.25
            testq
55
                                      # Prob 9%
                       ..B17.12
                                                                         #21.25
            jе
```

```
# LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
57

→ xmm0

    ..B17.8:
                                      # Preds ..B17.7
58
                                      # Execution count [9.00e-01]
59
                       %xmm0, %xmm1
            vmovapd
60
                                                                         #[spill]
            vmovsd
                       (%rsp), %xmm2
61
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15

→ xmm1 xmm2

    ..B17.9:
                                      # Preds ..B17.9 ..B17.8
63
                                      # Execution count [5.00e+00]
64
            vmulsd
                       (%rdx,%rsi,8), %xmm2, %xmm0
                                                                         #23.9
65
            incq
                       %rsi
                                                                         #21.25
66
            vaddsd
                       %xmm1, %xmm0, %xmm1
                                                                         #23.9
67
                       %rdi, %rsi
                                                                         #21.25
            cmpq
68
                       ..B17.9
                                      # Prob 82%
                                                                         #21.25
            jb
69
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
70

→ xmm1 xmm2

                                      # Preds ..B17.9
    ..B17.10:
71
                                      # Execution count [9.00e-01]
72
            vmovsd
                       %xmm1, 8(%rsp)
                                                                         #[spill]
73
                       ..B17.12
                                      # Prob 100%
            jmp
74
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
75
    ..B17.11:
                                      # Preds ..B17.4
76
                                      # Execution count [4.05e-01]
77
                                                                         #21.25
            movq
                       %rax, %rcx
78
                                                                         #21.25
                       $15, %rcx
            andq
                                                                         #21.25
80
            negq
                       %rcx
            addq
                       %rax, %rcx
                                                                         #21.25
81
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
82
    ..B17.12:
                                      # Preds ..B17.11 ..B17.10 ..B17.7 ..B17.28
                                      # Execution count [9.00e-01]
84
            vmovsd
                                                                         #20.16[spill]
                       8(%rsp), %xmm1
85
            vxorpd
                       %ymm3, %ymm3, %ymm3
                                                                         #20.16
86
                       %xmm0, %xmm0, %xmm0
            vxorpd
                                                                         #20.16
87
                       %ymm3, %ymm2
            vmovdga
                                                                         #20.16
88
                       %xmm1, %xmm0, %xmm4
            vmovsd
                                                                         #20.16
89
            vbroadcastsd (%rsp), %ymm0
                                                                         #17.8[spill]
90
            vmovdqa
                       %ymm2, %ymm1
                                                                         #20.16
            vmovaps
                       %xmm4, %xmm4
                                                                         #20.16
92
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
93
                                      \hookrightarrow ymm1 ymm2 ymm3 ymm4
    ..B17.13:
                                      # Preds ..B17.13 ..B17.12
94
                                      # Execution count [5.00e+00]
95
            vfmadd231pd (%rdx,%rdi,8), %ymm0, %ymm4
                                                                         #23.9
96
            vfmadd231pd 32(%rdx,%rdi,8), %ymm0, %ymm3
                                                                         #23.9
97
            vfmadd231pd 64(%rdx,%rdi,8), %ymm0, %ymm2
                                                                         #23.9
            vfmadd231pd 96(%rdx,%rdi,8), %ymm0, %ymm1
                                                                         #23.9
99
            addq
                       $16, %rdi
                                                                         #21.25
100
                       %rcx, %rdi
                                                                         #21.25
            cmpq
101
                       ..B17.13
                                      # Prob 82%
            jЬ
                                                                         #21.25
102
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
103

→ ymm1 ymm2 ymm3 ymm4

104 ...B17.14:
                                      # Preds ..B17.13
```

```
# Execution count [9.00e-01]
105
            vaddpd
                       %ymm3, %ymm4, %ymm0
                                                                         #20.16
106
                       %ymm1, %ymm2, %ymm1
                                                                         #20.16
            vaddpd
107
                       %ymm1, %ymm0, %ymm2
            vaddpd
                                                                         #20.16
108
            vextractf128 $1, %ymm2, %xmm3
                                                                         #20.16
109
            vaddpd
                       %xmm3, %xmm2, %xmm4
                                                                         #20.16
110
            vunpckhpd %xmm4, %xmm4, %xmm5
                                                                         #20.16
111
            vaddsd
                       %xmm5, %xmm4, %xmm6
                                                                         #20.16
112
                       %xmm6, 8(%rsp)
            vmovsd
                                                                         #20.16[spill]
113
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
114
    ..B17.15:
                                      # Preds ..B17.14 ..B17.25
115
                                      # Execution count [1.00e+00]
116
            lea
                       1(%rcx), %rsi
                                                                         #21.25
117
                                                                         #21.25
                       %rax, %rsi
            cmpq
118
                       ..B17.24
                                      # Prob 50%
                                                                         #21.25
            ja
119
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
120
    ..B17.16:
                                      # Preds ..B17.15
121
                                      # Execution count [9.00e-01]
^{122}
                       %rcx, %rax
123
            subq
                                                                         #21.25
            cmpq
                       $4, %rax
                                                                         #21.25
124
            jb
                        ..B17.27
                                      # Prob 10%
                                                                         #21.25
125
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
126
    ..B17.17:
                                      # Preds ..B17.16
127
                                      # Execution count [9.00e-01]
128
            vmovsd
                       8(%rsp), %xmm1
                                                                         #20.16[spill]
129
                       %rax, %rsi
                                                                         #21.25
130
            movq
                       %xmm0, %xmm0, %xmm0
                                                                         #20.16
131
            vxorpd
            lea
                       (%rdx,%rcx,8), %rdi
                                                                         #21.25
132
                       %xmm1, %xmm0, %xmm1
                                                                         #20.16
            vmovsd
133
                                                                         #21.25
            andq
                       $-4, %rsi
            vbroadcastsd (%rsp), %ymm0
                                                                         #17.8[spill]
135
                       %r8d, %r8d
            xorl
                                                                         #21.25
136
                       %xmm1, %xmm1
                                                                         #20.16
            vmovaps
137
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
138
                                       ..B17.18:
                                      # Preds ..B17.18 ..B17.17
139
                                      # Execution count [5.00e+00]
140
            vmulpd
                       (%rdi,%r8,8), %ymm0, %ymm2
                                                                         #23.9
141
            addq
                       $4, %r8
                                                                         #21.25
142
            vaddpd
                       %ymm1, %ymm2, %ymm1
                                                                         #23.9
143
                       %rsi, %r8
            cmpq
                                                                         #21.25
144
                        ..B17.18
            jЬ
                                      # Prob 82%
                                                                         #21.25
145
                                      # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
146
                                       → r15 ymm0 ymm1
    ..B17.19:
                                      # Preds ..B17.18
147
                                      # Execution count [9.00e-01]
148
            vextractf128 $1, %ymm1, %xmm0
                                                                         #20.16
149
            vaddpd
                       %xmm0, %xmm1, %xmm2
                                                                         #20.16
150
            vunpckhpd %xmm2, %xmm2, %xmm3
                                                                         #20.16
151
                       %xmm3, %xmm2, %xmm4
            vaddsd
                                                                         #20.16
152
                       %xmm4, 8(%rsp)
            vmovsd
                                                                         #20.16[spill]
153
                                       # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
154
                                      # Preds ..B17.19 ..B17.27
155 ...B17.20:
```

```
# Execution count [1.00e+00]
156
                       %rax, %rsi
            cmpq
                                                                         #21.25
157
                        ..B17.24
                                      # Prob 9%
                                                                         #21.25
158
            jae
                                      # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
159
    ..B17.21:
                                      # Preds ..B17.20
160
                                      # Execution count [9.00e-01]
161
            vmovsd
                       8(%rsp), %xmm1
                                                                         #21.25[spill]
162
            lea
                       (%rdx,%rcx,8), %rdx
                                                                         #21.25
163
                       (%rsp), %xmm2
                                                                         #21.25[spill]
            vmovsd
164
                                      # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
165
    ..B17.22:
                                      # Preds ..B17.22 ..B17.21
166
                                      # Execution count [5.00e+00]
167
            vmulsd
                       (%rdx,%rsi,8), %xmm2, %xmm0
                                                                         #23.9
168
                                                                         #21.25
                       %rsi
            incq
169
            vaddsd
                       %xmm1, %xmm0, %xmm1
                                                                         #23.9
170
                       %rax, %rsi
                                                                         #21.25
            cmpq
171
            jb
                       ..B17.22
                                      # Prob 82%
                                                                         #21.25
172
                                      # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
173
    ..B17.23:
                                      # Preds ..B17.22
174
                                      # Execution count [9.00e-01]
175
                       %xmm1, 8(%rsp)
            vmovsd
                                                                         #[spill]
176
                                       # LOE rbx rbp r13 r14 r15
177
    ..B17.24:
                                      # Preds ..B17.23 ..B17.20 ..B17.15 ..B17.32
178
                                      # Execution count [1.00e+00]
179
    # Begin ASM
180
    # # integrate dummy exit
    # End ASM
```

Векторизованное интегрирование

```
# Begin ASM
   # # integrate omp simd enter
   # End ASM
3
                                     # LOE rbx rbp r12 r13 r14 r15
                                     # Preds ..B19.34
5
   ..B19.33:
6
                                      # Execution count [1.00e+00]
                                                                        #47.24
           movq
                       (%r12), %rdx
7
           movq
                      8(%r12), %rax
                                                                        #47.24
8
                      %xmm0, %xmm0, %xmm0
                                                                        #45.16
            vxorpd
            subq
                      %rdx, %rax
                                                                        #47.24
10
                      %xmm0, 8(%rsp)
            vmovsd
                                                                        #45.16[spill]
11
                                                                        #47.24
                      $3, %rax
            sarq
12
                                                                        #47.5
                       ..B19.25
                                     # Prob 50%
            jе
13
                                     # LOE rax rdx rbx rbp r13 r14 r15 xmm0
14
   ..B19.2:
                                     # Preds ..B19.33
15
                                     # Execution count [5.00e-01]
16
17
           jе
                       ..B19.25
                                     # Prob 50%
                                                                        #47.5
                                     # LOE rax rdx rbx rbp r13 r14 r15 xmm0
18
   ..B19.3:
                                     # Preds ..B19.2
19
                                     # Execution count [4.50e-01]
20
            cmpq
                      $16, %rax
                                                                        #47.5
21
                       ..B19.26
                                     # Prob 10%
                                                                        #47.5
22
            jb
```

```
# LOE rax rdx rbx rbp r13 r14 r15 xmm0
23
   ..B19.4:
                                      # Preds ..B19.3
24
                                      # Execution count [4.50e-01]
25
            cmpq
                       $29, %rax
                                                                         #47.5
26
            jb
                       ..B19.29
                                      # Prob 10%
                                                                        #47.5
27
                                      # LOE rax rdx rbx rbp r13 r14 r15 xmm0
28
   ..B19.5:
                                      # Preds ..B19.4
29
                                      # Execution count [4.50e-01]
30
                      %rdx, %rdi
                                                                        #47.5
           movq
31
                      $31, %rdi
                                                                        #47.5
           andq
32
                       ..B19.12
                                      # Prob 50%
                                                                        #47.5
            jе
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
34
   ..B19.6:
                                      # Preds ..B19.5
35
                                      # Execution count [4.50e-01]
36
                       $7, %rdi
                                                                        #47.5
            testq
37
                       ..B19.26
                                      # Prob 10%
                                                                        #47.5
            ine
38
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
39
   ..B19.7:
                                      # Preds ..B19.6
40
                                      # Execution count [2.25e-01]
41
            negq
                       %rdi
                                                                        #47.5
42
            addq
                      $32, %rdi
                                                                        #47.5
43
                                                                        #47.5
            shrq
                      $3, %rdi
44
            lea
                      16(%rdi), %rcx
                                                                        #47.5
45
                      %rcx, %rax
                                                                        #47.5
            cmpq
46
                       ..B19.26
                                      # Prob 10%
                                                                        #47.5
            jЬ
47
                                      # LOE rax rdx rbx rbp rdi r13 r14 r15 xmm0
48
                                      # Preds ..B19.7
   ..B19.8:
                                      # Execution count [4.50e-01]
50
                                                                        #47.5
           movq
                      %rax, %rcx
51
                                                                        #47.5
            xorl
                      %esi, %esi
            subq
                      %rdi, %rcx
                                                                        #47.5
53
                      $15, %rcx
                                                                        #47.5
            andq
54
                                                                        #47.5
            negq
                      %rcx
55
                      %rax, %rcx
                                                                        #47.5
            addq
56
                       %rdi, %rdi
            testq
                                                                        #47.5
57
                       ..B19.13
                                      # Prob 9%
            jе
58
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
59

→ xmm0

   ..B19.9:
                                      # Preds ..B19.8
60
                                      # Execution count [4.50e-01]
61
                      %xmm0, %xmm1
            vmovapd
62
            vmovsd
                       (%rsp), %xmm2
                                                                        #[spill]
63
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
64
                                      → xmm1 xmm2
   ..B19.10:
                                      # Preds ..B19.10 ..B19.9
                                      # Execution count [2.50e+00]
            vmulsd
                       (%rdx,%rsi,8), %xmm2, %xmm0
                                                                        #49.9
67
            incq
                      %rsi
                                                                        #47.5
68
            vaddsd
                       %xmm1, %xmm0, %xmm1
                                                                        #49.9
69
                      %rdi, %rsi
                                                                        #47.5
            cmpq
70
                       ..B19.10
                                      # Prob 82%
            jЬ
71
                                      # LOE rax rdx rcx rbx rbp rsi rdi r13 r14 r15
72

→ xmm1 xmm2
```

```
..B19.11:
                                      # Preds ..B19.10
                                      # Execution count [4.50e-01]
74
                       %xmm1, 8(%rsp)
            vmovsd
                                                                         #[spill]
75
            jmp
                       ..B19.13
                                      # Prob 100%
76
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
77
    ..B19.12:
                                      # Preds ..B19.5
78
                                      # Execution count [2.03e-01]
79
            movq
                       %rax, %rcx
                                                                         #47.5
80
                       $15, %rcx
                                                                         #47.5
            andq
81
                                                                         #47.5
                       %rcx
            negq
82
                                                                         #47.5
            addq
                       %rax, %rcx
83
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15
84
    ..B19.13:
                                      # Preds ..B19.12 ..B19.11 ..B19.8 ..B19.29
85
                                      # Execution count [4.50e-01]
86
            vmovsd
                       8(%rsp), %xmm1
                                                                         #45.16[spill]
87
            vxorpd
                       %ymm3, %ymm3, %ymm3
                                                                         #45.16
88
                       %xmm0, %xmm0, %xmm0
            vxorpd
                                                                         #45.16
89
                       %ymm3, %ymm2
            vmovdqa
                                                                         #45.16
90
                       %xmm1, %xmm0, %xmm4
                                                                         #45.16
91
            vmovsd
            vbroadcastsd (%rsp), %ymm0
                                                                         #42.8[spill]
92
            vmovdqa
                       %ymm2, %ymm1
                                                                         #45.16
93
                       %xmm4, %xmm4
94
            vmovaps
                                                                         #45.16
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
95
                                       → vmm1 vmm2 vmm3 vmm4
    ..B19.14:
                                      # Preds ..B19.14 ..B19.13
96
                                      # Execution count [2.50e+00]
            vfmadd231pd (%rdx,%rdi,8), %ymm0, %ymm4
98
                                                                         #49.9
            vfmadd231pd 32(%rdx,%rdi,8), %ymm0, %ymm3
                                                                         #49.9
99
            vfmadd231pd 64(%rdx,%rdi,8), %ymm0, %ymm2
                                                                         #49.9
100
            vfmadd231pd 96(%rdx,%rdi,8), %ymm0, %ymm1
                                                                         #49.9
101
            addq
                       $16, %rdi
                                                                         #47.5
102
                       %rcx, %rdi
                                                                         #47.5
            cmpq
103
                       ..B19.14
                                      # Prob 82%
                                                                         #47.5
            jЬ
104
                                      # LOE rax rdx rcx rbx rbp rdi r13 r14 r15 ymm0
105

→ ymm1 ymm2 ymm3 ymm4

    ..B19.15:
                                      # Preds ..B19.14
106
                                      # Execution count [4.50e-01]
107
            vaddpd
                       %ymm3, %ymm4, %ymm0
                                                                         #45.16
108
            vaddpd
                       %ymm1, %ymm2, %ymm1
                                                                         #45.16
109
                       %ymm1, %ymm0, %ymm2
            vaddpd
                                                                         #45.16
110
            vextractf128 $1, %ymm2, %xmm3
                                                                         #45.16
111
                       %xmm3, %xmm2, %xmm4
            vaddpd
                                                                         #45.16
112
            vunpckhpd %xmm4, %xmm4, %xmm5
                                                                         #45.16
113
            vaddsd
                       %xmm5, %xmm4, %xmm6
                                                                         #45.16
114
            vmovsd
                       %xmm6, 8(%rsp)
                                                                         #45.16[spill]
115
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
116
    ..B19.16:
                                      # Preds ..B19.15 ..B19.26
117
                                      # Execution count [5.00e-01]
118
            lea
                       1(%rcx), %rsi
                                                                         #47.5
119
                       %rax, %rsi
                                                                         #47.5
            cmpq
120
                       ..B19.25
                                      # Prob 50%
            ja
121
                                      # LOE rax rdx rcx rbx rbp r13 r14 r15
122
                                      # Preds ..B19.16
123 ...B19.17:
```

```
# Execution count [4.50e-01]
124
             suba
                        %rcx, %rax
                                                                           #47.5
125
                        $4, %rax
                                                                           #47.5
126
             cmpq
             jb
                        ..B19.28
                                       # Prob 10%
                                                                           #47.5
127
                                       # LOE rax rdx rcx rbx rbp r13 r14 r15
128
    ..B19.18:
                                       # Preds ..B19.17
129
                                       # Execution count [4.50e-01]
130
             vmovsd
                        8(%rsp), %xmm1
                                                                           #45.16[spill]
131
                        %rax, %rsi
                                                                           #47.5
            movq
132
                        %xmm0, %xmm0, %xmm0
                                                                           #45.16
             vxorpd
133
                        (%rdx,%rcx,8), %rdi
                                                                           #49.26
             lea
             vmovsd
                        %xmm1, %xmm0, %xmm1
                                                                           #45.16
135
             anda
                        $-4, %rsi
                                                                           #47.5
136
             vbroadcastsd (%rsp), %ymm0
                                                                           #42.8[spill]
137
             xorl
                        %r8d, %r8d
                                                                           #47.5
138
                        %xmm1, %xmm1
                                                                           #45.16
             vmovaps
139
                                       # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
140

→ r15 ymm0 ymm1

    ..B19.19:
                                       # Preds ..B19.19 ..B19.18
141
                                       # Execution count [2.50e+00]
142
             vmulpd
                        (%rdi,%r8,8), %ymm0, %ymm2
                                                                           #49.9
143
                        $4, %r8
                                                                           #47.5
144
             addq
             vaddpd
                        %ymm1, %ymm2, %ymm1
                                                                           #49.9
145
                        %rsi, %r8
                                                                           #47.5
             cmpq
146
                        ..B19.19
                                       # Prob 82%
                                                                           #47.5
             jb
147
                                       # LOE rax rdx rcx rbx rbp rsi rdi r8 r13 r14
148

→ r15 ymm0 ymm1

    ..B19.20:
                                       # Preds ..B19.19
149
                                       # Execution count [4.50e-01]
150
             vextractf128 $1, %ymm1, %xmm0
                                                                           #45.16
151
             vaddpd
                        %xmm0, %xmm1, %xmm2
                                                                           #45.16
152
             vunpckhpd %xmm2, %xmm2, %xmm3
                                                                           #45.16
153
             vaddsd
                        %xmm3, %xmm2, %xmm4
                                                                           #45.16
154
                        %xmm4, 8(%rsp)
                                                                           #45.16[spill]
             vmovsd
155
                                       # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
156
                                       # Preds ..B19.20 ..B19.28
    ..B19.21:
157
                                       # Execution count [4.50e-01]
158
             cmpq
                        %rax, %rsi
                                                                           #47.5
159
                        ..B19.25
                                       # Prob 9%
                                                                           #47.5
             jae
160
                                       # LOE rax rdx rcx rbx rbp rsi r13 r14 r15
161
    ..B19.22:
                                       # Preds ..B19.21
162
                                       # Execution count [4.50e-01]
163
             vmovsd
                        8(%rsp), %xmm1
                                                                           #49.26[spill]
164
                                                                           #49.26
             lea
                        (%rdx,%rcx,8), %rdx
165
                        (%rsp), %xmm2
                                                                           #49.26[spill]
             vmovsd
166
                                       # LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
167
    ..B19.23:
                                       # Preds ..B19.23 ..B19.22
168
                                       # Execution count [2.50e+00]
169
                        (%rdx,%rsi,8), %xmm2, %xmm0
             vmulsd
                                                                           #49.9
170
                                                                           #47.5
             incq
                        %rsi
171
             vaddsd
                        %xmm1, %xmm0, %xmm1
                                                                           #49.9
172
             \operatorname{cmpq}
                        %rax, %rsi
                                                                           #47.5
173
                        ..B19.23
                                       # Prob 82%
                                                                           #47.5
174
             jЬ
```

```
# LOE rax rdx rbx rbp rsi r13 r14 r15 xmm1 xmm2
175
    ..B19.24:
                                      # Preds ..B19.23
176
                                      # Execution count [4.50e-01]
177
            vmovsd
                       %xmm1, 8(%rsp)
                                                                        #[spill]
178
                                      # LOE rbx rbp r13 r14 r15
179
    ..B19.25:
                                      # Preds ..B19.24 ..B19.21 ..B19.16 ..B19.2
180
    → ..B19.33
181
                                      # Execution count [1.00e+00]
182
    # Begin ASM
183
   # # integrate_omp_simd exit
    # End ASM
185
```

Параллельное интегрирование

```
# Begin ASM
   # # integrate omp parallel enter
   # End ASM
                                     # L0E
   ..B18.35:
                                     # Preds ..B18.36
6
                                     # Execution count [1.00e+00]
           movl
                      $.2.486_2_kmpc_loc_struct_pack.18, %edi
                                                                      #33.5
7
                      $0, 72(%rsp)
                                                                      #32.16
           movq
           call
                      kmpc global thread num
                                                                      #33.5
10
                                     # LOE eax
   ..B18.34:
                                     # Preds ..B18.35
11
                                     # Execution count [1.00e+00]
12
           movl
                      %eax, 80(%rsp)
                                                                      #33.5
13
           movl
                      $.2.486 2 kmpc loc struct pack.63, %edi
                                                                      #33.5
14
           xorl
                      %eax, %eax
                                                                      #33.5
15
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.385:
                        kmpc ok to fork
                                                                      #33.5
17
        _tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.386:
18
                                     # LOE eax
19
   ..B18.2:
                                     # Preds ..B18.34
21
                                     # Execution count [1.00e+00]
                      %eax, %eax
                                                                      #33.5
           testl
22
                      ..B18.4
                                     # Prob 50%
                                                                      #33.5
           jе
23
                                     # L0E
24
   ..B18.3:
                                     # Preds ..B18.2
25
                                     # Execution count [0.00e+00]
26
           addq
                      $-16, %rsp
                                                                      #33.5
27
           .cfi def cfa offset 128
28
           movl
29
               $L__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_33__par_region0_2.40,
            30
           movl
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
                                                                      #33.5
           lea
                      16(%rsp), %rax
                                                                      #33.5
31
           movl
                      $4, %esi
                                                                      #33.5
32
                      72(%rax), %rcx
           lea
                                                                      #33.5
33
           lea
                      56(%rax), %r8
                                                                      #33.5
34
                      %rax, (%rsp)
                                                                      #33.5
35
           movq
```

```
lea
                      64(%rax), %r9
                                                                       #33.5
36
           xorl
                      %eax, %eax
                                                                       #33.5
37
        tag value Z22integrate omp parallelRKSt6vectorIdSaIdEEd.388:
           call
                        kmpc fork call
39
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.389:
40
                                     # L0E
41
   ..B18.37:
                                     # Preds ..B18.3
42
                                     # Execution count [0.00e+00]
43
                                                                       #33.5
           addq
                      $16, %rsp
44
           .cfi_def_cfa_offset 112
45
                                                                       #33.5
                      ..B18.7
                                     # Prob 100%
           jmp
                                     # L0E
47
   ..B18.4:
                                     # Preds ..B18.2
48
                                     # Execution count [0.00e+00]
49
                      $.2.486 2 kmpc loc struct pack.63, %edi
           movl
                                                                       #33.5
50
                      %eax, %eax
                                                                       #33.5
           xorl
51
           movl
                      80(%rsp), %esi
                                                                       #33.5
52
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.391:
53
54
                        kmpc_serialized_parallel
                                                                       #33.5
        tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.392:
55
                                     # L0E
56
   ..B18.5:
                                     # Preds ..B18.4
57
                                     # Execution count [0.00e+00]
58
           movl
59
                    kmpv_zero_Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_0, %esi
                #33.5
                                                                       #33.5
                      80(%rsp), %rdi
           lea
60
           lea
                      -8(%rdi), %rdx
                                                                       #33.5
61
                      -16(%rdx), %rcx
                                                                       #33.5
           1 ea
62
           lea
                      -8(%rdx), %r8
                                                                       #33.5
                      (%rsp), %r9
                                                                       #33.5
64
        tag value Z22integrate omp parallelRKSt6vectorIdSaIdEEd.393:
65
           call.
66

→ L__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd_33__par_region0_2.40

            tag_value__Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.394:
67
                                     # L0E
68
   ..B18.6:
                                     # Preds ..B18.5
                                     # Execution count [0.00e+00]
70
           movl
                      $.2.486_2_kmpc_loc_struct_pack.63, %edi
                                                                       #33.5
71
           xorl
                      %eax, %eax
                                                                       #33.5
72
           movl
                      80(%rsp), %esi
                                                                       #33.5
73
                    Z22integrate_omp_parallelRKSt6vectorIdSaIdEEd.395:
        tag value
74
                        kmpc_end_serialized_parallel
           call
                                                                       #33.5
75
        tag value Z22integrate omp parallelRKSt6vectorIdSaIdEEd.396:
76
                                     # L0E
77
   ..B18.7:
                                     # Preds ..B18.37 ..B18.6
78
                                     # Execution count [1.00e+00]
79
   # Begin ASM
   # # integrate omp parallel exit
81
   # End ASM
```