ГУАП

КАФЕДРА № 44

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | К. А. Белов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| ПОЛУЧЕНИЕ ИСПОЛНЯЕМОГО ФАЙЛА ДЛЯ ПЛАТФОРМЫ «ЭЛЬБРУС» |
| по курсу: ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ ЭЛЬБРУС |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |
| --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4941 |  | Н. С. Горбунов |
|  |  | подпись, дата | инициалы, фамилия |

Санкт-Петербург

2023

**Цель работы** – ознакомление с компилятором LCC. Получение исполняемого файла для платформы «Эльбрус». Ознакомление со структурой широких команд.

**Вариант задания 4:**

Умножение матрицы А размером И х Ф на матрицу В размером Ф х О;начальные значения вещественны и задаются по формуле:

# Процесс выполнения работы

В процессе выполнения лабораторной работы, мной, в соответствии с

вариантом задания, была разработана программа, выполняющая умножение двух матриц, и Makefile, создающий исполняемый файл, приведённые в пункте 4. Мной были проведены сравнительные замеры времени, требуемого для сборки программы, и времени исполнения. Сборка проводилась с различными уровнями оптимизации и параметрами. Для сравнения использовались ВК на базе x86-64 и e2k с характеристиками, указанными в Таблице 1. Результаты замеров – в Таблице 2.

Таблица 1 – Характеристики используемых ВК

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Платформа | Тип ЦП | Тех. процесс | Ядра/потоки | Частота ЦП |
| x86-64 | Intel Core i3 7350К | 14 нм | 2(VM) | 4.2 GHz |
| e2k | Эльбрус 4С | 64 нм | 4 | 800 МГц |

Таблица 2 – Временные показатели сборки с вещественными значениями

|  |  |  |
| --- | --- | --- |
| Параметры сборки | Эльбрус – 4С  lcc 1.25 | Intel Core i3 9300H  g++ |
| -O0 | 2,49 сек | 0,60 сек |
| -О1 | 1,63 сек | 0,38 сек. |
| -О2 | 1,64 сек | 0,37 сек |

Из результатов следует, что оптимальным вариантом сборки для платформы E2K являются параметры O2.

В качестве демонстрации ассемблерного кода был выбран участок программы, представленный в таблице 3.

Таблица 3 – Ассемблерный листинг

|  |  |
| --- | --- |
| Ассемблер на e2k | Ассемблер на x86 |
| .file "Lab3.cpp"  .ignoreld\_st\_style  .ignorestrict\_delay  .text  .global main  .type main, #function  .align 8  main:  .cfi\_startproc  {  setwd wsz = 0x8,  nfx = 0x1, dbl = 0x0  setbnrsz = 0x3, rbs =  0x4, rcur = 0x0  getsp,0 \_f16s,\_lts1hi  0xfff0, %r2  }  {  adds,0,sm 0x5, 0x0,  %r3  adds,1,sm 0x6, 0x0,  %r4  adds,2,sm 0x9, 0x0,  %r5  addd,3,sm 0x0,  [ \_f64,\_lts0 \_ZSt4cout ], %b[0]  addd,4,sm 0x0,  \_f64,\_lts2 0x3ff0000000000000,  %b[1]  }  .LCS.1:  {  nop 4  disp %ctpr1, \_ZNSolsEd  }  {  call %ctpr1, wbs = 0x4  }  .LCS.2:  {  nop 5  addd,0,sm 0x0, 0x0,  %r0  return %ctpr3; ipd 2  }  {  ct %ctpr3  }  .cfi\_endproc  .size main, .- main  .local  \_\_sti\_\_\_15\_Lab3\_cpp\_main  .type  \_\_sti\_\_\_15\_Lab3\_cpp\_main,  #function  .align 8  \_\_sti\_\_\_15\_Lab3\_cpp\_main:  .cfi\_startproc  {  setwd wsz = 0x8,  nfx = 0x1, dbl = 0x0  setbnrsz = 0x3, rbs =  0x4, rcur = 0x0  getsp,0 \_f16s,\_lts1hi  0xffe0, %r1  }  {  addd,0,sm 0x0,  [ \_f64,\_lts0  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE ], %b[0]  }  .LCS.3:  {  nop 4  disp %ctpr1,  \_ZNSt8ios\_base4InitC1Ev  }  {  call %ctpr1, wbs = 0x4  }  .LCS.4:  {  addd,0,sm 0x0,  [ \_f64,\_lts0  \_ZNSt8ios\_base4InitD1Ev ], %b[0]  addd,1,sm 0x0,  [ \_f64,\_lts2  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE ], %b[1]  }  {  addd,0,sm 0x0,  [ \_f64,\_lts0 \_\_dso\_handle ], %b[2]  }  .LCS.5:  {  nop 4  disp %ctpr1, \_\_cxa\_atexit  }  {  call %ctpr1, wbs = 0x4  }  .LCS.6:  {  nop 5  return %ctpr3; ipd 2  }  {  ct %ctpr3  }  .cfi\_endproc  .size  \_\_sti\_\_\_15\_Lab3\_cpp\_main, .-  \_\_sti\_\_\_15\_Lab3\_cpp\_main  .section .bss  .local  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE  .type  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE, #object  .size  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE, 0x1  .align 1  \_ZN33\_INTERNAL\_15\_Lab3\_cpp\_m  ainSt8\_\_ioinitE:  .skip 0x1  .hidden \_\_dso\_handle  .section .ctors, "aw",  @progbits  .align 8  .dword  \_\_sti\_\_\_15\_Lab3\_cpp\_main  .weak  elbrus\_compiler\_v1.25.19\_Aug\_25  \_2021  elbrus\_compiler\_v1.25.19\_Aug\_25  \_2021 = 0x0 | .file "lab3.cpp"  .text  .local \_ZStL8\_\_ioinit  .comm \_ZStL8\_\_ioinit,1,1  .section .rodata  .LC0:  .string "Matrix A\n"  .LC1:  .string " "  .LC2:  .string "\n"  .LC3:  .string "Matrix B\n"  .LC5:  .string "Output Matrix:"  .text  .globl main  .type main, @function  main:  .LFB1761:  .cfi\_startproc  pushq %rbp  .cfi\_def\_cfa\_offset 16  .cfi\_offset 6, -16  movq %rsp, %rbp  .cfi\_def\_cfa\_register 6  subq $704, %rsp  movl $8, -16(%rbp)  movl $6, -20(%rbp)  movl $9, -24(%rbp)  movl $0, -4(%rbp)  jmp .L2  .L5:  movl $0, -8(%rbp)  jmp .L3  .L4:  movl -4(%rbp), %edx  movl %edx, %eax  sarl $31, %eax  shrl $31, %eax  addl %eax, %edx  andl $1, %edx  subl %eax, %edx  movl %edx, %eax  addl $1, %eax  imull -16(%rbp), %eax  movl %eax, %ecx  movl -8(%rbp), %edx  movl %edx, %eax  sarl $31, %eax  shrl $31, %eax  addl %eax, %edx  andl $1, %edx  subl %eax, %edx  movl %edx, %eax  subl $1, %eax  imull -24(%rbp), %eax  addl %ecx, %eax  pxor %xmm0, %xmm0  cvtsi2ssl %eax, %xmm0  pxor %xmm1, %xmm1  cvtsi2ssl -20(%rbp), %xmm1  divss %xmm1, %xmm0  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movss %xmm0, -224(%rbp,%rax,4)  addl $1, -8(%rbp)  .L3:  movl -8(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L4  addl $1, -4(%rbp)  .L2:  movl -4(%rbp), %eax  cmpl -20(%rbp), %eax  jl .L5  movl $0, -4(%rbp)  jmp .L6  .L9:  movl $0, -8(%rbp)  jmp .L7  .L8:  movl -4(%rbp), %edx  movl %edx, %eax  sarl $31, %eax  shrl $31, %eax  addl %eax, %edx  andl $1, %edx  subl %eax, %edx  movl %edx, %eax  subl $1, %eax  imull -20(%rbp), %eax  movl %eax, %ecx  movl -8(%rbp), %edx  movl %edx, %eax  sarl $31, %eax  shrl $31, %eax  addl %eax, %edx  andl $1, %edx  subl %eax, %edx  movl %edx, %eax  addl $1, %eax  imull -24(%rbp), %eax  subl %eax, %ecx  movl %ecx, %edx  pxor %xmm0, %xmm0  cvtsi2ssl %edx, %xmm0  pxor %xmm1, %xmm1  cvtsi2ssl -16(%rbp), %xmm1  divss %xmm1, %xmm0  movl -8(%rbp), %eax  movslq %eax, %rcx  movl -4(%rbp), %eax  movslq %eax, %rdx  movq %rdx, %rax  salq $3, %rax  addq %rdx, %rax  addq %rcx, %rax  movss %xmm0, -512(%rbp,%rax,4)  addl $1, -8(%rbp)  .L7:  movl -8(%rbp), %eax  cmpl -24(%rbp), %eax  jl .L8  addl $1, -4(%rbp)  .L6:  movl -4(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L9  movl $.LC0, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  movl $0, -4(%rbp)  jmp .L10  .L13:  movl $0, -8(%rbp)  jmp .L11  .L12:  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movl -224(%rbp,%rax,4), %eax  movd %eax, %xmm0  movl $\_ZSt4cout, %edi  call \_ZNSolsEf  movl $.LC1, %esi  movq %rax, %rdi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -8(%rbp)  .L11:  movl -8(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L12  movl $.LC2, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -4(%rbp)  .L10:  movl -4(%rbp), %eax  cmpl -20(%rbp), %eax  jl .L13  movl $.LC3, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  movl $0, -4(%rbp)  jmp .L14  .L17:  movl $0, -8(%rbp)  jmp .L15  .L16:  movl -8(%rbp), %eax  movslq %eax, %rcx  movl -4(%rbp), %eax  movslq %eax, %rdx  movq %rdx, %rax  salq $3, %rax  addq %rdx, %rax  addq %rcx, %rax  movl -512(%rbp,%rax,4), %eax  movd %eax, %xmm0  movl $\_ZSt4cout, %edi  call \_ZNSolsEf  movl $.LC1, %esi  movq %rax, %rdi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -8(%rbp)  .L15:  movl -8(%rbp), %eax  cmpl -24(%rbp), %eax  jl .L16  movl $.LC2, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -4(%rbp)  .L14:  movl -4(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L17  movl $0, -4(%rbp)  jmp .L18  .L23:  movl $0, -8(%rbp)  jmp .L19  .L22:  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  pxor %xmm0, %xmm0  movss %xmm0, -704(%rbp,%rax,4)  movl $0, -12(%rbp)  jmp .L20  .L21:  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movss -704(%rbp,%rax,4), %xmm1  movl -12(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movss -224(%rbp,%rax,4), %xmm2  movl -8(%rbp), %eax  movslq %eax, %rcx  movl -12(%rbp), %eax  movslq %eax, %rdx  movq %rdx, %rax  salq $3, %rax  addq %rdx, %rax  addq %rcx, %rax  movss -512(%rbp,%rax,4), %xmm0  mulss %xmm2, %xmm0  addss %xmm1, %xmm0  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movss %xmm0, -704(%rbp,%rax,4)  addl $1, -12(%rbp)  .L20:  movl -12(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L21  addl $1, -8(%rbp)  .L19:  movl -8(%rbp), %eax  cmpl -24(%rbp), %eax  jl .L22  addl $1, -4(%rbp)  .L18:  movl -4(%rbp), %eax  cmpl -20(%rbp), %eax  jl .L23  movl $.LC2, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  movl $.LC5, %esi  movq %rax, %rdi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  movl $.LC2, %esi  movq %rax, %rdi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  movl $0, -4(%rbp)  jmp .L24  .L27:  movl $0, -8(%rbp)  jmp .L25  .L26:  movl -8(%rbp), %eax  cltq  movl -4(%rbp), %edx  movslq %edx, %rdx  salq $3, %rdx  addq %rdx, %rax  movl -704(%rbp,%rax,4), %eax  movd %eax, %xmm0  movl $\_ZSt4cout, %edi  call \_ZNSolsEf  movl $.LC1, %esi  movq %rax, %rdi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -8(%rbp)  .L25:  movl -8(%rbp), %eax  cmpl -16(%rbp), %eax  jl .L26  movl $.LC2, %esi  movl $\_ZSt4cout, %edi  call \_ZStlsISt11char\_traitsIcEERSt13basic\_ostreamIcT\_ES5\_PKc  addl $1, -4(%rbp)  .L24:  movl -4(%rbp), %eax  cmpl -20(%rbp), %eax  jl .L27  movl $0, %eax  leave  .cfi\_def\_cfa 7, 8  ret  .cfi\_endproc  .LFE1761:  .size main, .-main  .type \_Z41\_\_static\_initialization\_and\_destruction\_0ii, @function  \_Z41\_\_static\_initialization\_and\_destruction\_0ii:  .LFB2289:  .cfi\_startproc  pushq %rbp  .cfi\_def\_cfa\_offset 16  .cfi\_offset 6, -16  movq %rsp, %rbp  .cfi\_def\_cfa\_register 6  subq $16, %rsp  movl %edi, -4(%rbp)  movl %esi, -8(%rbp)  cmpl $1, -4(%rbp)  jne .L31  cmpl $65535, -8(%rbp)  jne .L31  movl $\_ZStL8\_\_ioinit, %edi  call \_ZNSt8ios\_base4InitC1Ev  movl $\_\_dso\_handle, %edx  movl $\_ZStL8\_\_ioinit, %esi  movl $\_ZNSt8ios\_base4InitD1Ev, %edi  call \_\_cxa\_atexit  .L31:  nop  leave  .cfi\_def\_cfa 7, 8  ret  .cfi\_endproc  .LFE2289:  .size \_Z41\_\_static\_initialization\_and\_destruction\_0ii, .-\_Z41\_\_static\_initialization\_and\_destruction\_0ii  .type \_GLOBAL\_\_sub\_I\_main, @function  \_GLOBAL\_\_sub\_I\_main:  .LFB2290:  .cfi\_startproc  pushq %rbp  .cfi\_def\_cfa\_offset 16  .cfi\_offset 6, -16  movq %rsp, %rbp  .cfi\_def\_cfa\_register 6  movl $65535, %esi  movl $1, %edi  call \_Z41\_\_static\_initialization\_and\_destruction\_0ii  popq %rbp  .cfi\_def\_cfa 7, 8  ret  .cfi\_endproc  .LFE2290:  .size \_GLOBAL\_\_sub\_I\_main, .-\_GLOBAL\_\_sub\_I\_main  .section .init\_array,"aw"  .align 8  .quad \_GLOBAL\_\_sub\_I\_main  .hidden \_\_dso\_handle  .ident "GCC: (GNU) 12.2.1 20221121 (Red Hat 12.2.1-4)"  .section .note.GNU-stack,"",@progbits |

# Исходный код

**Файл Makefile:**

all:

time g++ lab3.cpp -o lab3

time g++ lab3.cpp -S lab3

time g++ lab3.cpp -c lab3

o1:

time g++ -O1 lab3.cpp -o lab3.o1

o2:

time g++ -O2 lab3.cpp -o lab3.o2

o3:

time g++ -O3 lab3.cpp -o lab3.o3

clean:

rm -rf \*.o lab3

rm -rf \*.s lab3

**Файл lab3.cpp**

#include <iostream>

using namespace std;

int main()

{

// Ф-8, И-6, О-9

int f = 8, n = 6, o = 9, i, j, k;

float a[6][8], b[8][9], mult[6][8];

for (i = 0; i < n; i++)

{

for (j = 0; j < f; j++)

{

a[i][j] = ((float)(f \* ((i % 2) + 1) + o \* ((j % 2) - 1))) / (float)n;

}

}

for (i = 0; i < f; i++)

{

for (j = 0; j < o; j++)

{

b[i][j] = ((float)(n \* ((i % 2) - 1) - o \* ((j % 2) + 1))) / (float)f;

}

}

cout << "Matrix A\n";

for (i = 0; i < n; i++)

{

for (j = 0; j < f; j++)

{

cout << a[i][j] << " ";

}

cout << "\n";

}

cout << "Matrix B\n";

for (i = 0; i < f; i++)

{

for (j = 0; j < o; j++)

{

cout << b[i][j] << " ";

}

cout << "\n";

}

// Multiplying matrix a and b and storing in array mult.

for (i = 0; i < n; i++)

{

for (j = 0; j < o; j++)

{

mult[i][j] = 0;

for (k = 0; k < f; k++)

{

mult[i][j] += a[i][k] \* b[k][j];

}

}

} // Displaying the multiplication of two matrix.

cout << "\n" << "Output Matrix:" << "\n";

for (i = 0; i < n; ++i)

{

for (j = 0; j < f; ++j)

{

cout << mult[i][j] << " ";

}

cout << "\n";

}

return 0;

}

# Вывод

В ходе данной лабораторной работы мной была разработана программа, производящая сложение и вычитание двух матриц. Затем, мной было проведено тестирование и отладка данной программы, произведены замеры времени компиляции и выполнения.

Следующим этапом являлась модификация программы таким образом, чтобы производить вычисления на матрицах с размерностью в 10 и 100- кратной исходной. Выполнение модифицированной программы так же было протестировано и произведены аналогичные замеры.

Таким образом можно заключить, что в результате выполнения данной лабораторной работы мной были получены навыки работы с утилитой make, получены навыки написания Makefile, а также отточены навыки программирования на языке высокого уровня C++. В дополнение к этому, мной были приобретены навыки переноса программы на архитектуру e2k