Лабораторная работа 0

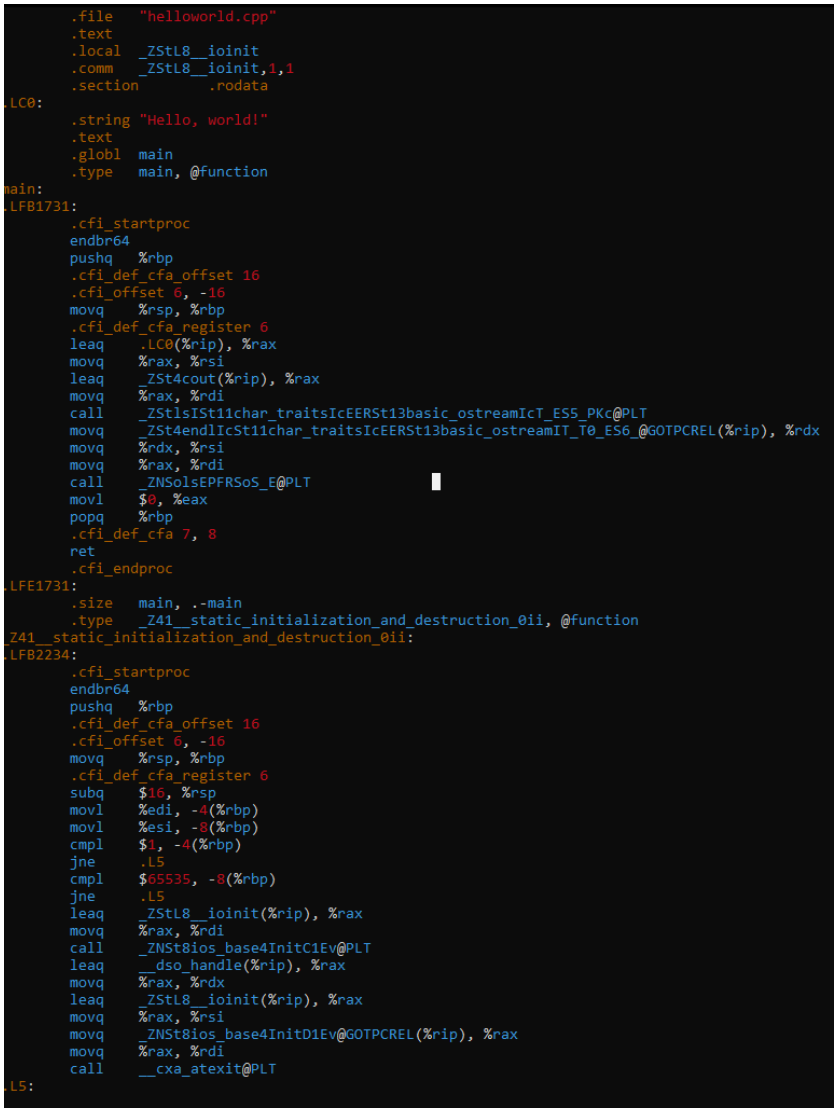
Калашников Михаил, Б03-205

Арифметика и основы работы с листингами

1. А я поставил WSL.
2. И установил g++.
3. Написал helloworld и сгенерировал листинг с помощью флага -S (рис. 1).
4. Сгенерируем листинг для программы, складывающей два числа. В листинге изменим значение одной из переменных. Скомпилируем результат и получим другой вывод (рис. 2).
5. Напишем helloworld на C и сравним с листингом из пункта 3. Листинг выглядит гораздо проще и компактнее (рис. 3). Для пункта 4 получает аналогичный результат.
6. Добавим в код операцию вычитания и определим, что за присвоение отвечает команда mov, за сложение – add, а за вычитание – sub (рис. 4).
7. Объявим три глобальные переменные типов int (a = 1), float (b = 2.5) и char (c = '0'). Найдем их объявление в листинге (рис. 5). В int записывается ее же численное значение, в char индекс символа в ASCII, а вот с float не очень понятно. Похоже, что он переводит число с плавающей точкой в целое число и хранит уже целочисленное значение.
8. Регистрами общего назначения являются регистры eax, edx, ecx и ebx. Их размер составляет 4 байта. Так же существуют расширенные регистры rax, rdx, rcx и rbx, имеющие размер 8 байт. Вызываются они с помощью %(название регистра). Например, команда "movl %eax, %edx"переложит содержимое регистра eax в регистр edx.

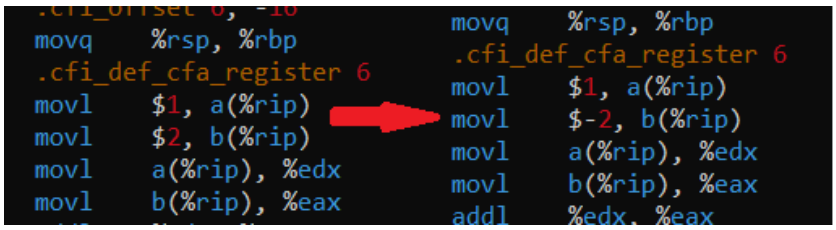
9. Команда mul отвечает за умножение беззнаковых переменных, а imul за умножение знаковых. Команды div/idiv работают аналогично, только отвечают уже за деление. Это написано в интернете. На деле же отличие заключается в том, что команда mul принимает только один аргумент, а второй множитель берет из регистра eax (рис. 6). Знак переменных на работу функции никак не влияет. idiv и div принимают только один аргумент – делитель, делимое они берут из регистра eax. idiv работает с целочисленными значениями любого знака, а div – только с положительными.

Приложения



```
        .file   "helloworld.cpp"
        .text
        .local  _ZStL8__ioinit
        .comm   _ZStL8__ioinit,1,1
        .section        .rodata
.LC0:
        .string "Hello, world!"
        .text
        .globl  main
        .type   main, @function
main:
.LFB1731:
        .cfi_startproc
        endbr64
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register 6
        leaq    .LC0(%rip), %rax
        movq    %rax, %rsi
        leaq    _ZSt4cout(%rip), %rax
        movq    %rax, %rdi
        call    _ZStlsISt11char_traitsIcEERSt13basic_ostreamIcT_ES5_PKc@PLT
        movq    _ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_@GOTPCREL(%rip), %rdx
        movq    %rdx, %rsi
        movq    %rax, %rdi
        call    _ZNSolsEPFRSoS_E@PLT
        movl    $0, %eax
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE1731:
        .size   main, .-main
        .type   _Z41__static_initialization_and_destruction_0ii, @function
_Z41__static_initialization_and_destruction_0ii:
.LFB2234:
        .cfi_startproc
        endbr64
        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     .L5
        cmpl    $65535, -8(%rbp)
        jne     .L5
        leaq    _ZStL8__ioinit(%rip), %rax
        movq    %rax, %rdi
        call    _ZNSt8ios_base4InitC1Ev@PLT
        leaq    __dso_handle(%rip), %rax
        movq    %rax, %rdx
        leaq    _ZStL8__ioinit(%rip), %rax
        movq    %rax, %rsi
        movq    _ZNSt8ios_base4InitD1Ev@GOTPCREL(%rip), %rax
        movq    %rax, %rdi
        call    __cxa_atexit@PLT
.L5:
```

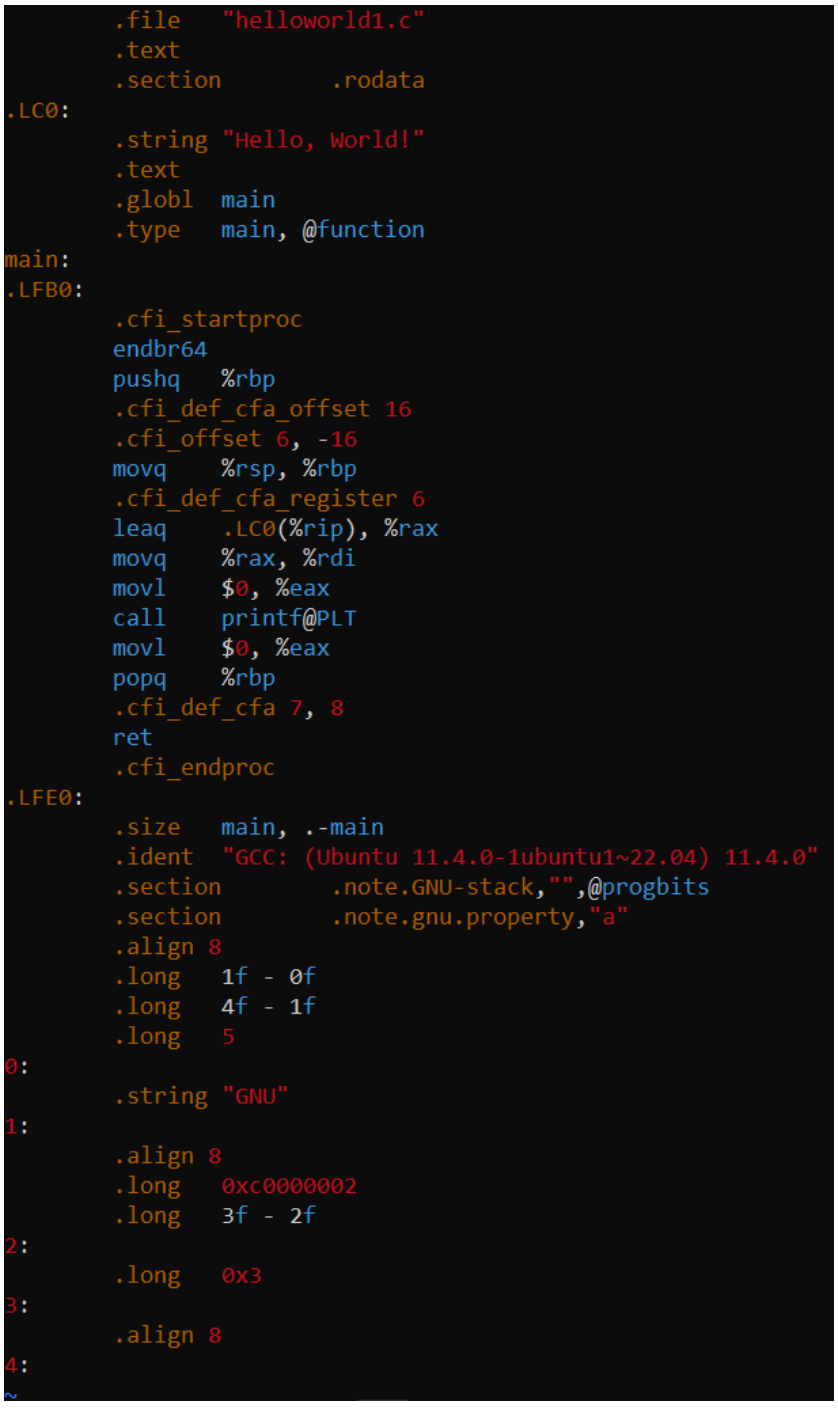
Рис. 1: Ассемблерный листинг helloworld.cpp



```
movq    %rsp, %rbp
.cfi_def_cfa_register 6
movl    $1, a(%rip)
movl    $2, b(%rip)
movl    a(%rip), %edx
movl    b(%rip), %eax
```

```
movq    %rsp, %rbp
.cfi_def_cfa_register 6
movl    $1, a(%rip)
movl    $-2, b(%rip)
movl    a(%rip), %edx
movl    b(%rip), %eax
addl    %edx, %eax
```

Рис. 2: Изменение листинга



```
        .file   "helloworld1.c"
        .text
        .section        .rodata
.LC0:
        .string "Hello, World!"
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        endbr64
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register 6
        leaq    .LC0(%rip), %rax
        movq    %rax, %rdi
        movl    $0, %eax
        call    printf@PLT
        movl    $0, %eax
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0"
        .section        .note.GNU-stack,"",@progbits
        .section        .note.gnu.property,"a"
        .align 8
        .long   1f - 0f
        .long   4f - 1f
        .long   5
0:
        .string "GNU"
1:
        .align 8
        .long   0xc0000002
        .long   3f - 2f
2:
        .long   0x3
3:
        .align 8
4:
~
```

Рис. 3: Листинг helloworld на C

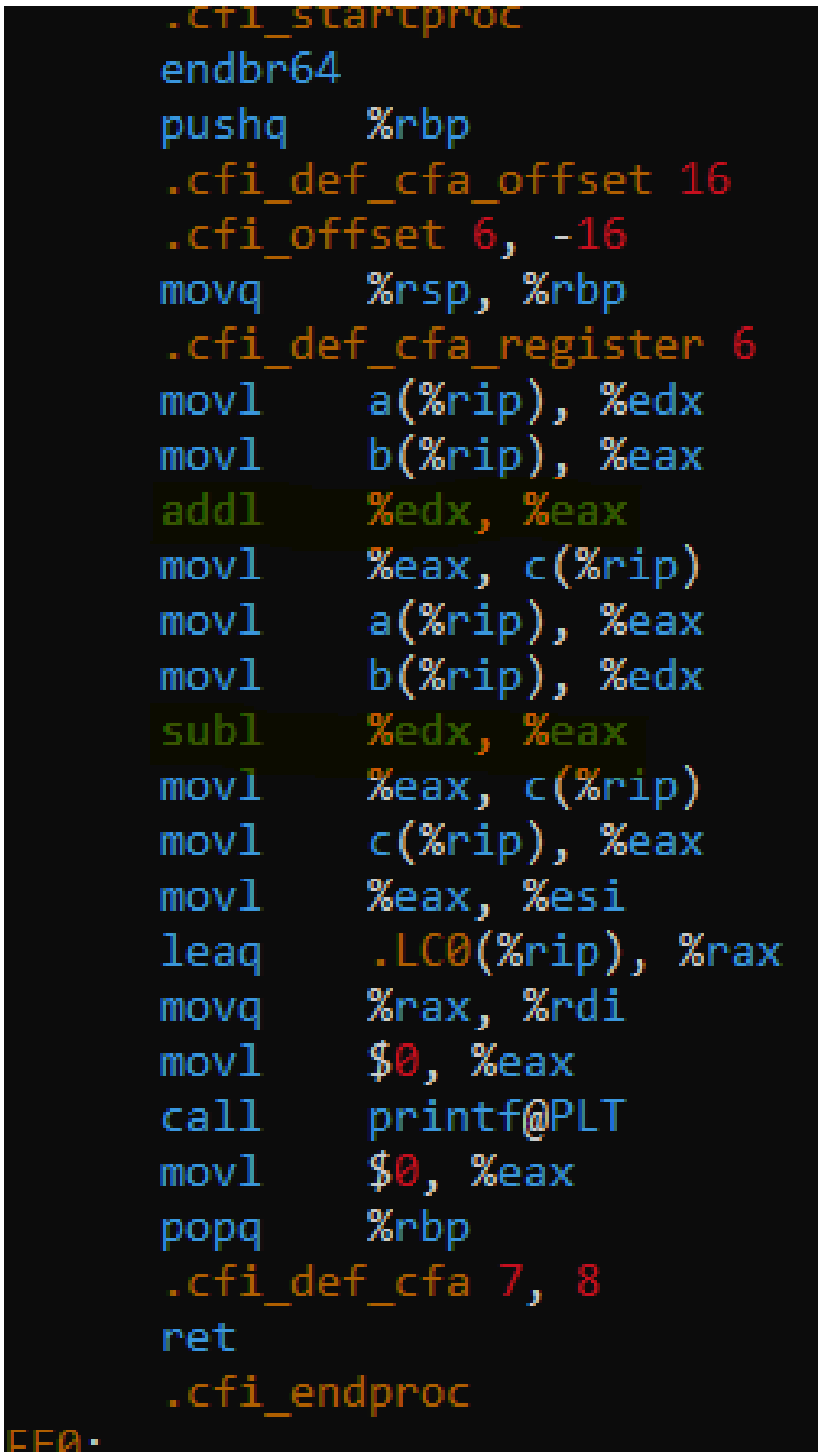


Рис. 4: Сложение и вычитание

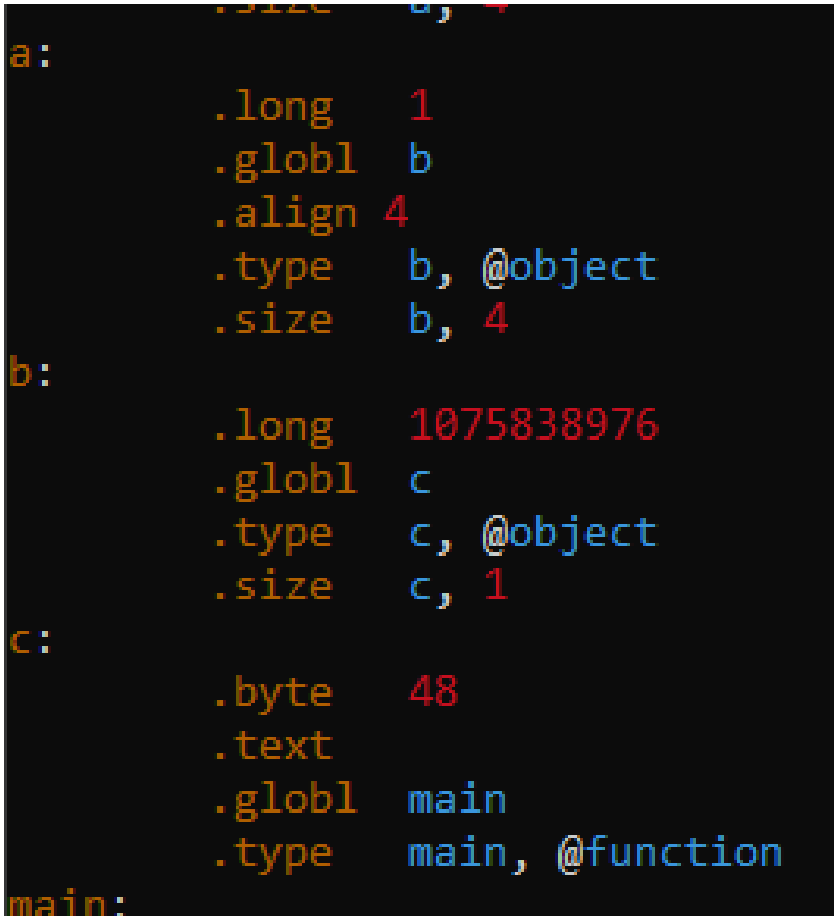


Рис. 5: Глобальные переменные различных типов

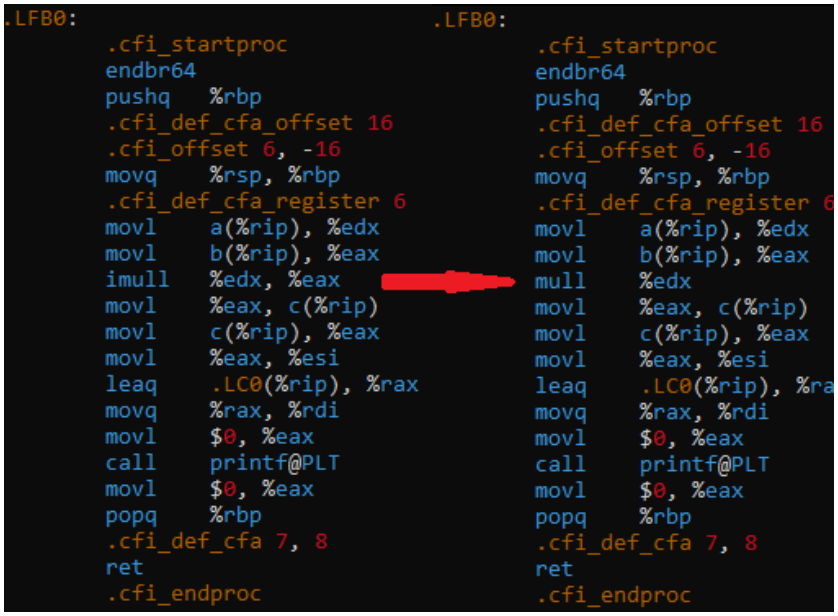


Рис. 6: Замена imul на mul