Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа информатики и вычислительной техники

**Лабораторная работа № 4**

**Дисциплина**: Низкоуровневое программирование

**Тема:** Раздельная компиляция

Выполнил студент гр. 3530901/10005 Тучков Д.А.

Преподаватель КореневД.А.

«30»ноября 2022 г.

Санкт-Петербург

2022

Оглавление

[**Техническое задание** 3](#_Toc120300519)

[**1. Программа на языке C** 3](#_Toc120300520)

[**2. Сборка программы «по шагам»** 5](#_Toc120300521)

[Препроцессирование 6](#_Toc120300522)

[Компиляция 6](#_Toc120300523)

[Ассемблирование 11](#_Toc120300524)

[Компоновка 13](#_Toc120300525)

[**3. Создание статической библиотеки и make-файлов** 16](#_Toc120300526)

[**Вывод** 20](#_Toc120300527)

Техническое задание

1. **Формулировка задачи**

1) На языке C разработать функцию, реализующую определенную вариантом задания функциональность. Поместить определение функции в отдельный исходный файл, оформить заголовочный файл. Разработать тестовую программу на языке C.

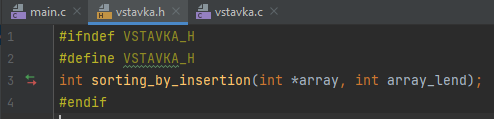
2) Собрать программу «по шагам». Проанализировать выход препроцессора и компилятора. Проанализировать состав и содержимое секций, таблицы символов, таблицы перемещений и отладочную информацию, содержащуюся в объектных файлах иисполняемом файле.

3) Выделить разработанную функцию в статическую библиотеку. Разработать make-файлыд ля сборки библиотеки и использующей ее тестовой программы. Проанализировать ход сборки библиотеки и программы, созданные файлы зависимостей.

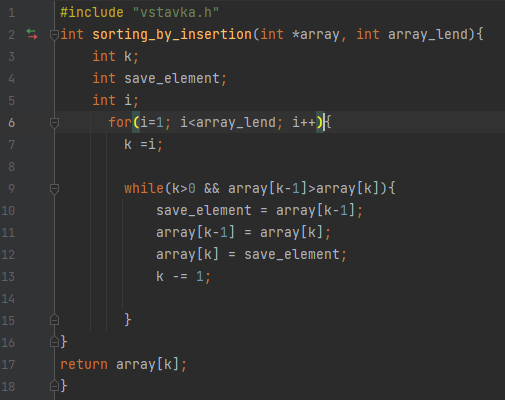
**Вариант задания**

Сортировка массива методом вставки.

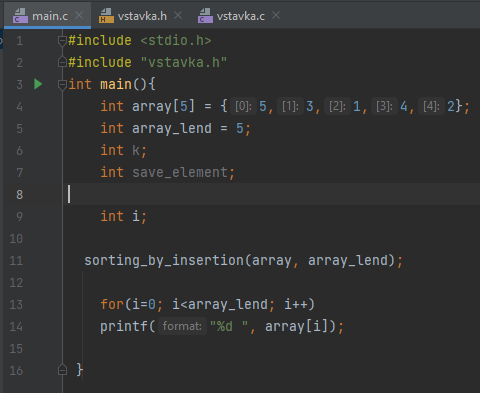
Листинг 1.1. Заголовочный файл vstavka.h



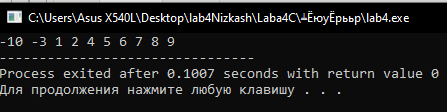
Листинг 1.2. Основной файл vstavka.c



Листинг 1.3. Файл main.c

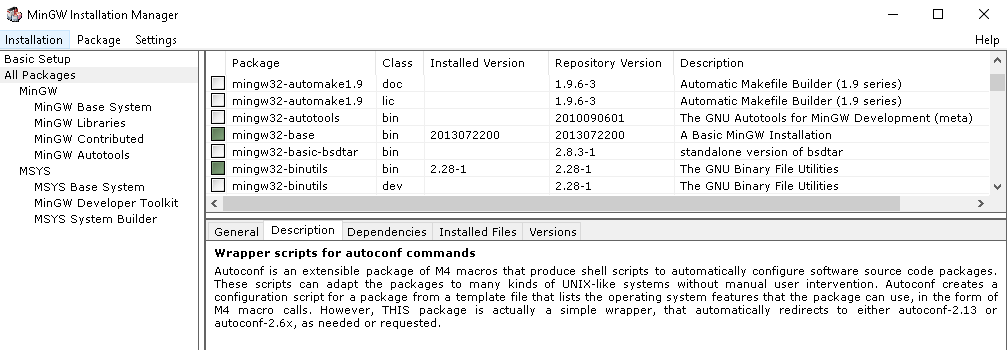


Работа программы

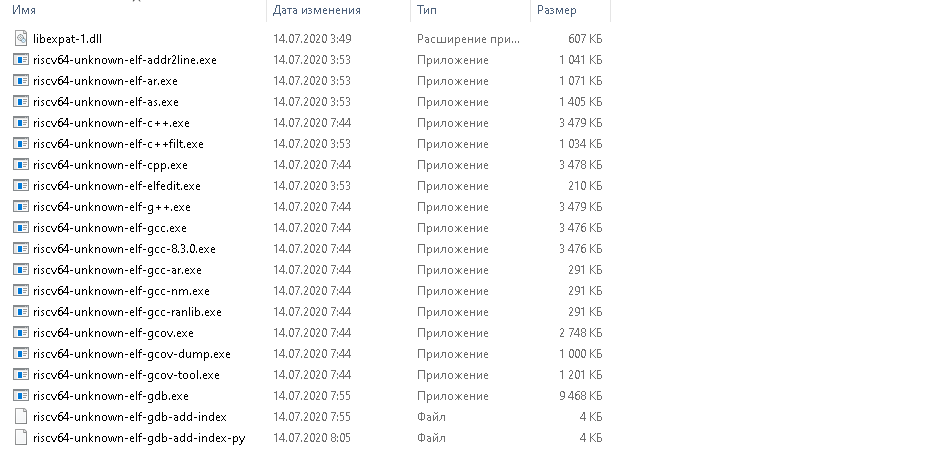


1. **Реализация. Подготовка к проведению работы.**

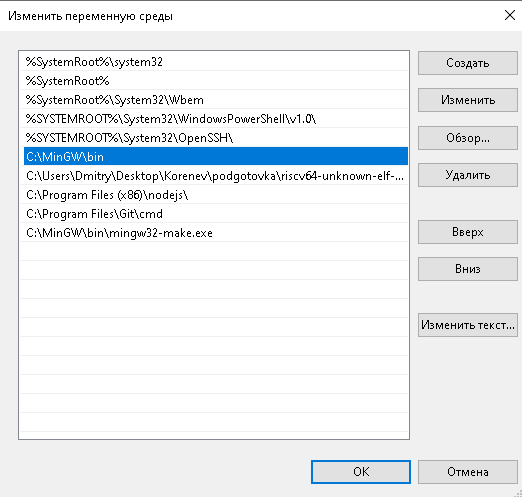
А) Скачиваем программу MinGW и выставляем необходимые настройки.



Б) Скачиваем файл riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32

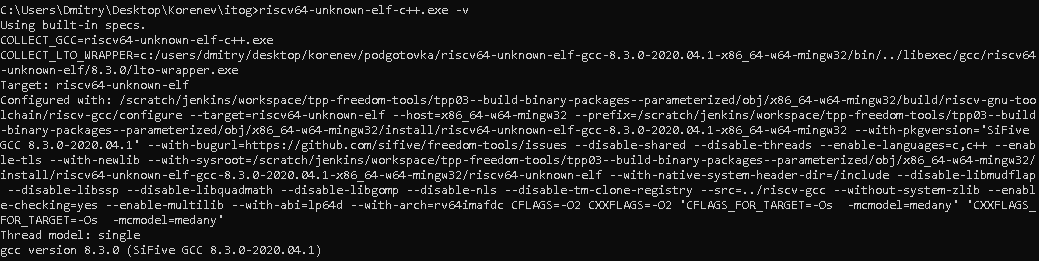


В) Добавить исполняемые файлы в Path.

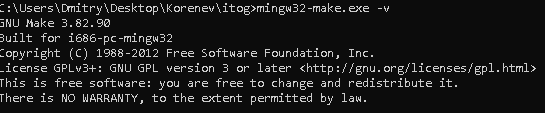


Д) Проверяем работу програм:

Riscv64



Mingw32

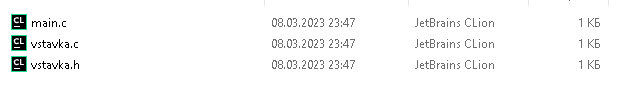


1. **Сборка простейшей программы**

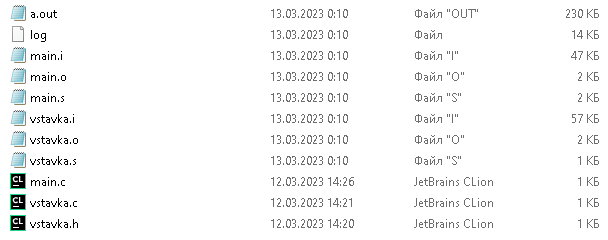
Для сборки программы выполним команду:

riscv64-unknown-elf-gcc-8.3.0.exe .\vstavka.c --save-temps -march=rv32i -mabi=ilp32 -O1 -v >log 2>&1

Было:



Стало:



Программа *riscv64-unknown-elf-gcc* является драйвером компилятора gcc, в данном случае она запускается со следующими параметрами командной строки:

*--save-temps* – сохранять промежуточные файлы, создаваемые в процессе сборки;

*-march=rv32i -mabi=ilp32* – целевым является процессор с базовой архитектурой системы команд RV32I;

*-O1* – выполнять простые оптимизации генерируемого кода (мы используем эту опцию в примерах, потому что обычно генерируемый код получается более простым);

*-v* – печатать (в стандартный поток ошибок) выполняемые драйвером команды, а также дополнительную информацию.

«перенаправление вывода»:

*>log* - вместо печати в консоли (обычно, это означает «на экране») вывод программы направляется в файл с именем “log” (если файл не существует, он создается; если файл существует, его содержимое будет утеряно);

*2>&1* – поток вывода ошибок (2 – стандартный «номер» этого потока) «связывается» с поток вывода («номер» 1), т.е. сообщения об ошибках (и информация, вывод которой вызван использованием флага “-v”, см.выше) также выводятся в файл “log”.

Результат:

*log* – текстовый файл, содержащий сообщения компилятора, ассемблера и компоновщика, а также выполняемые команды и дополнительную информацию;

*Добавились файл*:

*Vstavka*.i – текстовый файл

Vstavka.o – бинарный файл

Vstavka.s – код ассемблера

1. **Процесс сборки, реализуемый драйвером компилятора log (Посмотрим содержимое файла log).**

Using built-in specs.

COLLECT\_GCC=riscv64-unknown-elf-gcc-8.3.0.exe

COLLECT\_LTO\_WRAPPER=c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/lto-wrapper.exe

Target: riscv64-unknown-elf

Configured with: /scratch/jenkins/workspace/tpp-freedom-tools/tpp03--build-binary-packages--parameterized/obj/x86\_64-w64-mingw32/build/riscv-gnu-toolchain/riscv-gcc/configure --target=riscv64-unknown-elf --host=x86\_64-w64-mingw32 --prefix=/scratch/jenkins/workspace/tpp-freedom-tools/tpp03--build-binary-packages--parameterized/obj/x86\_64-w64-mingw32/install/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32 --with-pkgversion='SiFive GCC 8.3.0-2020.04.1' --with-bugurl=https://github.com/sifive/freedom-tools/issues --disable-shared --disable-threads --enable-languages=c,c++ --enable-tls --with-newlib --with-sysroot=/scratch/jenkins/workspace/tpp-freedom-tools/tpp03--build-binary-packages--parameterized/obj/x86\_64-w64-mingw32/install/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/riscv64-unknown-elf --with-native-system-header-dir=/include --disable-libmudflap --disable-libssp --disable-libquadmath --disable-libgomp --disable-nls --disable-tm-clone-registry --src=../riscv-gcc --without-system-zlib --enable-checking=yes --enable-multilib --with-abi=lp64d --with-arch=rv64imafdc CFLAGS=-O2 CXXFLAGS=-O2 'CFLAGS\_FOR\_TARGET=-Os -mcmodel=medany' 'CXXFLAGS\_FOR\_TARGET=-Os -mcmodel=medany'

Thread model: single

gcc version 8.3.0 (SiFive GCC 8.3.0-2020.04.1)

COLLECT\_GCC\_OPTIONS='-save-temps' '-march=rv32i' '-mabi=ilp32' '-O1' '-v'

c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/cc1.exe -E -quiet -v -imultilib rv32i/ilp32 -iprefix c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../lib/gcc/riscv64-unknown-elf/8.3.0/ -isysroot c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../riscv64-unknown-elf .\vstavka.c -march=rv32i -mabi=ilp32 -O1 -fpch-preprocess -o vstavka.i

ignoring duplicate directory "c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/lib/gcc/../../lib/gcc/riscv64-unknown-elf/8.3.0/include"

ignoring nonexistent directory "c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../riscv64-unknown-elf/scratch/jenkins/workspace/tpp-freedom-tools/tpp03--build-binary-packages--parameterized/obj/x86\_64-w64-mingw32/install/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/lib/gcc/riscv64-unknown-elf/8.3.0/../../../../include"

ignoring duplicate directory "c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/lib/gcc/../../lib/gcc/riscv64-unknown-elf/8.3.0/include-fixed"

ignoring duplicate directory "c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/lib/gcc/../../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/include"

ignoring duplicate directory "c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../riscv64-unknown-elf/include"

#include "..." search starts here:

#include <...> search starts here:

c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../lib/gcc/riscv64-unknown-elf/8.3.0/include

c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../lib/gcc/riscv64-unknown-elf/8.3.0/include-fixed

c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/include

End of search list.

COLLECT\_GCC\_OPTIONS='-save-temps' '-march=rv32i' '-mabi=ilp32' '-O1' '-v'

c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/cc1.exe -fpreprocessed vstavka.i -quiet -dumpbase vstavka.c -march=rv32i -mabi=ilp32 -auxbase vstavka -O1 -version -o vstavka.s

GNU C17 (SiFive GCC 8.3.0-2020.04.1) version 8.3.0 (riscv64-unknown-elf)

compiled by GNU C version 5.3.1 20160211, GMP version 6.1.0, MPFR version 3.1.4, MPC version 1.0.3, isl version isl-0.18-GMP

GGC heuristics: --param ggc-min-expand=30 --param ggc-min-heapsize=4096

GNU C17 (SiFive GCC 8.3.0-2020.04.1) version 8.3.0 (riscv64-unknown-elf)

compiled by GNU C version 5.3.1 20160211, GMP version 6.1.0, MPFR version 3.1.4, MPC version 1.0.3, isl version isl-0.18-GMP

GGC heuristics: --param ggc-min-expand=30 --param ggc-min-heapsize=4096

Compiler executable checksum: be92589c6f39f92b6bc439d1cd72ba15

COLLECT\_GCC\_OPTIONS='-save-temps' '-march=rv32i' '-mabi=ilp32' '-O1' '-v'

c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/bin/as.exe -v --traditional-format -march=rv32i -mabi=ilp32 -o vstavka.o vstavka.s

GNU assembler version 2.32 (riscv64-unknown-elf) using BFD version (SiFive Binutils 2.32.0-2020.04.1) 2.32

COMPILER\_PATH=c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/bin/

LIBRARY\_PATH=c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/rv32i/ilp32/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib/rv32i/ilp32/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../riscv64-unknown-elf/lib/rv32i/ilp32/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib/;c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../riscv64-unknown-elf/lib/

COLLECT\_GCC\_OPTIONS='-save-temps' '-march=rv32i' '-mabi=ilp32' '-O1' '-v'

c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/collect2.exe -plugin c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/liblto\_plugin-0.dll -plugin-opt=c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../libexec/gcc/riscv64-unknown-elf/8.3.0/lto-wrapper.exe -plugin-opt=-fresolution=vstavka.res -plugin-opt=-pass-through=-lgcc -plugin-opt=-pass-through=-lc -plugin-opt=-pass-through=-lgloss -plugin-opt=-pass-through=-lgcc --sysroot=c:\users\dmitry\desktop\korenev\podgotovka\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\bin\../riscv64-unknown-elf -melf32lriscv c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib/rv32i/ilp32/crt0.o c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/rv32i/ilp32/crtbegin.o -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/rv32i/ilp32 -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib/rv32i/ilp32 -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../riscv64-unknown-elf/lib/rv32i/ilp32 -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0 -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib -Lc:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../riscv64-unknown-elf/lib vstavka.o -lgcc --start-group -lc -lgloss --end-group -lgcc c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/rv32i/ilp32/crtend.o

c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/bin/ld.exe: c:/users/dmitry/desktop/korenev/podgotovka/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32/bin/../lib/gcc/riscv64-unknown-elf/8.3.0/../../../../riscv64-unknown-elf/lib/rv32i/ilp32/crt0.o: in function `.L0 ':

(.text+0x38): undefined reference to `main'

collect2.exe: error: ld returned 1 exit status

Шаги сборки программы:

1. Запускается первая фаза сборки, т.е. программа cc1.exe с параметрами:

-E -quiet -v -imultilib rv32i/ilp32 -iprefix

Полныйсинтаксвызова:

cc1.exe -E -quiet -v -imultilib rv32i/ilp32 –iprefix

1. Запуск программы *cc1* с параметром *“-fpreprocessed”*. Исполняемая команда в упрощенном виде:

cc1.exe -fpreprocessed vstavka.i -quiet -dumpbase vstavka.c -march=rv32i -mabi=ilp32 -auxbase vstavka -O1 -version -o vstavka.s

На данном шаге выполняется компиляция файла *“gray.i”*, уже обработанного препроцессором (опция “-fpreprocessed”), результат работы компилятора – код на языке ассемблера – сохраняется в файле *”\*.s”*.

1. Запуск программы *as*. Исполняемая команда в упрощенном виде:

as.exe -v --traditional-format -march=rv32i -mabi=ilp32 -o vstavka.o vstavka.s

На данном шаге выполняется ассемблирование файла *“vstavka.s”*, результат работы ассемблера – объектный код – сохраняется в файле *”vstavka.o”*.

1. Запуск программы collect2. Исполняемая команда в упрощенном виде:

/collect2.exe -plugin c:/users/dmitry/desktop/../ilp32/crtend.o

Программа collect2 является утилитой *gcc*, запускающей компоновщик. На данном шаге выполняется компоновка – формирование исполнимого файла из ранее созданных объектных файлов.

1. **Выход препроцессора (файл vstavka.i)**

# 797 "c:\\users\\dmitry\\desktop\\korenev\\podgotovka\\riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-w64-mingw32\\riscv64-unknown-elf\\include\\stdio.h" 3

# 2 ".\\vstavka.c" 2

# 1 ".\\vstavka.h"

# 3 ".\\vstavka.h"

int sorting\_by\_insertion(int \*array, int array\_lend);

# 3 ".\\vstavka.c" 2

int sorting\_by\_insertion(int \*array, int array\_lend){

int k;

int save\_element;

int i;

for(i=1; i<array\_lend; i++){

k =i;

while(k>0 && array[k-1]>array[k]){

save\_element = array[k-1];

array[k-1] = array[k];

array[k] = save\_element;

k -= 1;

}

}

return array[k];

}

в начале файла содержится порядка 1200 строк с инструкциями по линковке stdio.h к проекту, а затем следует код на C, который соответствует программе.

1. **Выход компилятора (vstavka.s)**

Компилятором сгенерирован код на языке низкого уровня – Assembly

.file "vstavka.c"

.option nopic

.attribute arch, "rv32i2p0"

.attribute unaligned\_access, 0

.attribute stack\_align, 16

.text

.align 2

.globl sorting\_by\_insertion

.type sorting\_by\_insertion, @function

sorting\_by\_insertion:

li a5,1

ble a1,a5,.L2

mv t5,a0

addi t3,a0,4

li t4,1

j .L3

.L6:

mv a4,t4

.L4:

addi t4,t4,1

addi t5,t5,4

addi t3,t3,4

beq a1,t4,.L2

.L3:

ble t4,zero,.L6

mv a7,t5

lw a2,0(t5)

mv a6,t3

lw a3,0(t3)

mv a5,t3

mv a4,t4

addi t1,t3,-8

sub t1,t1,t5

ble a2,a3,.L4

.L5:

sw a3,0(a7)

sw a2,0(a6)

addi a4,a4,-1

beq a4,zero,.L4

add a7,a7,t1

lw a2,-8(a5)

addi a6,a6,-4

addi a5,a5,-4

lw a3,0(a5)

bgt a2,a3,.L5

j .L4

.L2:

slli a4,a4,2

add a4,a0,a4

lw a0,0(a4)

ret

.size sorting\_by\_insertion, .-sorting\_by\_insertion

.ident "GCC: (SiFive GCC 8.3.0-2020.04.1) 8.3.0"

1. **Выход ассемблера – объектный файл.**

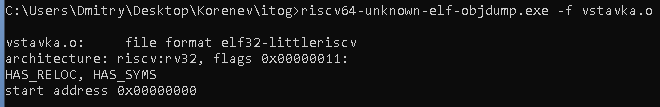
Сформированный объектный файл “gray.o” должен содержать коды инструкций, таблицу символов и таблицу перемещений. Бинарный файл для его чтения используем утилиту objdump.

Вывод утилиты:

Файл имеет формат ELF.

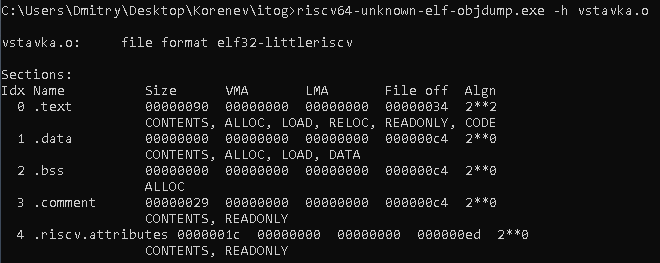
* содержит символы (флаг HAS\_SYMS)
* содержит таблицу перемещений (флаг HAS\_RELOC)
* не содержит адрес точки входа

riscv64-unknown-elf-objdump.exe -f vstavka.o



Следующая команда обеспечивает отображение заголовков секций файла ”vstavka.o”:

riscv64-unknown-elf-objdump.exe -h vstavka.o



В файле “vstavka*.o”* имеются следующие секции:

*.text* – секция кода, в которой содержатся коды инструкций (название секции обусловлено историческими причинами);

.*data* – секция инициализированных данных;

*.bss* – секция неинициализированных статических переменных (название секции также обусловлено историческими причинами);

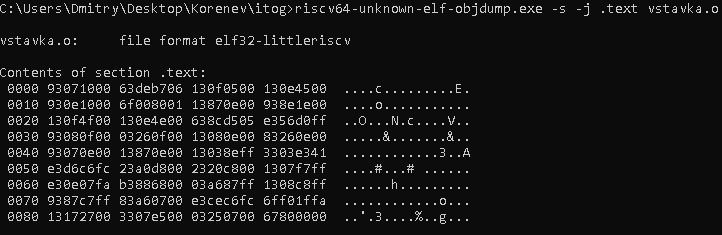
*.rodata* – аналог .data для неизменяемых данных

*.comment* – секция данных о версиях размером 12 байт

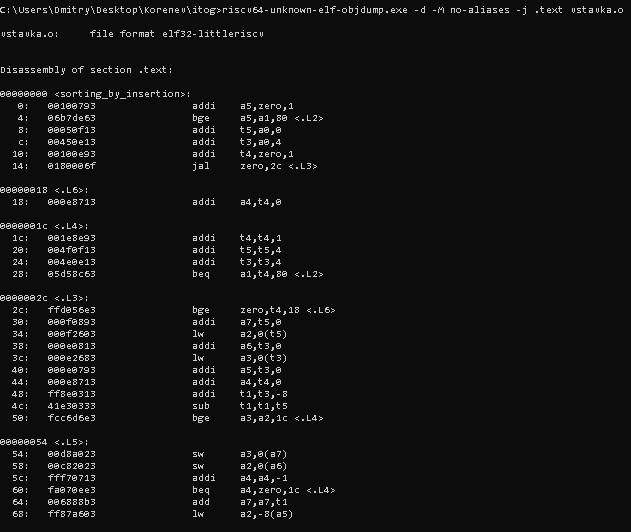
.riscv.attributes – информация про RISC-V

1. **Содержимое секции .text (файл vstavka.o)**

riscv64-unknown-elf-objdump.exe -s -j .text vstavka.o



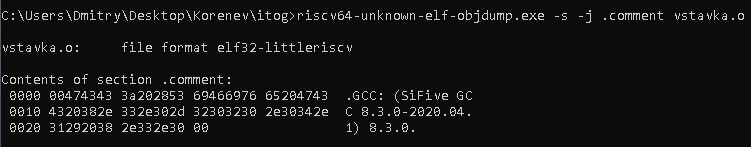
Разумеется, процедура декодирования кодов инструкций является «механической» (иначе как бы ее реализовывало техническое устройство – процессор), следовательно, разумно поручить ее выполнение ЭВМ:



* Опция *“-d”* инициирует процесс дизассемблирования
* опция *“-M no-aliases”* требует использовать в выводе только инструкции системы команд (но не псевдоинструкции ассемблера).

1. **Секция .comment (файл vstavka.o)**

**riscv64-unknown-elf-objdump.exe -s -j .comment vstavka.o**

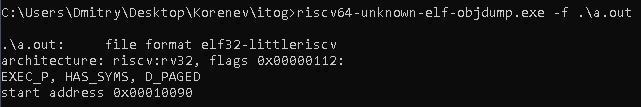


В секции .comment, записаны инициалы компилятора, которым проводилась процедура.

1. **Вывод компоновщика – исполняемый файл**

Сформированный компоновщиком файл *“a.out”* является «бинарным», и для изучения его содержимого будем пользоваться утилитой *objdump*:

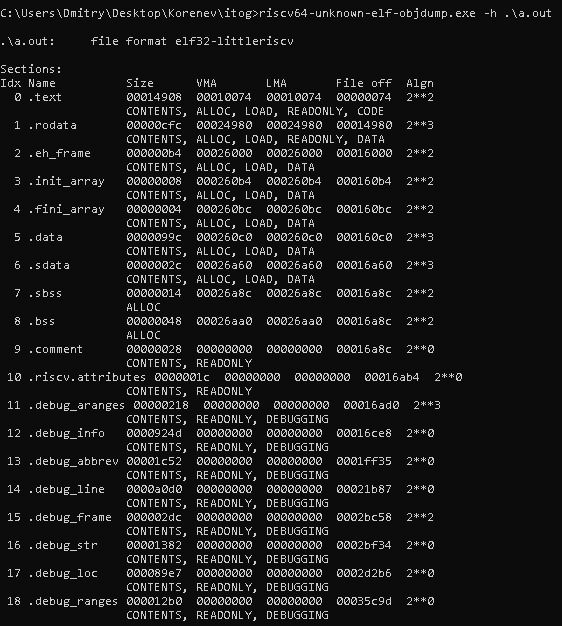
riscv64-unknown-elf-objdump.exe -f .\a.out



* формат файла – ELF
* EXEC\_P –файл исполняемый
* После загрузки файла его выполнение начинается с адреса 0x00010090

Посмотрим секции файла:

riscv64-unknown-elf-objdump.exe -h .\a.out



Состав секций *“a.out”* значительно расширен по сравнению с *“vstavka.o”*.

Изучим содержимое секции “.text” результат будет записан в файл a.ds

riscv64-unknown-elf-objdump.exe -j .text -d -M no-aliases a.out >a.ds



00010090 <\_start>:

10090: 00017197 auipc gp,0x17

10094: 83018193 addi gp,gp,-2000 # 268c0 <\_\_global\_pointer$>

10098: 1cc18513 addi a0,gp,460 # 26a8c <\_edata>

1009c: 22818613 addi a2,gp,552 # 26ae8 <\_\_BSS\_END\_\_>

100a0: 40a60633 sub a2,a2,a0

100a4: 00000593 addi a1,zero,0

100a8: 2f4000ef jal ra,1039c <memset>

100ac: 00000517 auipc a0,0x0

100b0: 1fc50513 addi a0,a0,508 # 102a8 <\_\_libc\_fini\_array>

100b4: 1ac000ef jal ra,10260 <atexit>

100b8: 250000ef jal ra,10308 <\_\_libc\_init\_array>

100bc: 00012503 lw a0,0(sp)

100c0: 00410593 addi a1,sp,4

100c4: 00000613 addi a2,zero,0

100c8: 07c000ef jal ra,10144 <main>

100cc: 1a80006f jal zero,10274 <exit>

00010144 <main>:

10144: fd010113 addi sp,sp,-48

10148: 02112623 sw ra,44(sp)

1014c: 02812423 sw s0,40(sp)

10150: 02912223 sw s1,36(sp)

10154: 03212023 sw s2,32(sp)

10158: 000257b7 lui a5,0x25

1015c: 98078793 addi a5,a5,-1664 # 24980 <\_\_clzsi2+0x50>

10160: 0007a583 lw a1,0(a5)

10164: 0047a603 lw a2,4(a5)

10168: 0087a683 lw a3,8(a5)

1016c: 00c7a703 lw a4,12(a5)

10170: 0107a783 lw a5,16(a5)

10174: 00b12623 sw a1,12(sp)

10178: 00c12823 sw a2,16(sp)

1017c: 00d12a23 sw a3,20(sp)

10180: 00e12c23 sw a4,24(sp)

10184: 00f12e23 sw a5,28(sp)

10188: 00500593 addi a1,zero,5

1018c: 00c10513 addi a0,sp,12

10190: 040000ef jal ra,101d0 <sorting\_by\_insertion>

10194: 00c10413 addi s0,sp,12

10198: 02010913 addi s2,sp,32

1019c: 000254b7 lui s1,0x25

101a0: 00042583 lw a1,0(s0)

101a4: 99448513 addi a0,s1,-1644 # 24994 <\_\_clzsi2+0x64>

101a8: 314000ef jal ra,104bc <printf>

101ac: 00440413 addi s0,s0,4

101b0: ff2418e3 bne s0,s2,101a0 <main+0x5c>

101b4: 00000513 addi a0,zero,0

101b8: 02c12083 lw ra,44(sp)

101bc: 02812403 lw s0,40(sp)

101c0: 02412483 lw s1,36(sp)

101c4: 02012903 lw s2,32(sp)

101c8: 03010113 addi sp,sp,48

101cc: 00008067 jalr zero,0(ra)

000101d0 <sorting\_by\_insertion>:

101d0: 00100793 addi a5,zero,1

101d4: 06b7de63 bge a5,a1,10250 <sorting\_by\_insertion+0x80>

101d8: 00050f13 addi t5,a0,0

101dc: 00450e13 addi t3,a0,4

101e0: 00100e93 addi t4,zero,1

101e4: 0180006f jal zero,101fc <sorting\_by\_insertion+0x2c>

101e8: 000e8713 addi a4,t4,0

101ec: 001e8e93 addi t4,t4,1

101f0: 004f0f13 addi t5,t5,4

101f4: 004e0e13 addi t3,t3,4

101f8: 05d58c63 beq a1,t4,10250 <sorting\_by\_insertion+0x80>

101fc: ffd056e3 bge zero,t4,101e8 <sorting\_by\_insertion+0x18>

10200: 000f0893 addi a7,t5,0

10204: 000f2603 lw a2,0(t5)

10208: 000e0813 addi a6,t3,0

1020c: 000e2683 lw a3,0(t3)

10210: 000e0793 addi a5,t3,0

10214: 000e8713 addi a4,t4,0

10218: ff8e0313 addi t1,t3,-8

1021c: 41e30333 sub t1,t1,t5

10220: fcc6d6e3 bge a3,a2,101ec <sorting\_by\_insertion+0x1c>

10224: 00d8a023 sw a3,0(a7)

10228: 00c82023 sw a2,0(a6)

1022c: fff70713 addi a4,a4,-1

10230: fa070ee3 beq a4,zero,101ec <sorting\_by\_insertion+0x1c>

10234: 006888b3 add a7,a7,t1

10238: ff87a603 lw a2,-8(a5)

1023c: ffc80813 addi a6,a6,-4

10240: ffc78793 addi a5,a5,-4

10244: 0007a683 lw a3,0(a5)

10248: fcc6cee3 blt a3,a2,10224 <sorting\_by\_insertion+0x54>

1024c: fa1ff06f jal zero,101ec <sorting\_by\_insertion+0x1c>

10250: 00271713 slli a4,a4,0x2

10254: 00e50733 add a4,a0,a4

10258: 00072503 lw a0,0(a4)

1025c: 00008067 jalr zero,0(ra)

“\_start” – “точка входа” в программу

1. **Сборка простейшей программы «по шагам»**

Для выполнения шагов будем запускать драйвер компилятора, и контролировать его действия, используя флаг “-v”.

**А. Препроцессирование**

Препроцессирование файла исходного текста (файла “vstavka.c”), результат записывается в файл “vstavka.i”:

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -E vstavka.c -o vstavka.i >log 2>&1



Параметры:

* -march=rv32i, -mabi=ilp32 – целевым является процессор с базовой архитектурой системы команд RV32I
* -O1 – выполнять простые оптимизации генерируемого кода (мы используем эту опцию в примерах, потому что обычно генерируемый код получается более простым)
* -v – печатать (в стандартный поток ошибок) выполняемые драйвером команды, а также дополнительную информацию
* -o – путь к выходному файлу
* -E – остановить процесс сборки после препроцессирования

**B. Компиляция**

Компиляця файла “vstavka.i”, сохраняем сгенерированный код на языке ассемблера – в файл “vstavka.s”.

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -S vstavka.i -o vstavka.s >log 2>&1



Параметры:

* -S – остановить процесс сборки после компиляции, не запуская ассемблер
* -fpreprocessed – выполняется компиляция файла, уже обработанного препроцессором

**C. Ассемблирование**

Ассемблирование файла “vstavka.s” => vstavka.o

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -c vstavka.s -o vstavka.o >log 2>&1



* -c – остановить процесс сборки после ассемблирования

**D. Компоновка**

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v main.o vstavka.exe >log 2>&1



**6. Раздельная компиляция**

**a. сборка программы**

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v main.c vstavka.c >log 2>&1



Препроцессирование, компиляция и ассемблирование выполнялось по шагам, выполним все стадии обработки исходного файла, получив в результате объектный файл.

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -c vstavka.c -o vstavka.o



* *“-c”* - приводит к останову процесса сборки после ассемблирования, т.е. после формирования объектного файла

**Скомпоновываем файл с программой**

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -c main.c -o vstavka.o



**d. Создание и использование статической библиотеки**

Статическая библиотека (static library) является, по сути, архивом (набором, коллекцией) объектных файлов, среди которых компоновщик выбирает «полезные» для данной программы: объектный файл считается «полезным», если в нем определяется еще не разрешенный компоновщиком символ.

Поместим vstavka.o в такой архив:

riscv64-unknown-elf-ar.exe -rsc vs.a vstavka.o



Параметры:

* -r – заменить старые файлы с такими названиями (*gray.o*), если они уже есть в архиве
* -s – записать «index» в архив. Index – это список всех символов, объявленных во включенных в архив объектных файлах, и его присутствие ускоряет линковку
* -с – создать архив, если его еще не было

Результирующим файлом является “vs.a” (“.a” – от “archive”).

Используем статическую библиотеку для сборки программ:

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 --save-temps main.c .\vs.a -o static.out



Изучим таблицы символов полученных исполняемых файлов

riscv64-unknown-elf-objdump.exe -t .\static.out > \_symtable



.\static.out: file format elf32-littleriscv

SYMBOL TABLE:

00010074 l d .text 00000000 .text

00024980 l d .rodata 00000000 .rodata

00026000 l d .eh\_frame 00000000 .eh\_frame

000260b4 l d .init\_array 00000000 .init\_array

000260bc l d .fini\_array 00000000 .fini\_array

000260c0 l d .data 00000000 .data

00026a60 l d .sdata 00000000 .sdata

00026a8c l d .sbss 00000000 .sbss

00026aa0 l d .bss 00000000 .bss

00000000 l d .comment 00000000 .comment

00000000 l d .riscv.attributes 00000000 .riscv.attributes

00000000 l d .debug\_aranges 00000000 .debug\_aranges

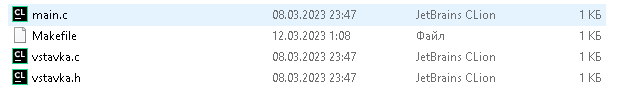
00000000 l d .debug\_info 00000000 .debug\_info

**3. Makefile**

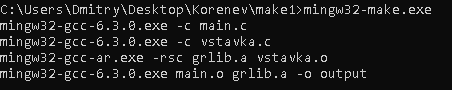
Makefile - это набор инструкций для программы make, которая позволяет собирать проекты, состоящие из большого числа “\*.c” и “\*.h” файлов. Обычно эта программа используется в связке с системами сборки, например cmake, позволяя вести проекты модульно (т.е. проект с включенными подпроектами).

Сборка с помощью Makefile:

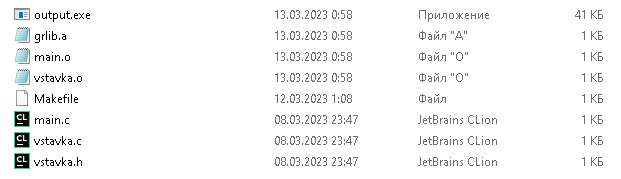
Начальные файлы



Mingw32-make.exe



Созданные файлы



Демонстрация работы



Содержание Makefile

output: main.o grlib.a

mingw32-gcc-6.3.0.exe main.o grlib.a -o output

main.o: main.c

mingw32-gcc-6.3.0.exe -c main.c

grlib.a: vstavka.o vstavka.h

mingw32-gcc-ar.exe -rsc grlib.a vstavka.o

vstavka.o:

mingw32-gcc-6.3.0.exe -c vstavka.c

clean:

rm .o.a \*.exe

Что происходит в Makefile:

1. Создаём объектный файл *main.o* из исходного *main.c*
2. Создаём объектный файл *vstavka.o* из исходного *vstavka.c*
3. Архивируем объектный файл gray.o (создаём статическую библиотеку *grlib.a*)
4. Компонуем статическую библиотеку *grlib.a* с объектным файлом *main.o* и получаем исполняемый файл *output*

Все использованные команды

riscv64-unknown-elf-gcc-8.3.0.exe .\vstavka.c --save-temps -march=rv32i -mabi=ilp32 -O1 -v >log 2>&1

Файл log (процесс сборки программы)

riscv64-unknown-elf-gcc --save-temps -march=rv32i -mabi=ilp32 -O1 -v main.c vstavka.c >log 2>&1

(создает все файлы main.i, zero.i;

main.s, zero.s;

main.o, zero.o;

a.out;

log.)

riscv64-unknown-elf-objdump -t vstavka.o main.o

(Таблица символов)

riscv64-unknown-elf-objdump –d –M no-aliases –j .text vstavka.o main.o

(диссассемблирование секции text )(Дизассемблирование – преобразование программы на машинном языке к ее ассемблерному представлению.)

riscv64-unknown-elf-objdump.exe -f vstavka.o

(содержание что в нем находится)

riscv64-unknown-elf-objdump.exe -h vstavka.o

(Секции объектных файлов vstavka.o => итоговый файл с помощью компановки)

riscv64-unknown-elf-objdump.exe -s -j .text vstavka.o

(Изучим содержимое секции “.text” файла vstavka.o)

riscv64-unknown-elf-objdump.exe -d -M no-aliases -j .text vstavka.o

(декодирование инструкиции Секция text файлов vstavka.o при помощи Эвм)

riscv64-unknown-elf-objdump.exe -s -j .comment vstavka.o

(Изучим содержимое секции “.comment”: файла vstavka.o)

6. вывод компановщика a.out

riscv64-unknown-elf-objdump.exe -f .\a.out

(содержимое a.out)

riscv64-unknown-elf-objdump.exe -h .\a.out

(изучаем секции файла a.out)

riscv64-unknown-elf-objdump.exe -j .text -d -M no-aliases a.out >a.ds

(результат компановки переведенный в текст (изучим содержимое секции .text => a.ds))

сборка программы

драйвер компилятора, и контролировать его действия, используя флаг “-v”.

А. Препроцессирование

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -E vstavka.c -o vstavka.i >log 2>&1

Препроцессирование

Компиляция

Далее необходимо выполнить компиляцию файла “gray.i”, сохранив результат – сгенерированный код на языке ассемблера – в файл “gray.s”.

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -S vstavka.i -o vstavka.s >log 2>&1

C. Ассемблирование

Ассемблирование файла “gray.s”

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v -c vstavka.s -o vstavka.o >log 2>&1

D. Компоновка

//riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v main.o vstavka.exe >log 2>&1

riscv64-unknown-elf-gcc-ar.exe -rsc vstav.a vstavka.o

(создаем библиотеку vstav.a)

3. Раздельная компиляция

сборка программы

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -v main.c vstavka.c >log 2>&1

(создание файлов a.out log)

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -c vstavka.c -o vstavka.o

(создание vstavka.o выполнить все стадии обработки исходного файла, получив в результате объектный файл)

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 -c main.c -o vstavka.o

(После создания объектного файла его нужно скомпоновать с нашей программой)

d. Создание и использование статической библиотеки

riscv64-unknown-elf-ar.exe -rsc vstav.a vstavka.o (помещаем vstavka.o в статическую библиотеку)

riscv64-unknown-elf-gcc-8.3.0.exe -march=rv32i -mabi=ilp32 -O1 --save-temps main.c .\vstav.a -o static.out

Используем статическую библиотеку для сборки программ (создание файлов main.o main.s main.i static.out)

riscv64-unknown-elf-objdump.exe -t .\static.out > \_symtable

(таблицы символов полученных исполняемых файлов) сборки объектных файлов была возложена на компоновщик (а не нас).

3. Makefile

mingw32-make.exe

4. Вывод

В данной лабораторной работе мы познакомились с процессом сборки проекта на языке C.

Он состоит из:

1. **Препроцессирования**: исходного *.c* файл препроцессируем в *.i* файл
2. **Компиляции**: полученный *.i*файл компилируется в файл ассемблера *.s*
3. **Ассемблирования**: файл *.s* асссемблируется в объектный файл *.o*
4. **Компоновки**: объектный файл *.o*компонуется в исполняемый файл

Также мы ознакомились в *makefile’ами,* которые упрощают процесс сборки.

Утилита Make позволяет собирать проекты, состоящие из большого количества файлов, вместо использования PS/SH скриптов, и прописывания файлов вручную.