Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Лабораторная работа № 2**

Программирование RISC-V

по дисциплине «Низкоуровневое программирование»

Выполнил

студент гр. 3530901/90004

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гасилов Д.О.

(подпись)

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Алексюк А.О.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Санкт-Петербург   
2021

**Задача**

1. Изучить методические материалы, опубликованные на сайте курса.

2. Установить пакет средств разработки “SiFive GNU Embedded Toolchain” для RISC-V.

3. 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.

4. 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 и исполняеммом файле.

5. Выделить разработанную функцию в статическую библиотеку. Разработать make-файлы для сборки библиотеки и использующей ее тестовой программы. Проанализировать ход сборки библиотеки и программы, созданные файлы зависимостей.

**Вариант задания**

Вариант: 9 - Расчет биномиальных коэффициентов для данного показателя по треугольнику Паскаля.  
  
Алгоритм был описан в предыдущих работах, поэтому оставим его без дополнительных комментариев

**Листинг программы**

Файл main.c  
#include "coeffs.h"  
#include <stdio.h>  
#include <stdlib.h>  
  
int main() {  
 int param = 5;  
 int i;  
 int \*array = (int \*)calloc(param + 1, sizeof(int));  
 array[0] = 1;  
 coeffs(param, array);  
 for (i = 0; i <= param ; i++) {  
 printf("%i \n", array[i]);  
 }  
 return 0;  
}

Файл coeffs.c  
  
int coeffs(int param, int \*array){  
 int i;  
 int j;  
 for(i=1; i<param; i++){  
 for(j=i; j>0; j--){  
 array[j]=array[j]+array[j-1];  
 }  
 }  
 return 0;  
}

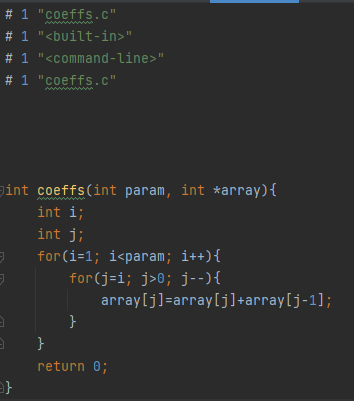
Файл сoeffs.h  
#ifndef COEFFS\_COEFFS\_H  
#define COEFFS\_COEFFS\_H  
int coeffs(int param, int \*array);  
#endif //COEFFS\_COEFFS\_H

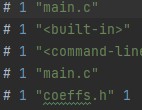
**Сборка программы**

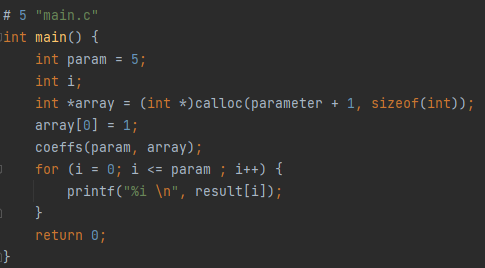
Приступим к пошаговой сборке написанных на языке C программ. Сначала препроцессируем исходные файлы coeffs.c и main.c в файлы coeffs.i и main.i испльзуя следующие команды:

**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
riscv64-unknown-elf-gcc.exe -march=rv32i -mabi=ilp32 -O1 -v -E main.c -o main.i  
  
C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
riscv64-unknown-elf-gcc.exe -march=rv32i -mabi=ilp32 -O1 -v -E coeffs.c -o coeffs.i**

Драйвер компилятора gcc riscv64-unknown-elf-gcc запускается с параметрами командной строки «-march=rv32i» и «-mabi=ilp32», которые указывают, что целевым является процессор с базовой архитектурой системы команд RV32I. Параметр «-O1» указывает, что надо выполнять простые оптимизации генерируемого кода, параметр «-E» – что надо остановить процесс сборки после препроцессирования.







Между началом файла и кодом программы “main.i’’ находится примерно 1500 строк кода с инструкциями по линковке имеющейся в файле “main.c” директивы “#include <stdio.h>” и “#include stdlib.h”  
  
Появившиеся нестандартные директивы, начинающиеся с символа “#”, используются для передачи информации об исходном тексте из препроцессора в компилятор. В файле “main.i” четвертая директива «# 1 “main.c”» информирует компилятор о том, что следующая строка является результатом обработки первой строки исходного файла “main.c”, где была прописана команда “#include “сoeffs.h”, поэтому препроцессор произвел вставку содержимого этого заголовочного файла, то есть определение функции сoeffs. Описание самого содержимого файла начинается после директивы “# 5 “main.c”. Исходный код тестирующей функции main после работы препроцессора остался без изменений, также как и исходный код функции сoeffs из файла сoeffs.i  
  
Далее проведем компиляцию файлов coeffs.i и main.i в код на языке ассемблера coeffs.s и main.s:  
  
**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
riscv64-unknown-elf-gcc.exe -march=rv32i -mabi=ilp32 -O1 -s main.i -o main.s  
  
C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
riscv64-unknown-elf-gcc.exe -march=rv32i -mabi=ilp32 -O1 -s coeffs.i -o coeffs.s**

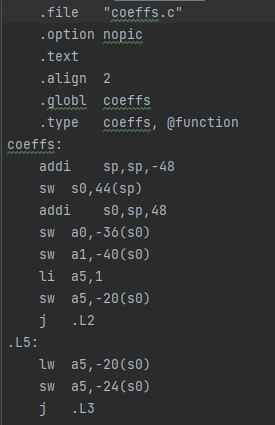
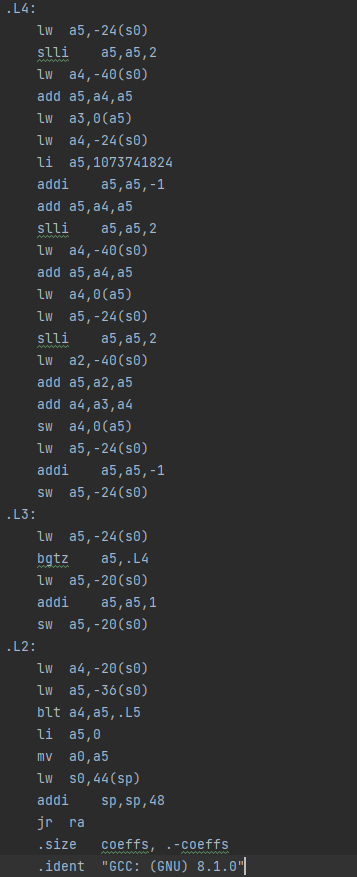
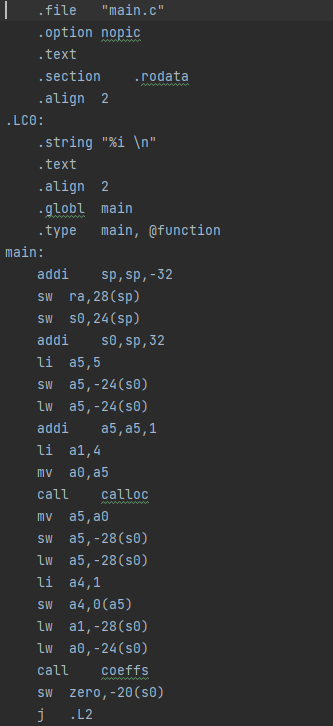
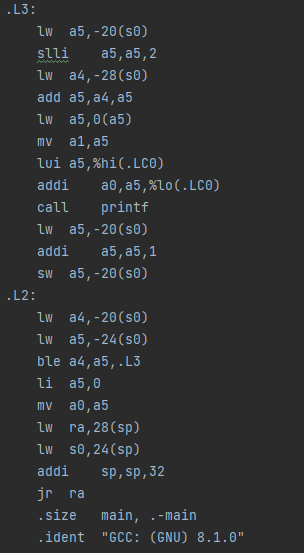
Драйвер компилятора riscv64-unknown-elf-gcc запускается с параметрами, где параметр “-S” указывает на необходимость остановить процесс сборки после компиляции .   
  


Рис.3. ''coeffs.s''

Как мы можем заметить, начиная с метки coeffs начинается тело функции, выполняющей нахождение биноминальных коэффициентов 

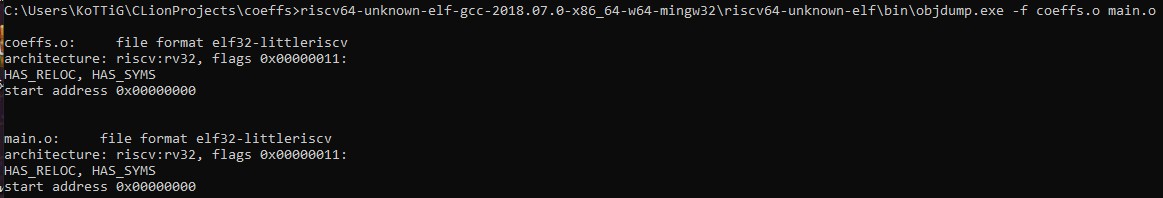
  
Рис.4. ''main.s''

По метке main начинается тестовая программа, которая внутри себя уже вызывает функцию coeffs, однако эта функция ещё не определена, так как еще не было проведено ассемблирование.  
  
Далее проведем ассемблирование файлов coeffs.s и main.s в объектные файлы coeffs.o и main.o

**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
  
C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\  
riscv64-unknown-elf-gcc.exe -march=rv32i -mabi=ilp32 -O1 -c coeffs.s -o coeffs.o**

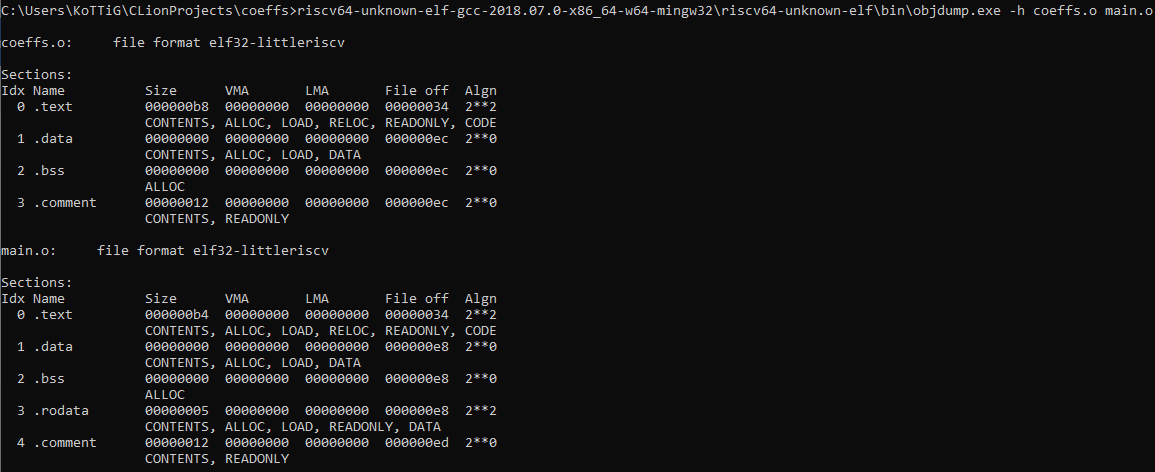
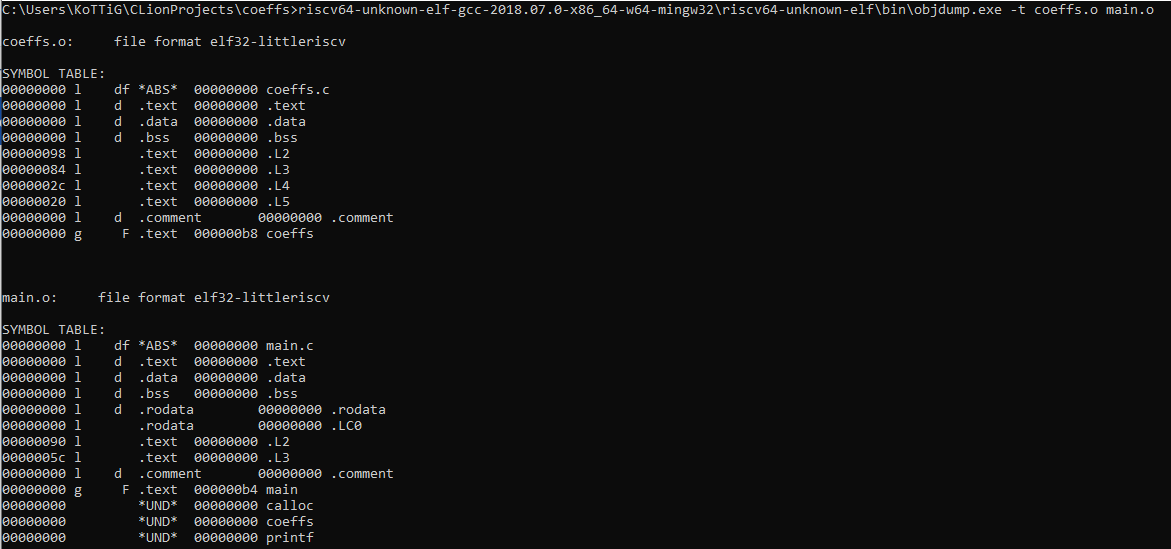
Драйвер компилятора riscv64-unknown-elf-gcc запускается с параметрами командной строки, где параметр “-c” указывает на необходимость остановить процесс сборки после ассемблирования.

Так как объектные файлы не являются текстовыми, для просмотра их содержимого используется objdump:

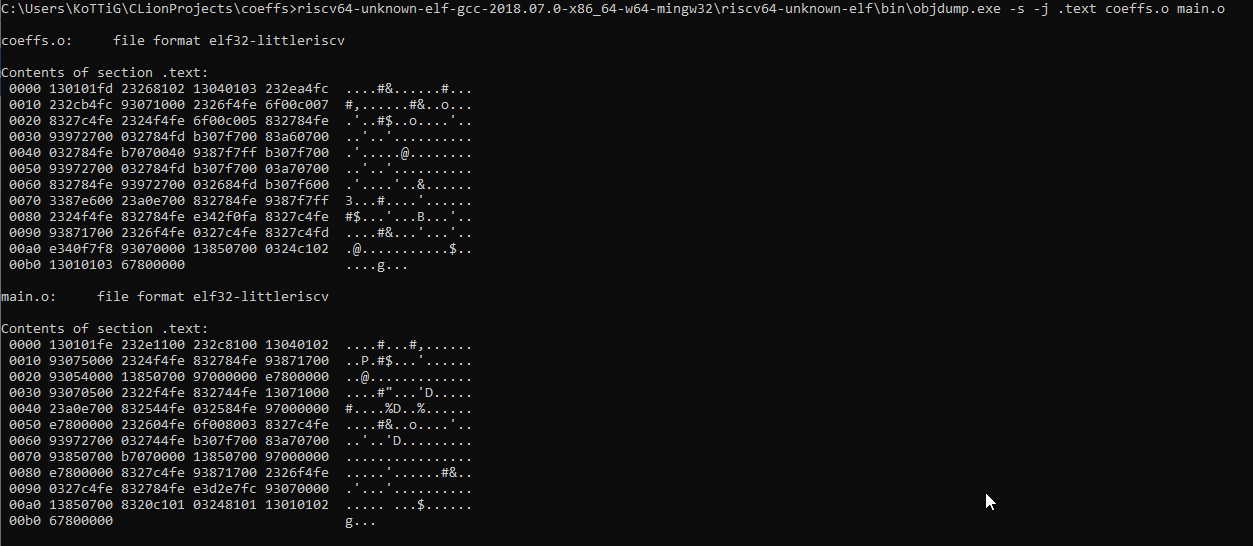
****

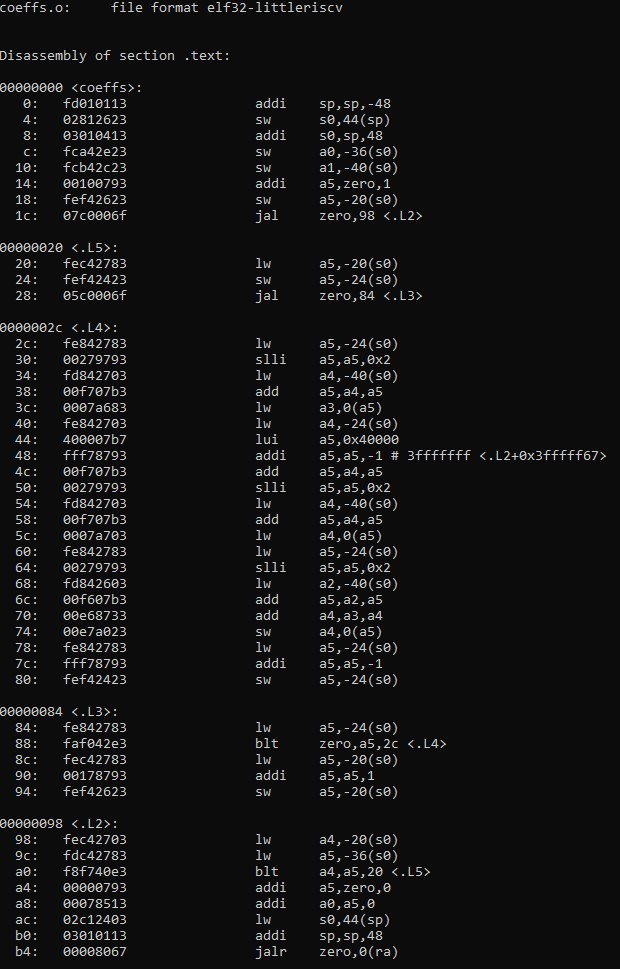
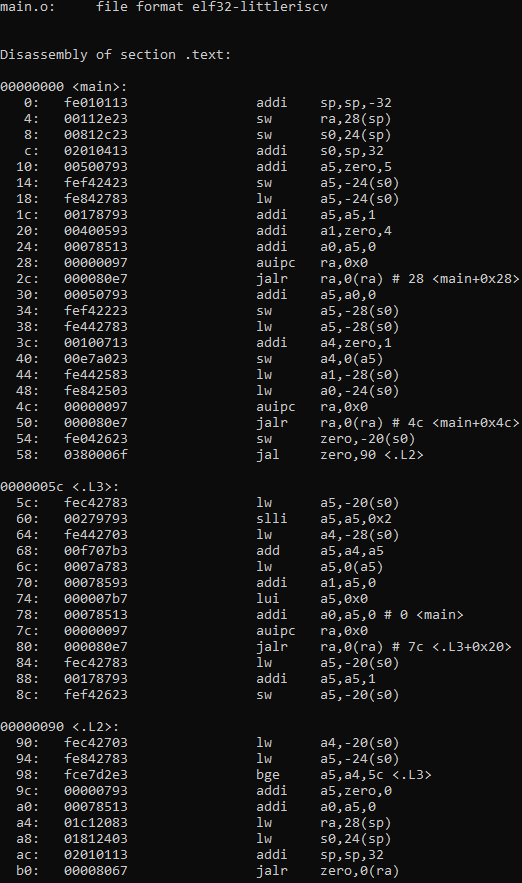
Оба файла содержат таблицу перемещений и таблицу символов (об этом нам говорят флаги HAS\_RELOC и HAS\_SYMS).

Выведем в консоль заголовки секций объектных файлов:

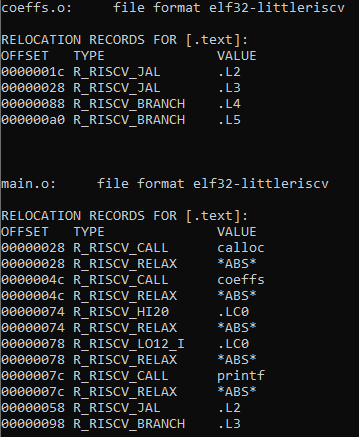
  
Далее просмотрим таблицы символов тех же файлов:  
  


Проанализируем секции .text объектных файлов, в которых содержатся коды инструкций:



Процедура декодирования кодов инструкций является «механической», следовательно, разумно поручить ее выполнение ЭВМ, введя параметры “-d -M no-aliases”:  
  
  


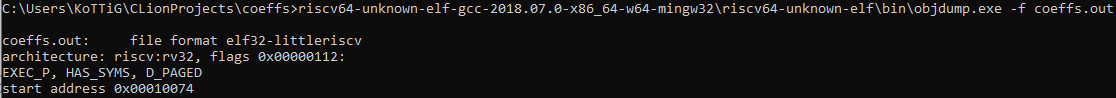
Дизассемблированный код очень похож на сгенерированный за исключением псевдо-инструкций.

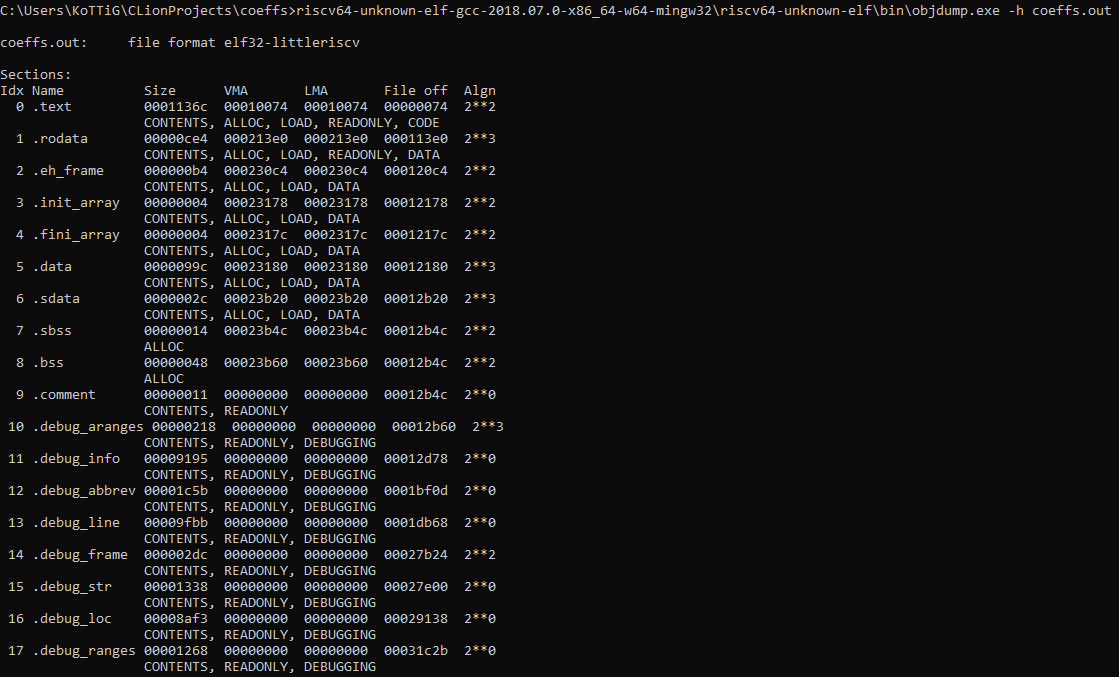
Рассмотрим таблицы перемещений объектных файлов:  
  


В таблицах перемещения fib.o содержится информация о ветвлениях(R\_RISCV\_BRANCH).

В таблицах перемещения main.o, есть R\_RISCV\_CALL. Записи типа R\_RISCV\_RELAX заносятся в таблицу перемещений в дополнение к записям типа R\_RISCV\_CAL.

Далее выполним компоновку и формирование исполняемых файлов программ:  
  
**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\riscv64-unknown-elf-gcc-8.1.0.exe -march=rv32i -mabi=ilp32 main.o coeffs.o -o coeffs.out**

Сформированный компоновщиком файл “ coeffs.out”, также является «бинарным», так что рассматриваем его с помощью “objdump”.  
  


Флаг EXEC\_P указывает, что файл действительно является исполняемым, после загрузки его выполнение должно начаться с адреса 0x00010074  
  
Откроем секции исполняемого файла:  
  


В данном файле производится слияние содержания секций обоих объектных файлов, а также расширение списка секций новыми блоками.  
  
  
Откроем таблицу символов исполняемого файла:  
  
SYMBOL TABLE:

00010074 l d .text 00000000 .text

000213e0 l d .rodata 00000000 .rodata

000230c4 l d .eh\_frame 00000000 .eh\_frame

00023178 l d .init\_array 00000000 .init\_array

0002317c l d .fini\_array 00000000 .fini\_array

00023180 l d .data 00000000 .data

00023b20 l d .sdata 00000000 .sdata

00023b4c l d .sbss 00000000 .sbss

00023b60 l d .bss 00000000 .bss

00000000 l d .comment 00000000 .comment

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

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

00000000 l d .debug\_abbrev 00000000 .debug\_abbrev

00000000 l d .debug\_line 00000000 .debug\_line

00000000 l d .debug\_frame 00000000 .debug\_frame

00000000 l d .debug\_str 00000000 .debug\_str

00000000 l d .debug\_loc 00000000 .debug\_loc

00000000 l d .debug\_ranges 00000000 .debug\_ranges

00000000 l df \*ABS\* 00000000 crtstuff.c

000230c4 l O .eh\_frame 00000000 \_\_EH\_FRAME\_BEGIN\_\_

000100b8 l F .text 00000000 deregister\_tm\_clones

000100e0 l F .text 00000000 register\_tm\_clones

0001011c l F .text 00000000 \_\_do\_global\_dtors\_aux

00023b60 l O .bss 00000001 completed.5435

0002317c l O .fini\_array 00000000 \_\_do\_global\_dtors\_aux\_fini\_array\_entry

00010164 l F .text 00000000 frame\_dummy

00023b64 l O .bss 00000018 object.5440

00023178 l O .init\_array 00000000 \_\_frame\_dummy\_init\_array\_entry

00000000 l df \*ABS\* 00000000 main.c

00000000 l df \*ABS\* 00000000 coeffs.c

00000000 l df \*ABS\* 00000000 atexit.c

00000000 l df \*ABS\* 00000000 calloc.c

00000000 l df \*ABS\* 00000000 mallocr.c

00000000 l df \*ABS\* 00000000 exit.c

00000000 l df \*ABS\* 00000000 fini.c

00000000 l df \*ABS\* 00000000 impure.c

00023180 l O .data 00000428 impure\_data

00000000 l df \*ABS\* 00000000 init.c

00000000 l df \*ABS\* 00000000 mallocr.c

00000000 l df \*ABS\* 00000000 mlock.c

00000000 l df \*ABS\* 00000000 printf.c

00000000 l df \*ABS\* 00000000 sbrkr.c

00000000 l df \*ABS\* 00000000 vfprintf.c

00012da4 l F .text 000000b8 \_\_sbprintf

00021554 l O .rodata 00000010 blanks.4513

00021564 l O .rodata 00000010 zeroes.4514

00000000 l df \*ABS\* 00000000 wsetup.c

00000000 l df \*ABS\* 00000000 \_\_atexit.c

00000000 l df \*ABS\* 00000000 \_\_call\_atexit.c

00000000 l df \*ABS\* 00000000 fflush.c

00000000 l df \*ABS\* 00000000 findfp.c

000133b0 l F .text 00000008 \_\_fp\_lock

000133b8 l F .text 00000088 std

0001344c l F .text 00000004 \_\_fp\_unlock

00000000 l df \*ABS\* 00000000 mallocr.c

00000000 l df \*ABS\* 00000000 fwalk.c

00000000 l df \*ABS\* 00000000 ldtoa.c

00013b5c l F .text 00000014 eclear

00013b70 l F .text 00000024 emov

00013b94 l F .text 00000014 ecleaz

00013ba8 l F .text 00000028 emovz

00013bd0 l F .text 0000003c ecmpm

00013c0c l F .text 00000050 eshdn1

00013c5c l F .text 00000054 eshup1

00013cb0 l F .text 00000034 eshdn8

00013ce4 l F .text 00000028 eshup8

00013d0c l F .text 00000020 eshup6

00013d2c l F .text 00000020 eshdn6

00013d4c l F .text 00000038 eaddm

00013d84 l F .text 00000038 esubm

00013dbc l F .text 000000dc m16m

00013e98 l F .text 00000034 eisnan

00013ecc l F .text 00000038 eisneg

00013f04 l F .text 000000d0 emovi

00013fd4 l F .text 00000108 ecmp

000140dc l F .text 00000034 eisinf

00014110 l F .text 0000002c einfin.isra.2

0001413c l F .text 00000140 eshift.part.3

0001427c l F .text 000000d8 enormlz

00014354 l F .text 00000020 enan.constprop.12

00014374 l F .text 00000070 emovo.isra.6

000143e4 l F .text 00000104 e113toe.isra.8

000144ec l F .text 0000031c emdnorm

00014808 l F .text 000000e4 eiremain

000148ec l F .text 00000290 emul

00014b7c l F .text 00000334 ediv

000215b0 l O .rodata 00000014 ezero

000215c4 l O .rodata 00000104 etens

000216c8 l O .rodata 00000014 eone

000216dc l O .rodata 00000022 bmask

00021700 l O .rodata 00000104 emtens

00000000 l df \*ABS\* 00000000 localeconv.c

00000000 l df \*ABS\* 00000000 makebuf.c

00000000 l df \*ABS\* 00000000 memchr.c

00000000 l df \*ABS\* 00000000 mprec.c

00021830 l O .rodata 0000000c p05.3308

00000000 l df \*ABS\* 00000000 reent.c

00000000 l df \*ABS\* 00000000 s\_frexp.c

00000000 l df \*ABS\* 00000000 sprintf.c

00000000 l df \*ABS\* 00000000 stdio.c

00000000 l df \*ABS\* 00000000 strcpy.c

00000000 l df \*ABS\* 00000000 strlen.c

00000000 l df \*ABS\* 00000000 strncpy.c

00000000 l df \*ABS\* 00000000 vfprintf.c

00021ac4 l O .rodata 00000010 blanks.4498

00021ad4 l O .rodata 00000010 zeroes.4499

00000000 l df \*ABS\* 00000000 vfprintf.c

00019e5c l F .text 000000b8 \_\_sbprintf

00021c50 l O .rodata 00000010 blanks.4489

00021c60 l O .rodata 00000010 zeroes.4490

00000000 l df \*ABS\* 00000000 writer.c

00000000 l df \*ABS\* 00000000 closer.c

00000000 l df \*ABS\* 00000000 fclose.c

00000000 l df \*ABS\* 00000000 fputwc.c

00000000 l df \*ABS\* 00000000 fstatr.c

00000000 l df \*ABS\* 00000000 fvwrite.c

00000000 l df \*ABS\* 00000000 isattyr.c

00000000 l df \*ABS\* 00000000 locale.c

00000000 l df \*ABS\* 00000000 lseekr.c

00000000 l df \*ABS\* 00000000 mbtowc\_r.c

00000000 l df \*ABS\* 00000000 memcpy.c

00000000 l df \*ABS\* 00000000 memmove.c

00000000 l df \*ABS\* 00000000 readr.c

00000000 l df \*ABS\* 00000000 mallocr.c

00000000 l df \*ABS\* 00000000 vfprintf.c

00021dec l O .rodata 00000010 blanks.4475

00021dfc l O .rodata 00000010 zeroes.4476

00000000 l df \*ABS\* 00000000 wbuf.c

00000000 l df \*ABS\* 00000000 wcrtomb.c

00000000 l df \*ABS\* 00000000 wctomb\_r.c

00000000 l df \*ABS\* 00000000 ctype\_.c

00000000 l df \*ABS\* 00000000 sys\_close.c

00000000 l df \*ABS\* 00000000 sys\_exit.c

00000000 l df \*ABS\* 00000000 sys\_fstat.c

00000000 l df \*ABS\* 00000000 sys\_isatty.c

00000000 l df \*ABS\* 00000000 sys\_lseek.c

00000000 l df \*ABS\* 00000000 sys\_read.c

00000000 l df \*ABS\* 00000000 sys\_sbrk.c

00023b5c l O .sbss 00000004 heap\_end.1850

00000000 l df \*ABS\* 00000000 sys\_write.c

00000000 l df \*ABS\* 00000000 sys\_conv\_stat.c

00000000 l df \*ABS\* 00000000 errno.c

00000000 l df \*ABS\* 00000000 libgcc2.c

00000000 l df \*ABS\* 00000000 libgcc2.c

00000000 l df \*ABS\* 00000000 divdf3.c

00000000 l df \*ABS\* 00000000 muldf3.c

00000000 l df \*ABS\* 00000000 eqtf2.c

00000000 l df \*ABS\* 00000000 getf2.c

00000000 l df \*ABS\* 00000000 letf2.c

00000000 l df \*ABS\* 00000000 multf3.c

00000000 l df \*ABS\* 00000000 subtf3.c

00000000 l df \*ABS\* 00000000 fixtfsi.c

00000000 l df \*ABS\* 00000000 floatsitf.c

00000000 l df \*ABS\* 00000000 extenddftf2.c

00000000 l df \*ABS\* 00000000 trunctfdf2.c

00000000 l df \*ABS\* 00000000 libgcc2.c

00000000 l df \*ABS\* 00000000 libgcc2.c

00000000 l df \*ABS\* 00000000 crtstuff.c

00023174 l O .eh\_frame 00000000 \_\_FRAME\_END\_\_

00000000 l df \*ABS\* 00000000

00023180 l .fini\_array 00000000 \_\_fini\_array\_end

0002317c l .fini\_array 00000000 \_\_fini\_array\_start

0002317c l .init\_array 00000000 \_\_init\_array\_end

00023178 l .init\_array 00000000 \_\_preinit\_array\_end

00023178 l .init\_array 00000000 \_\_init\_array\_start

00023178 l .init\_array 00000000 \_\_preinit\_array\_start

00016af8 g F .text 00000088 \_mprec\_log10

00016bd0 g F .text 00000068 \_\_any\_on

0001a6d4 g F .text 00000044 \_isatty\_r

00021908 g O .rodata 00000028 \_\_mprec\_tinytens

00017054 g F .text 0000001c strcpy

00016c38 g F .text 00000040 cleanup\_glue

0001a7f0 g F .text 0000004c \_lseek\_r

0001ded0 g F .text 00000148 .hidden \_\_getf2

0001de04 g F .text 000000cc .hidden \_\_eqtf2

00010cc4 g F .text 00000050 printf

00024320 g .sdata 00000000 \_\_global\_pointer$

0001c054 g F .text 0000008c \_wcrtomb\_r

00016ff4 g F .text 00000054 \_\_sseek

000134d8 g F .text 00000080 \_\_sinit

0001bf10 g F .text 00000130 \_\_swbuf\_r

0001a718 g F .text 0000007c \_setlocale\_r

00013454 g F .text 00000078 \_\_sfmoreglue

00010c80 g F .text 00000004 \_\_malloc\_unlock

00020bc4 g F .text 00000194 .hidden \_\_floatsitf

0001a990 g F .text 0000005c memmove

000134cc g F .text 0000000c \_cleanup

0001a7bc g F .text 00000020 \_\_locale\_ctype\_ptr

00015d08 g F .text 000000a8 \_Balloc

0001c4b8 g F .text 0000000c \_\_errno

0001c434 g F .text 00000084 \_conv\_stat

00015b10 g F .text 00000008 \_\_localeconv\_l

0001a22c g F .text 00000048 \_fstat\_r

00023ba4 g O .bss 00000004 errno

00016f68 g F .text 00000008 \_\_seofread

0001a8b8 g F .text 000000d8 memcpy

00023b1c g O .data 00000000 .hidden \_\_TMC\_END\_\_

00013440 g F .text 0000000c \_cleanup\_r

000170c4 g F .text 00001fd8 \_svfprintf\_r

00016a40 g F .text 000000b8 \_\_ratio

00010c84 g F .text 00000040 \_printf\_r

0001e018 g F .text 00000148 .hidden \_\_letf2

000212e8 g F .text 00000048 .hidden \_\_udivsi3

0001c1e0 g F .text 0000006c \_fstat

00021840 g O .rodata 000000c8 \_\_mprec\_tens

00023b54 g O .sbss 00000004 \_\_malloc\_top\_pad

00023b3c g O .sdata 00000000 .hidden \_\_dso\_handle

00015b18 g F .text 00000020 \_localeconv\_r

0001a274 g F .text 00000460 \_\_sfvwrite\_r

00016130 g F .text 00000034 \_\_i2b

00010240 g F .text 000000b8 coeffs

00010d14 g F .text 00000044 \_sbrk\_r

0001a9ec g F .text 0000004c \_read\_r

00019fa8 g F .text 000000dc \_fclose\_r

00013388 g F .text 00000028 fflush

00023b50 g O .sbss 00000004 \_\_malloc\_max\_sbrked\_mem

00016800 g F .text 00000104 \_\_b2d

0001ca98 g F .text 000004f0 .hidden \_\_umoddi3

0001c24c g F .text 00000038 \_isatty

00023b38 g O .sdata 00000004 \_global\_impure\_ptr

0001aa38 g F .text 00000550 \_realloc\_r

00010460 g F .text 000000a8 \_\_libc\_init\_array

0001c4c4 g F .text 000005d4 .hidden \_\_udivdi3

0001a1a8 g F .text 0000002c \_fputwc\_r

00021930 g O .rodata 00000028 \_\_mprec\_bigtens

00015f08 g F .text 00000108 \_\_s2b

0001c31c g F .text 000000cc \_sbrk

000165e0 g F .text 00000054 \_\_mcmp

00013658 g F .text 00000014 \_\_fp\_lock\_all

000100b4 g F .text 00000000 \_init

00010404 g F .text 0000005c \_\_libc\_fini\_array

00016c78 g F .text 00000110 \_reclaim\_reent

00016010 g F .text 00000070 \_\_hi0bits

0001030c g F .text 00000014 calloc

00020a74 g F .text 00000150 .hidden \_\_fixtfsi

000163a8 g F .text 000000fc \_\_pow5mult

00021330 g F .text 00000010 .hidden \_\_umodsi3

00021394 g F .text 0000004c .hidden \_\_clzsi2

00013648 g F .text 00000004 \_\_sfp\_lock\_acquire

00015ce4 g F .text 00000024 memchr

00016e40 g F .text 00000064 \_sprintf\_r

00013794 g F .text 00000274 \_free\_r

0001a794 g F .text 00000020 \_\_locale\_mb\_cur\_max

00013018 g F .text 00000110 \_\_call\_exitprocs

00023b44 g O .sdata 00000004 \_\_malloc\_sbrk\_base

00010074 g F .text 00000040 \_start

0001c284 g F .text 0000004c \_lseek

000164a4 g F .text 0000013c \_\_lshift

0001e018 g F .text 00000148 .hidden \_\_lttf2

0001de04 g F .text 000000cc .hidden \_\_netf2

0001b104 g F .text 000001bc \_\_ssprint\_r

00012f98 g F .text 00000080 \_\_register\_exitproc

00015aac g F .text 00000064 \_ldcheck

00016164 g F .text 00000244 \_\_multiply

000212bc g F .text 00000024 .hidden \_\_mulsi3

0001708c g F .text 00000038 strncpy

00023b7c g O .bss 00000028 \_\_malloc\_current\_mallinfo

00016904 g F .text 0000013c \_\_d2b

00020d58 g F .text 00000214 .hidden \_\_extenddftf2

00019f64 g F .text 00000044 \_close\_r

00012e60 g F .text 00000138 \_\_swsetup\_r

0001cf88 g F .text 000007ec .hidden \_\_divdf3

0001355c g F .text 000000ec \_\_sfp

00016b80 g F .text 00000050 \_\_copybits

000235a8 g O .data 00000408 \_\_malloc\_av\_

00013654 g F .text 00000004 \_\_sinit\_lock\_release

0001d774 g F .text 00000690 .hidden \_\_muldf3

00016f18 g F .text 00000050 \_\_sread

00010c7c g F .text 00000004 \_\_malloc\_lock

00013330 g F .text 00000058 \_fflush\_r

00010320 g F .text 000000b0 \_calloc\_r

00023b4c g .sbss 00000000 \_\_bss\_start

00010ba0 g F .text 000000dc memset

00010198 g F .text 000000a8 main

00023b4c g O .sbss 00000004 \_\_malloc\_max\_total\_mem

0001c040 g F .text 00000014 \_\_swbuf

0001704c g F .text 00000008 \_\_sclose

0001a084 g F .text 00000010 fclose

00010508 g F .text 00000698 \_malloc\_r

0001c118 g F .text 00000030 \_\_ascii\_wctomb

00013a08 g F .text 000000a0 \_fwalk

0001a83c g F .text 00000020 \_mbtowc\_r

000212e0 g F .text 00000084 .hidden \_\_divsi3

00013680 g F .text 00000114 \_malloc\_trim\_r

0001af88 g F .text 0000017c strcmp

00019e44 g F .text 00000018 vfiprintf

0001e160 g F .text 00001374 .hidden \_\_multf3

000100b4 g F .text 00000000 \_fini

00016ea4 g F .text 00000074 sprintf

00021fc4 g O .rodata 00000100 .hidden \_\_clz\_tab

00023b58 g O .sbss 00000004 \_PathLocale

000102f8 g F .text 00000014 atexit

00019f18 g F .text 0000004c \_write\_r

0001a7dc g F .text 00000014 setlocale

00023b40 g O .sdata 00000004 \_impure\_ptr

00013128 g F .text 00000208 \_\_sflush\_r

0001ded0 g F .text 00000148 .hidden \_\_gttf2

0001b2c0 g F .text 00000c50 \_svfiprintf\_r

0001a85c g F .text 0000005c \_\_ascii\_mbtowc

0001f4d4 g F .text 000015a0 .hidden \_\_subtf3

000167a4 g F .text 0000005c \_\_ulp

0001366c g F .text 00000014 \_\_fp\_unlock\_all

00015b38 g F .text 00000020 localeconv

00015b58 g F .text 0000009c \_\_swhatbuf\_r

0001c3e8 g F .text 0000004c \_write

00023b4c g .sdata 00000000 \_edata

00023ba8 g .bss 00000000 \_end

0001a094 g F .text 00000114 \_\_fputwc

00016f70 g F .text 00000084 \_\_swrite

00023b48 g O .sdata 00000004 \_\_malloc\_trim\_threshold

000103d0 g F .text 00000034 exit

00019198 g F .text 00000cac \_vfiprintf\_r

00013aa8 g F .text 000000b4 \_fwalk\_reent

00016634 g F .text 00000170 \_\_mdiff

00021364 g F .text 00000030 .hidden \_\_modsi3

0001364c g F .text 00000004 \_\_sfp\_lock\_release

00014eb0 g F .text 00000bfc \_ldtoa\_r

00021e0c g O .rodata 00000101 \_ctype\_

0001c2d0 g F .text 0000004c \_read

0001c19c g F .text 00000044 \_exit

00015bf4 g F .text 000000f0 \_\_smakebuf\_r

00017070 g F .text 0000001c strlen

0001a7b4 g F .text 00000008 \_\_locale\_ctype\_ptr\_l

0001909c g F .text 000000fc \_\_sprint\_r

0001c0f8 g F .text 00000020 \_wctomb\_r

00010d58 g F .text 00002034 \_vfprintf\_r

00016080 g F .text 000000b0 \_\_lo0bits

0001c0e0 g F .text 00000018 wcrtomb

00016d8c g F .text 000000b4 frexp

000239b0 g O .data 0000016c \_\_global\_locale

00012d8c g F .text 00000018 vfprintf

00020f6c g F .text 00000350 .hidden \_\_trunctfdf2

0001a1d4 g F .text 00000058 fputwc

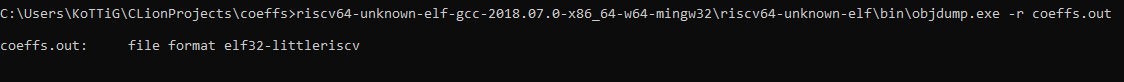
0001c148 g F .text 00000054 \_close

00013650 g F .text 00000004 \_\_sinit\_lock\_acquire

00015dd4 g F .text 00000134 \_\_multadd

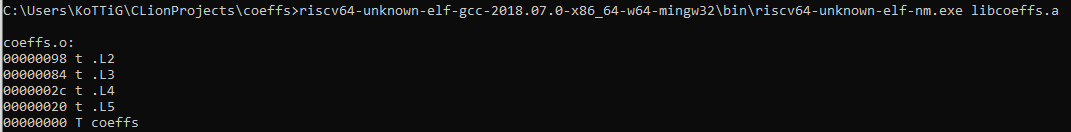
00015db0 g F .text 00000024 \_Bfree

Как мы можем заметить, таблица символов содержит множество дополнительных вхождений, однако определяет все нужные секции, метки и адреса. Функции coeffs (00010240 g F .text 000000b8 coeffs) и main (00010198 g F .text 000000a8 main) помечены флагом F, но в отличие от стадии ассемблирования все они являются определенными и содержатся по корректным адресам, чтобы их можно было успешно вызвать из других участков программы  
  
Откроем таблицу перемещений исполняемого файла:

  
  
Как мы можем заметить, она пуста. Все необходимые релокации, оптимизации и замены инструкций были успешно проведены компоновщиком.

Итогом сборки по шагам является исполняемый на процессорах архитектуры RISC-V файл, выполняющий биноминальных коэффициентов по заданному показателю

**Формирование статической библиотеки, разработка make-файлов для сборки библиотеки.**Статическая библиотека - архив объектных файлов, среди которых компоновщик выбирает «полезные» для данной программы: объектный файл можно считать «полезным», если в нем определяется еще не разрешенный компоновщиком символ. Разработанная нами функция содержится в единственном исходном файле на языке C. Выделим этот файл в статическую библиотеку:  
  
**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\riscv64-unknown-elf-ar.exe -rsc libcoeffs.a coeffs.o**

В выводе утилиты nm кодом T обозначаются символы, определенные в соответствующем объектном файле.  
  


Как мы видим, символ функции сoeffs является основным символом, определяемым в этом объектном файле, остальные символы определяют лишь локальные для этого файла метки.  
  
Используем статическую библиотеку для сборки программ:  
  
  
**C:\Users\KoTTiG\CLionProjects\coeffs>riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\riscv64-unknown-elf-gcc-8.1.0.exe -march=rv32i -mabi=ilp32 -O1 --save-temps main.o libcoeffs.a -o coeffs**  
Получим файл coeffs  
  
Выполним сборку через make-файл:  
  
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).

Листинг написанного make-файла приведен ниже:  
  
TARGET=coeffsProgramm

CC=C:\Users\KoTTiG\CLionProjects\coeffs\riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\riscv64-unknown-elf-gcc-8.1.0

AR=C:\Users\KoTTiG\CLionProjects\coeffs\riscv64-unknown-elf-gcc-2018.07.0-x86\_64-w64-mingw32\bin\riscv64-unknown-elf-ar

CFLAGS=-march=rv32i -mabi=ilp32 -O1

coeffsProgramm: main.c libcoeffs.a

$(CC) $(CFLAGS) main.c libcoeffs.a -o $(TARGET)

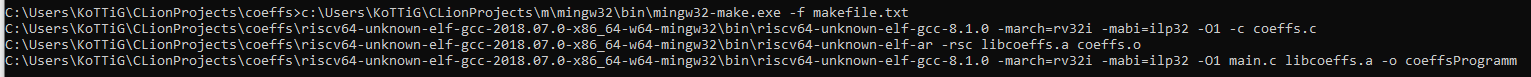
libcoeffs.a: coeffs.o coeffs.h

$(AR) -rsc libcoeffs.a coeffs.o

coeffs.o: coeffs.c

$(CC) $(CFLAGS) -c coeffs.c

clean:

rm -f \*.o \*.a  
  
Запуск make-файла и процесс сборки:  


Что происходит в Make-файле:

1. Создается объектный файл main.o из исходного main.c

2. Создается объектный файл coeffs.o из исходного coeffs.c

3. Архивируется объектный файл coeffs.o (создаём статическую библиотеку libcoeffs.a)

4. Компонуем статическую библиотеку libcoeffs.a с объектным файлом main.o и получаем исполняемый файл coeffsProgram.exe

**Выводы**

В ходе выполнения лабораторной работы были закреплены знания языка C, ассемблера RISC-V, получены навыки работы с препроцессором, компилятором, ассемблером и компоновщиком пакета GCC и драйвером компилятора riscv64-unknown-elf-gcc. Были изучены особенности каждого этапа пошаговой сборки набора программ, а также инструменты, позволяющие выделить разработанные программы в статическую библиотеку и автоматизировать сборку этой библиотеки.