Санкт-Петербургский политехнический университет Петра Великого

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе№ 4**

**Дисциплина**: Низкоуровневое программирование

**Тема**: Раздельная компиляция

Выполнил студент гр. 3530901/10003 Курочкин С.В

(подпись)

Преподаватель Коренев Д. А. (подпись)

“ ”

Санкт-Петербург

Оглавление

1. ТЗ – 3 стр
2. Метод решения – 3 стр
3. Руководство программисту – 3 стр
4. Реализация программы 1 – 3 стр
5. Работа программы 1 – 4 стр
6. Реализация программы 2 – 6 стр
7. Работа программы 2 – 7 стр

1. Техническое задание

Сделать перестановку в массиве чисел. Сначала элементы с нечетными индексами, потом с четными

2. Метод решения

Метод решения заключается в проходе по всем четным числам и перемещении их в конец массива

3. Ход решения

**Текст программы на языке C:**

main.c

Graphical user interface, text

Description automatically generated

fun.h

Text

Description automatically generated

fun.c

A screenshot of a computer

Description automatically generated with medium confidence

Соберем программу по шагам

**Результат работы препроцессора**

**Text

Description automatically generated**

В программе используются директивы #include<stdio.h> и #include<stddef.h>, поэтому в начале файла main.i в начале файла содержится порядка 1300 строк с инструкциями по присоединению stdio.h и stddef.h к проекту, а затем идет наш код на C. В файле fun.i тоже есть строки по присоединению stddef.h, но их значительно меньше, около 30

main.i

Text

Description automatically generated

fun.i

Text

Description automatically generated

Как видно, результат работы препроцессора не сильно различается с исходным кодом программы

**Результат работы компилятора**

**A screenshot of a computer

Description automatically generated with medium confidence**

**main.s**

.file "main.c"

.option nopic

.attribute arch, "rv32i2p0"

.attribute unaligned\_access, 0

.attribute stack\_align, 16

.text

.align 2

.globl main

.type main, @function

main:

addi sp,sp,-64

sw ra,60(sp)

sw s0,56(sp)

sw s1,52(sp)

sw s2,48(sp)

lui a5,%hi(.LANCHOR0)

addi a5,a5,%lo(.LANCHOR0)

lw t4,0(a5)

lw t3,4(a5)

lw t1,8(a5)

lw a7,12(a5)

lw a6,16(a5)

lw a0,20(a5)

lw a1,24(a5)

lw a2,28(a5)

lw a3,32(a5)

lw a4,36(a5)

lw a5,40(a5)

sw t4,4(sp)

sw t3,8(sp)

sw t1,12(sp)

sw a7,16(sp)

sw a6,20(sp)

sw a0,24(sp)

sw a1,28(sp)

sw a2,32(sp)

sw a3,36(sp)

sw a4,40(sp)

sw a5,44(sp)

li a1,11

addi a0,sp,4

call fun

addi s0,sp,4

addi s2,sp,48

lui s1,%hi(.LC1)

.L2:

lw a1,0(s0)

addi a0,s1,%lo(.LC1)

call printf

addi s0,s0,4

bne s0,s2,.L2

li a0,0

lw ra,60(sp)

lw s0,56(sp)

lw s1,52(sp)

lw s2,48(sp)

addi sp,sp,64

jr ra

.size main, .-main

.section .rodata

.align 2

.set .LANCHOR0,. + 0

.LC0:

.word 0

.word 1

.word 2

.word 3

.word 4

.word 5

.word 6

.word 7

.word 8

.word 9

.word 10

.section .rodata.str1.4,"aMS",@progbits,1

.align 2

.LC1:

.string "%d "

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

В данном файле можно видеть выделение памяти в стеке, вызов функции fun, вызов printf

**fun.s**

.file "fun.c"

.option nopic

.attribute arch, "rv32i2p0"

.attribute unaligned\_access, 0

.attribute stack\_align, 16

.text

.align 2

.globl fun

.type fun, @function

fun:

addi a6,a1,1

srli a6,a6,1

beq a6,zero,.L1

mv a7,a0

slli a2,a1,2

addi a0,a0,-4

add a2,a2,a0

li a0,0

addi a1,a1,-1

j .L3

.L6:

addi a0,a0,1

beq a0,a6,.L1

.L3:

slli a5,a0,2

add a5,a5,a7

bleu a1,a0,.L6

.L4:

lw a4,0(a5)

lw a3,4(a5)

sw a3,0(a5)

sw a4,4(a5)

addi a5,a5,4

bne a5,a2,.L4

j .L6

.L1:

ret

.size fun, .-fun

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

В данном же файле можно видеть логику по перестановке элементов массива

**Результат работы ассемблера**

**A screenshot of a computer

Description automatically generated with medium confidence**

Объектный файл не является текстовым, для изучения его содержимого используем утилиту objdump:

**Text

Description automatically generated**

Text

Description automatically generated

Видно, что оба файла имеют таблицу перемещений и таблицу символов

Выведем все заголовки секций объектных файлов

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

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

*.rodata.str1.4 –* наша строка “%d ”

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

.riscv.attributes – информация про RISC-V

Теперь изучим таблицы символов

Text

Description automatically generated

В таблицах находится по одному глобальному символу (флаг g) и по одному символу типа функция (F)

В таблице символов файла main.o есть два неопределенных символа (\*UND\*)

**Содержимое секций**

Изучим содержимое секции .text

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated with low confidence

Дизассемблированный код практически идентичен сгенерированному (за исключением псевдоинструкций).

Секция .data не содержит данных

Text

Description automatically generated

Секция .bss также не содержит данных

Text

Description automatically generated

Секция .rodata

Text

Description automatically generated

Секция .comment содержит сведения о компиляторе, который был использован - GCC версии 8.3.0 от SiFive

Text

Description automatically generated

Секция .riscv.attributes содержит информацию об используемой архитектуре команд - RV32I

Text

Description automatically generated

Посмотрим содержимое таблицы перемещений:

Text

Description automatically generated

Таблица перемещений содержит информацию о переходах (R\_RISCV\_JAL), ветвлениях (R\_RISCV\_BRANCH), вызовах (R\_RISCV\_CALL) и оптимизации (R\_RISCV\_RELAX)

**Результат компоновщика**

****

Результатом работы компоновщика является исполняемый файл a.out

Он тоже является бинарным, будем использовать все тот же objdump

Text

Description automatically generated

Видно, что файл действительно является исполняемым (флаг EXEC\_P).

Также можно увидеть информацию о начальном адресе программы 0x00010090

Изучим секции файла:

Graphical user interface, text

Description automatically generated

Видно, что секций стало больше, по сравнению с объектным файлом, а также увеличились их размеры

Изучим содержимое секции .text

Строк кода стало намного больше (от 80 до 100 тыс), чем в объектном файле, так что здесь я оставлю только самые важные

Disassembly of section .text:

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: 2f0000ef jal ra,10398 <memset>

100ac: 00000517 auipc a0,0x0

100b0: 1f850513 addi a0,a0,504 # 102a4 <\_\_libc\_fini\_array>

100b4: 1a8000ef jal ra,1025c <atexit>

100b8: 24c000ef jal ra,10304 <\_\_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: 1a40006f jal zero,10270 <exit>

00010144 <main>:

10144: fc010113 addi sp,sp,-64

10148: 02112e23 sw ra,60(sp)

1014c: 02812c23 sw s0,56(sp)

10150: 02912a23 sw s1,52(sp)

10154: 03212823 sw s2,48(sp)

10158: 000257b7 lui a5,0x25

1015c: 97878793 addi a5,a5,-1672 # 24978 <\_\_clzsi2+0x4c>

10160: 0007ae83 lw t4,0(a5)

10164: 0047ae03 lw t3,4(a5)

10168: 0087a303 lw t1,8(a5)

1016c: 00c7a883 lw a7,12(a5)

10170: 0107a803 lw a6,16(a5)

10174: 0147a503 lw a0,20(a5)

10178: 0187a583 lw a1,24(a5)

1017c: 01c7a603 lw a2,28(a5)

10180: 0207a683 lw a3,32(a5)

10184: 0247a703 lw a4,36(a5)

10188: 0287a783 lw a5,40(a5)

1018c: 01d12223 sw t4,4(sp)

10190: 01c12423 sw t3,8(sp)

10194: 00612623 sw t1,12(sp)

10198: 01112823 sw a7,16(sp)

1019c: 01012a23 sw a6,20(sp)

101a0: 00a12c23 sw a0,24(sp)

101a4: 00b12e23 sw a1,28(sp)

101a8: 02c12023 sw a2,32(sp)

101ac: 02d12223 sw a3,36(sp)

101b0: 02e12423 sw a4,40(sp)

101b4: 02f12623 sw a5,44(sp)

101b8: 00b00593 addi a1,zero,11

101bc: 00410513 addi a0,sp,4

101c0: 040000ef jal ra,10200 <fun>

101c4: 00410413 addi s0,sp,4

101c8: 03010913 addi s2,sp,48

101cc: 000254b7 lui s1,0x25

101d0: 00042583 lw a1,0(s0)

101d4: 9a448513 addi a0,s1,-1628 # 249a4 <\_\_clzsi2+0x78>

101d8: 2e0000ef jal ra,104b8 <printf>

101dc: 00440413 addi s0,s0,4

101e0: ff2418e3 bne s0,s2,101d0 <main+0x8c>

101e4: 00000513 addi a0,zero,0

101e8: 03c12083 lw ra,60(sp)

101ec: 03812403 lw s0,56(sp)

101f0: 03412483 lw s1,52(sp)

101f4: 03012903 lw s2,48(sp)

101f8: 04010113 addi sp,sp,64

101fc: 00008067 jalr zero,0(ra)

00010200 <fun>:

10200: 00158813 addi a6,a1,1

10204: 00185813 srli a6,a6,0x1

10208: 04080863 beq a6,zero,10258 <fun+0x58>

1020c: 00050893 addi a7,a0,0

10210: 00259613 slli a2,a1,0x2

10214: ffc50513 addi a0,a0,-4

10218: 00a60633 add a2,a2,a0

1021c: 00000513 addi a0,zero,0

10220: fff58593 addi a1,a1,-1

10224: 00c0006f jal zero,10230 <fun+0x30>

10228: 00150513 addi a0,a0,1

1022c: 03050663 beq a0,a6,10258 <fun+0x58>

10230: 00251793 slli a5,a0,0x2

10234: 011787b3 add a5,a5,a7

10238: feb578e3 bgeu a0,a1,10228 <fun+0x28>

1023c: 0007a703 lw a4,0(a5)

10240: 0047a683 lw a3,4(a5)

10244: 00d7a023 sw a3,0(a5)

10248: 00e7a223 sw a4,4(a5)

1024c: 00478793 addi a5,a5,4

10250: fec796e3 bne a5,a2,1023c <fun+0x3c>

10254: fd5ff06f jal zero,10228 <fun+0x28>

10258: 00008067 jalr zero,0(ra)

00010270 <exit>:

10270: ff010113 addi sp,sp,-16

10274: 00000593 addi a1,zero,0

10278: 00812423 sw s0,8(sp)

1027c: 00112623 sw ra,12(sp)

10280: 00050413 addi s0,a0,0

10284: 050030ef jal ra,132d4 <\_\_call\_exitprocs>

10288: 1b818793 addi a5,gp,440 # 26a78 <\_global\_impure\_ptr>

1028c: 0007a503 lw a0,0(a5)

10290: 03c52783 lw a5,60(a0)

10294: 00078463 beq a5,zero,1029c <exit+0x2c>

10298: 000780e7 jalr ra,0(a5)

1029c: 00040513 addi a0,s0,0

102a0: 5080f0ef jal ra,1f7a8 <\_exit>

0001f7a8 <\_exit>:

1f7a8: 00000593 addi a1,zero,0

1f7ac: 00000613 addi a2,zero,0

1f7b0: 00000693 addi a3,zero,0

1f7b4: 00000713 addi a4,zero,0

1f7b8: 00000793 addi a5,zero,0

1f7bc: 05d00893 addi a7,zero,93

1f7c0: 00000073 ecall

1f7c4: 00054463 blt a0,zero,1f7cc <\_exit+0x24>

1f7c8: 0000006f jal zero,1f7c8 <\_exit+0x20>

1f7cc: ff010113 addi sp,sp,-16

1f7d0: 00812423 sw s0,8(sp)

1f7d4: 00050413 addi s0,a0,0

1f7d8: 00112623 sw ra,12(sp)

1f7dc: 40800433 sub s0,zero,s0

1f7e0: 2f0000ef jal ra,1fad0 <\_\_errno>

1f7e4: 00852023 sw s0,0(a0)

1f7e8: 0000006f jal zero,1f7e8 <\_exit+0x40>

Видно функции main и fun, а также метки для входа и выхода из программы: start и exit

Секция .comment хранит все те же сведения о GCCверсии 8.3.0 от SiFive:

Text

Description automatically generated

Секция .riscv.attributes по-прежнему содержит информацию об используемой архитектуре команд RV32I:

Text

Description automatically generated

Проанализируем таблицу символов исполняемого файла

a.out: file format elf32-littleriscv

SYMBOL TABLE:

00010074 l d .text 00000000 .text

00024978 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

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 \_\_call\_atexit.c

00010074 l F .text 0000001c register\_fini

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

00026000 l O .eh\_frame 00000000 \_\_EH\_FRAME\_BEGIN\_\_

000100d0 l F .text 00000000 \_\_do\_global\_dtors\_aux

00026aa0 l O .bss 00000001 completed.5434

000260bc l O .fini\_array 00000000 \_\_do\_global\_dtors\_aux\_fini\_array\_entry

00010120 l F .text 00000000 frame\_dummy

00026aa4 l O .bss 00000018 object.5439

000260b8 l O .init\_array 00000000 \_\_frame\_dummy\_init\_array\_entry

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

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

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

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

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

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

000260c0 l O .data 00000428 impure\_data

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

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

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

00013010 l F .text 000000c0 \_\_sbprintf

00024b14 l O .rodata 00000010 blanks.4504

00024b24 l O .rodata 00000010 zeroes.4505

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

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

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

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

000136d4 l F .text 00000008 \_\_fp\_lock

000136e8 l F .text 00000184 \_\_sinit.part.0

0001386c l F .text 00000008 \_\_fp\_unlock

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

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

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

00013fdc l F .text 00000068 eshdn1

00014044 l F .text 00000070 eshup1

000140b4 l F .text 000000e0 m16m

00014194 l F .text 00000024 eisnan.part.0

000141b8 l F .text 0000004c eneg

00014204 l F .text 00000040 eisneg

00014244 l F .text 000000e8 emovi

0001432c l F .text 0000011c ecmp

00014448 l F .text 0000001c eisinf.part.1

00014464 l F .text 000001ac eshift.part.3

00014610 l F .text 0000017c enormlz

0001478c l F .text 00000408 emdnorm

00014b94 l F .text 00000164 eiremain

00014cf8 l F .text 000000b0 emovo.isra.6

00014da8 l F .text 00000408 emul

000151b0 l F .text 00000504 ediv

000156b4 l F .text 00000144 e113toe.isra.8

00024b78 l O .rodata 00000014 ezero

00024b8c l O .rodata 00000014 eone

00024ba0 l O .rodata 00000104 etens

00024ca4 l O .rodata 00000104 emtens

00024da8 l O .rodata 00000022 bmask

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

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

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

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

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

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

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

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

00024e08 l O .rodata 0000000c p05.3298

00000000 l df \*ABS\* 00000000 sbrkr.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

0002509c l O .rodata 00000010 blanks.4489

000250ac l O .rodata 00000010 zeroes.4490

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

0001b928 l F .text 000000f0 \_\_sprint\_r.part.0

0001cd18 l F .text 000000c0 \_\_sbprintf

00025228 l O .rodata 00000010 blanks.4480

00025238 l O .rodata 00000010 zeroes.4481

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

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

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

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

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

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

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

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

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

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

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

000254b8 l O .rodata 00000010 blanks.4466

000254c8 l O .rodata 00000010 zeroes.4467

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

00000000 l df \*ABS\* 00000000 wcrtomb.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

00026a9c l O .sbss 00000004 heap\_end.1862

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

000260b0 l O .eh\_frame 00000000 \_\_FRAME\_END\_\_

00000000 l df \*ABS\* 00000000

000260c0 l .fini\_array 00000000 \_\_fini\_array\_end

000260bc l .fini\_array 00000000 \_\_fini\_array\_start

000260bc l .init\_array 00000000 \_\_init\_array\_end

000260b4 l .init\_array 00000000 \_\_preinit\_array\_end

000260b4 l .init\_array 00000000 \_\_init\_array\_start

000260b4 l .init\_array 00000000 \_\_preinit\_array\_start

00018648 g F .text 0000009c \_mprec\_log10

0001874c g F .text 00000078 \_\_any\_on

0001d7a4 g F .text 00000054 \_isatty\_r

00024ee0 g O .rodata 00000028 \_\_mprec\_tinytens

00018c88 g F .text 000000b0 strcpy

0001e0c4 g F .text 00000040 cleanup\_glue

0001d7f8 g F .text 00000060 \_lseek\_r

000214ac g F .text 00000144 .hidden \_\_getf2

000213e0 g F .text 000000cc .hidden \_\_eqtf2

000104b8 g F .text 00000054 printf

000268c0 g \*ABS\* 00000000 \_\_global\_pointer$

0001f648 g F .text 00000078 \_wcrtomb\_r

00018a9c g F .text 00000068 \_\_sseek

00013a04 g F .text 00000010 \_\_sinit

0001f4b0 g F .text 00000184 \_\_swbuf\_r

00016b68 g F .text 0000007c \_setlocale\_r

00013874 g F .text 00000078 \_\_sfmoreglue

00017710 g F .text 00000004 \_\_malloc\_unlock

00024190 g F .text 00000188 .hidden \_\_floatsitf

0001d974 g F .text 00000120 memmove

000139f0 g F .text 00000014 \_cleanup

00017714 g F .text 000000a8 \_Balloc

0001fad0 g F .text 0000000c \_\_errno

0001fa2c g F .text 000000a4 \_conv\_stat

00016b50 g F .text 00000008 \_\_localeconv\_l

0001d28c g F .text 0000005c \_fstat\_r

00026ae4 g O .bss 00000004 errno

00018a0c g F .text 00000008 \_\_seofread

00026a60 g .sdata 00000000 \_\_SDATA\_BEGIN\_\_

0001d858 g F .text 0000011c memcpy

000136dc g F .text 0000000c \_cleanup\_r

00018e90 g F .text 00002a98 \_svfprintf\_r

000185a4 g F .text 000000a4 \_\_ratio

00010474 g F .text 00000044 \_printf\_r

000215f0 g F .text 00000144 .hidden \_\_letf2

00024880 g F .text 00000048 .hidden \_\_udivsi3

0001f7ec g F .text 00000070 \_fstat

00024e18 g O .rodata 000000c8 \_\_mprec\_tens

00026a98 g O .sbss 00000004 \_\_malloc\_top\_pad

00026a7c g O .sdata 00000000 .hidden \_\_dso\_handle

00016b58 g F .text 00000008 \_localeconv\_r

0001d2e8 g F .text 000004bc \_\_sfvwrite\_r

00017b74 g F .text 00000034 \_\_i2b

000187c4 g F .text 00000054 \_sbrk\_r

0001da94 g F .text 00000060 \_read\_r

0001cf90 g F .text 00000110 \_fclose\_r

000136ac g F .text 00000028 fflush

00026a94 g O .sbss 00000004 \_\_malloc\_max\_sbrked\_mem

00018348 g F .text 00000118 \_\_b2d

000200b0 g F .text 000004f0 .hidden \_\_umoddi3

0001f85c g F .text 00000040 \_isatty

00026a78 g O .sdata 00000004 \_global\_impure\_ptr

0001daf4 g F .text 000005d0 \_realloc\_r

00010304 g F .text 00000094 \_\_libc\_init\_array

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

0001d1dc g F .text 0000002c \_fputwc\_r

00024f08 g O .rodata 00000028 \_\_mprec\_bigtens

00017938 g F .text 00000110 \_\_s2b

0001f93c g F .text 000000a0 \_sbrk

000180a8 g F .text 0000005c \_\_mcmp

00013a24 g F .text 00000014 \_\_fp\_lock\_all

000102a4 g F .text 00000060 \_\_libc\_fini\_array

0001e104 g F .text 00000110 \_reclaim\_reent

00017a48 g F .text 00000074 \_\_hi0bits

00024040 g F .text 00000150 .hidden \_\_fixtfsi

00017e14 g F .text 00000148 \_\_pow5mult

000248c8 g F .text 00000010 .hidden \_\_umodsi3

00010200 g F .text 0000005c fun

0002492c g F .text 0000004c .hidden \_\_clzsi2

00013a14 g F .text 00000004 \_\_sfp\_lock\_acquire

00017630 g F .text 000000dc memchr

000188c8 g F .text 0000006c \_sprintf\_r

00013b74 g F .text 000002f8 \_free\_r

00016be4 g F .text 00000008 \_\_locale\_mb\_cur\_max

000132d4 g F .text 0000011c \_\_call\_exitprocs

00026a84 g O .sdata 00000004 \_\_malloc\_sbrk\_base

00010090 g F .text 00000040 \_start

0001f89c g F .text 00000050 \_lseek

00017f5c g F .text 0000014c \_\_lshift

000215f0 g F .text 00000144 .hidden \_\_lttf2

000213e0 g F .text 000000cc .hidden \_\_netf2

0001e214 g F .text 000001bc \_\_ssprint\_r

00013238 g F .text 0000009c \_\_register\_exitproc

00016aec g F .text 00000064 \_ldcheck

00017ba8 g F .text 0000026c \_\_multiply

00024854 g F .text 00000024 .hidden \_\_mulsi3

00018dc4 g F .text 000000cc strncpy

00026abc g O .bss 00000028 \_\_malloc\_current\_mallinfo

00018460 g F .text 00000144 \_\_d2b

00024318 g F .text 00000208 .hidden \_\_extenddftf2

0001cf3c g F .text 00000054 \_close\_r

000130d0 g F .text 00000168 \_\_swsetup\_r

000205a0 g F .text 000007d0 .hidden \_\_divdf3

000138ec g F .text 00000104 \_\_sfp

000186e4 g F .text 00000068 \_\_copybits

00026ae8 g .bss 00000000 \_\_BSS\_END\_\_

00026654 g O .data 00000408 \_\_malloc\_av\_

00013a20 g F .text 00000004 \_\_sinit\_lock\_release

00020d70 g F .text 00000670 .hidden \_\_muldf3

000189b0 g F .text 0000005c \_\_sread

0001770c g F .text 00000004 \_\_malloc\_lock

0001364c g F .text 00000060 \_fflush\_r

0001ce74 g F .text 000000c8 \_calloc\_r

00026a8c g .sbss 00000000 \_\_bss\_start

00010398 g F .text 000000dc memset

00010144 g F .text 000000bc main

00026a90 g O .sbss 00000004 \_\_malloc\_max\_total\_mem

0001f634 g F .text 00000014 \_\_swbuf

00018b04 g F .text 00000008 \_\_sclose

0001d0a0 g F .text 00000010 fclose

00016df0 g F .text 000007cc \_malloc\_r

0001cde4 g F .text 00000030 \_\_ascii\_wctomb

00013e6c g F .text 000000b0 \_fwalk

000175bc g F .text 0000000c \_mbtowc\_r

00024878 g F .text 00000084 .hidden \_\_divsi3

00013a4c g F .text 00000128 \_malloc\_trim\_r

00018b0c g F .text 0000017c strcmp

0001cd00 g F .text 00000018 vfiprintf

00021734 g F .text 0000136c .hidden \_\_multf3

00018934 g F .text 0000007c sprintf

0002558c g O .rodata 00000100 .hidden \_\_clz\_tab

00026a8c g O .sbss 00000004 \_PathLocale

0001025c g F .text 00000014 atexit

0001ce14 g F .text 00000060 \_write\_r

00016bec g F .text 00000014 setlocale

00026a80 g O .sdata 00000004 \_impure\_ptr

000133f0 g F .text 0000025c \_\_sflush\_r

000214ac g F .text 00000144 .hidden \_\_gttf2

0001e3d0 g F .text 000010e0 \_svfiprintf\_r

000175c8 g F .text 00000068 \_\_ascii\_mbtowc

00022aa0 g F .text 000015a0 .hidden \_\_subtf3

000182e8 g F .text 00000060 \_\_ulp

00013a38 g F .text 00000014 \_\_fp\_unlock\_all

00016b60 g F .text 00000008 localeconv

00016c00 g F .text 000000d0 \_\_swhatbuf\_r

000260c0 g .data 00000000 \_\_DATA\_BEGIN\_\_

0001f9dc g F .text 00000050 \_write

00026a8c g .sdata 00000000 \_edata

00026ae8 g .bss 00000000 \_end

0001d0b0 g F .text 0000012c \_\_fputwc

00018a14 g F .text 00000088 \_\_swrite

00026a88 g O .sdata 00000004 \_\_malloc\_trim\_threshold

00010270 g F .text 00000034 exit

0001ba30 g F .text 000012d0 \_vfiprintf\_r

00013f1c g F .text 000000c0 \_fwalk\_reent

00018104 g F .text 000001e4 \_\_mdiff

000248fc g F .text 00000030 .hidden \_\_modsi3

00013a18 g F .text 00000004 \_\_sfp\_lock\_release

000157f8 g F .text 000012f4 \_ldtoa\_r

00025248 g O .rodata 00000101 \_ctype\_

0001f8ec g F .text 00000050 \_read

0001f7a8 g F .text 00000044 \_exit

00016cd0 g F .text 00000120 \_\_smakebuf\_r

00018d38 g F .text 0000008c strlen

0001ba18 g F .text 00000018 \_\_sprint\_r

0001cdd8 g F .text 0000000c \_wctomb\_r

0001050c g F .text 00002aec \_vfprintf\_r

00017abc g F .text 000000b8 \_\_lo0bits

0001f6c0 g F .text 00000090 wcrtomb

00018818 g F .text 000000b0 frexp

000264e8 g O .data 0000016c \_\_global\_locale

00012ff8 g F .text 00000018 vfprintf

00024520 g F .text 00000334 .hidden \_\_trunctfdf2

0001d208 g F .text 00000084 fputwc

0001f750 g F .text 00000058 \_close

00013a1c g F .text 00000004 \_\_sinit\_lock\_acquire

000177e0 g F .text 00000158 \_\_multadd

000177bc g F .text 00000024 \_Bfree

Таблица символов содержит множество дополнительных вхождений, однако в целом определяет все нужные секции, метки и адреса. Функции fun и main так же помечены флагом F и являются глобальными, но в отличие от стадии ассемблирования, все они являются определенными и содержатся по корректным адресам для успешного вызова этих функций из других участков программ.

A screenshot of a computer

Description automatically generated with low confidence

Таблица перемещений оказывается пуста, все необходимые релокации, оптимизации и замены инструкций были успешно проведены компоновщиком.

Итогом сборки программ на языке C по шагам является исполняемый на процессорах архитектуры RISC-V файл, решающий задачу перемещения элементов массива с четными индексами в конец массива

**Разработка статической библиотеки, разработка make-файлов для сборки 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.

Выделим функцию перестановки элементов массива с четными индексами в конец массива в статическую библиотеку

Text

Description automatically generated

Text

Description automatically generated

Поиск библиотеки и сборка программы



На выходе получили файл main

Создадим make-файл

a.out: main.o libfun.a

riscv64-unknown-elf-gcc -march=rv32i -mabi=ilp32 -O1 main.c libfun.a -o a.out

libfun.a: fun.o

riscv64-unknown-elf-ar -rsc libfun.a main.o

main.o: main.c

riscv64-unknown-elf-gcc -march=rv32i -mabi=ilp32 -O1 -c main.c –o main.o

Text

Description automatically generated

**Вывод**

В ходе выполнения лабораторной работы были получены навыки программирования на языке C, получены навыки пошаговой сборки программ и навыки работы с препроцессором, компилятором, ассемблером и компоновщиком. Был изучен каждый этап пошаговой сборки программ. Также были изучены инструменты, позволяющие выделить разработанные программы в статическую библиотеку. Были получены навыки работы с make-файлами. Была реализована поставленная задача – «перестановка элементов массива с четными индексами в конец массива».