Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчёт по лабораторной работе
Дисциплина: Низкоуровневое программирование
Тема: Программирование RISC-V

Выполнил студент гр. 3530901/00002 Сергеева Е.О._____(подпись)
Преподаватель Степанов Д. С. _____(подпись)

“18” ноября 2021

г. Санкт-Петербург
2021

Вариант 10: реализовать слияние двух отсортированных массивов.

Решение задачи:

1) В цикле сравниваем элементы из массивов A и B, меньший элемент добавляем в выходной массив.

2) Если мы добавили элемент из массива A, то счетчик элемента массива A уменьшается на 1, и указатель на элемент массива A переходит к след ячейке.

3) Если мы добавили элемент из массива B, то счетчик элемента массива B уменьшается на 1, и указатель на элемент массива Bпереходит к след ячейке.

4) При добавлении элемента в выходной массив, указатель сдвигается на след ячейку.

5) Выход из цикла происходит, когда сумма счетчиков элементов массивов равно 0.

Реализация программы

```
.text
.globl __start
__start:
la a0, arrayC #выходной массив
lw a1, sizeA #размер массива A(счетчик элементов A)
lw a2, sizeB #размер массива B (счетчик элементов B)
la a3, arrayA #A[0]
la a4, arrayB #B[0]

loop:
  add a5, a1, a2 #оставшееся количество незаполненных элементов в выходном массиве
  beq a5, zero, endloop #если это значение = 0, то мы заполнили выходной массив, необходимо выйти из цикла

  lw t0, 0(a3) #загружаем элемент из массива A
  lw t1, 0(a4) #загружаем элемент из массива B

  beq a1, zero, writefromB #если счетчик элементов массива A дошел до 0, записываем элемент из массива B
  beq a2, zero, writefromA #если счетчик элементов массива B дошел до 0, записываем элемент из массива A

  bgtu t0, t1, writefromB #если элемент из первого массива >=, чем из второго, то добавляем элемент из B
```

```
  writefromA:
  sw t0, 0(a0) #сохраняем в память в arrayC C[k] = A[i]
  addi a1, a1, -1 #уменьшаем счетчик элементов массива A
  addi a3, a3, 4 #переходим к след элементу массива A a3 = A[i+1]
  addi a0, a0, 4 #переходим к след ячейке, куда будем сохранять C[k+1]
  jal zero, loop #переходим в начало цикла

  writefromB:
  sw t1, 0(a0) #сохраняем в память в arrayC C[k] = B[j]
  addi a2, a2, -1 #уменьшаем счетчик элементов массива B
  addi a4, a4, 4 #переходим к след элементу массива B a4 = B[j+1]
  addi a0, a0, 4 #переходим к след ячейке, куда будем сохранять C[k+1]
  jal zero, loop #переходим в начало цикла
endloop:

finish:
  li a0, 10 #останов
  ecall

.rodata
sizeA:
  .word 5
arrayA:
  .word 1, 2, 3, 4, 9
sizeB:
  .word 5
arrayB:
  .word 5, 6, 7, 8, 10

.data
arrayC:
  .word 10
```

```
0x00010090      00   00   00   09
0x0001008c      00   00   00   04
0x00010088      00   00   00   03
0x00010084      00   00   00   02
0x00010080      00   00   00   01
```

Рис.1. Массив A[]

```
 0x000100a8      00   00   00   0a
 0x000100a4      00   00   00   08
 0x000100a0      00   00   00   07
 0x0001009c      00   00   00   06
 0x00010098      00   00   00   05
```

Рис. 2. Массив B[]

```
0x000100d0      00   00   00   0a
0x000100cc      00   00   00   09
0x000100c8      00   00   00   08
0x000100c4      00   00   00   07
0x000100c0      00   00   00   06
0x000100bc      00   00   00   05
0x000100b8      00   00   00   04
0x000100b4      00   00   00   03
0x000100b0      00   00   00   02
0x000100ac      00   00   00   01
```

Рис. 3. Массив C[]

Реализация подпрограммы :

join.s

```
.text
__join:
.globl __join

loop:
add a5, a1, a2 #оставшееся количество незаполненных элементов в
выходном массиве
beq a5, zero, endloop #если это значение = 0, то мы заполнили выходной
массив, необходимо выйти из цикла

lw t0, 0(a3) #загружаем элемент из массива A
lw t1, 0(a4) #загружаем элемент из массива B

beq a1, zero, writefromB #если счетчик элементов массива A дошел до 0,
записываем элемент из массива B
beq a2, zero, writefromA #если счетчик элементов массива B дошел до 0,
записываем элемент из массива A

bgtu t0, t1, writefromB #если элемент из первого массива >=, чем из
второго, то добавляем элемент из B

writefromA:
sw t0, 0(a0) #сохраняем в память в arrayC C[k] = A[i]
addi a1, a1, -1 #уменьшаем счетчик элементов массива A
addi a3, a3, 4 #переходим к след элементу массива A a3 = A[i+1]
addi a0, a0, 4 #переходим к след ячейке, куда будем сохранять C[k+1]
jal zero, loop #переходим в начало цикла

writefromB:
sw t1, 0(a0) #сохраняем в память в arrayC C[k] = B[j]
addi a2, a2, -1 #уменьшаем счетчик элементов массива B
addi a4, a4, 4 #переходим к след элементу массива B a4 = B[j+1]
addi a0, a0, 4 #переходим к след ячейке, куда будем сохранять C[k+1]
jal zero, loop #переходим в начало цикла
endloop:
ret #выход из подрограммы
```

mainprogamm.s

```
.text
 __start:
.globl __start
la a0, arrayC #выходной массив
lw a1, sizeA #размер массива A(счетчик элементов A)
lw a2, sizeB #размер массива B (счетчик элементов B)
la a3, arrayA #A[0]
la a4, arrayB #B[0]

addi sp, sp, -16 #выделяем место на стеке (так как надо выровнить по границе)
sw ra, 12(sp) #сохраняем регистр ra(адрес возврата) на стеке

call __join

lw ra, 12(sp) #загружаем обратно ra
addi sp, sp, 16 #освобождаем место на стеке

finish:
li a0, 10 #останов
  ecall

.rodata
sizeA:
  .word 5
arrayA:
  .word 1, 2, 3, 4, 9
sizeB:
  .word 5
arrayB:
  .word 5, 6, 7, 8, 10

.data
arrayC:
  .word 10
```

```
0x000100ac      00    00    00    09

0x000100a8      00    00    00    04

0x000100a4      00    00    00    03

0x000100a0      00    00    00    02

0x0001009c      00    00    00    01
```

Рис.4. Массив A []

```
0x000100c4      00    00    00    0a

0x000100c0      00    00    00    08

0x000100bc      00    00    00    07

0x000100b8      00    00    00    06

0x000100b4      00    00    00    05
```

Рис.5. Массив B []

```
0x000100ec      00    00    00    0a

0x000100e8      00    00    00    09

0x000100e4      00    00    00    08

0x000100e0      00    00    00    07

0x000100dc      00    00    00    06

0x000100d8      00    00    00    05

0x000100d4      00    00    00    04

0x000100d0      00    00    00    03

0x000100cc      00    00    00    02

0x000100c8      00    00    00    01
```

Рис.6. Массив C []

Вывод:

В ходе выполнения лабораторной работы была реализована программа на RISCV, реализующая слияние двух отсортированных массивов