

ГУАП

КАФЕДРА № 44

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель

должность, уч. степень, звание

подпись, дата

А.Н.Долидзе

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

Программная реализация алгоритма выполнения целочисленной
операции для архитектуры набора команд VAX

по курсу: Организация ЭВМ и систем

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

4142

подпись, дата

К.С. Некрасов

инициалы, фамилия

Санкт-Петербург 2024

Задание

Деление целых чисел без знака для получения целого числа без восстановления остатка с неподвижным делителем и сдвигом делимого

Задание по варианту

Номер алгоритма $14 \bmod 16 + 1 = 15$

Разрядность: $14 \bmod 7 + 4 = 4$

Адрес начала расположения исходных данных: $\$ 14 + 20 = 34_{10} = 22_{16} \$$

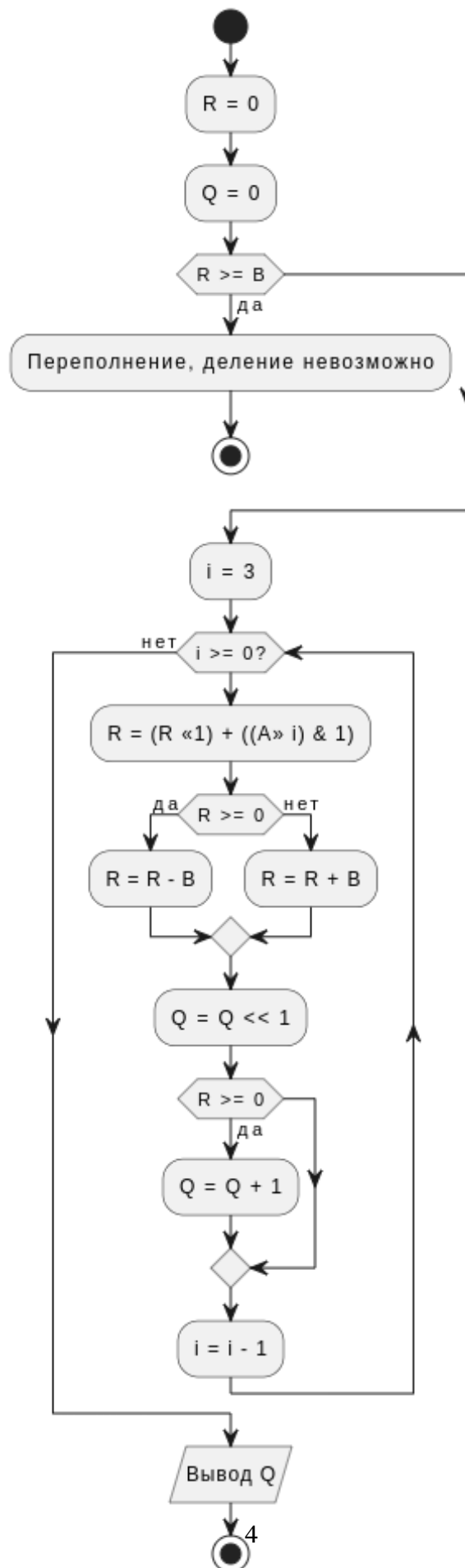
Адрес начала расположения команд программы: $\$ 14 + 120 = 134_{10} = 86_{16} \$$

Типы адресации: $1 + 14 \bmod 18 = 15$

Тип1 (Перенос данных из памяти в регистры): 6 - косвенная регистровая (простая косвенная)

Тип2 (Сохранение результата в память): 9 - косвенная автоинкрементная (двойная косвенная с автоувеличением)

Блок-схема алгоритма



Код программы на языке ассемблера ARM

Таблица 1 – Используемые регистры

Мнемокод	HEX код	Комментарий
movb 0, r0	90 00 50	# Остаток R = 0
movb 0, r1	90 00 51	# Результат Q = 0
movl #22, r6	d0 8f 22 56	# r6 указывает на A
movl #23, r7	d0 8f 23 57	# r7 указывает на B
movb (r6), r2	90 66 52	# Загружаем A в r2
movb (r7), r3	90 67 53	# Загружаем B в r3
cmpl r0, r3	d1 50 53	# Сравниваем R с B
bgtru overflow	1A <i>overflow</i>	# если R >= B, переходим в overflow
movb 3, r4	90 03 54	# i = 3
ashl 01, r0, r0	78 01 50 50	# R = (R « 1)
movb r2, r5	90 52 55	# Кладём A в r5
mcomb r4, r8	90 54 58	# кладём в r8 инвертированную r4, чтобы сдвигать влево
ashl r8, r5, r5	78 55 55	# A » i
bitb 01, r5	93 01 55	# (A » i) & 1
beql 00 afteradd	13 00 <i>afteradd</i>	# если i-ный бит A оказался 0 - пропускаем увеличение R
addb2 01, r0	80 01 50	# инкрементируем R
cmpb r0, 00	91 50 00	# Сравниваем R с 0
blssu plus_b	1f <i>plus_b</i>	# Если R < 0, прыгаем в plus_b
subb2 r3, r0	82 53 50	# Уменьшаем R на B
brb cont	11	# Перепрыгиваем увеличение
	<i>CONT_ADDRESS</i>	
addb2 r3, r0	80 53 50	# Увеличиваем R на B
ashl 01, r1, r1	78 01 51 51	# Q = Q « 1 (Сдвигаем Q влево на 1)
cmpb r0, 00	91 50 00	# Снова сравниваем R с 0
blssu	1f	# Если меньше, то перепрыгиваем
dec_i_and_check	<i>dec_i_and_check</i>	инкрементацию Q
addb2 01, r1	80 01 51	# Иначе увеличиваем Q
subb2 01, r4	82 01 54	# i -= 1
cmpb r4, 00	91 54 00	# Проверяем что i >= 0
bge i_loop	1e <i>i_loop</i>	# если да, прыгаем назад, иначе идём дальше
movl #24, r6	d0 8f 24 66	# r6 указывает на result
movb r1, (r6)+	90 51 86	# Сохраняем Q в память и инкрементируем r6

Мнемокод	HEX код	Комментарий
movb 00, r8	90 00 58	# Успешное завершение
HALT	00	
movb 01, r8	90 01 58	# Переполнение - деление невозможно
halt	00	# Останавливаем выполнение

Адрес	Что лежит
R0	Остаток R
24,R1	Результат Q
22,R2	Делимое (A)
23,R3	Делитель (B)
R4	i счетчик
86	Начало программы

Результат работы программы

The screenshot displays a software interface with a main table, a register window, and a comment window.

Main Table:

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
00000000	11	86	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000020	00	00	05	05	05	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040	00	00	00	00	00	12	00	00	00	00	00	00	00	00	00	00
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080	00	00	00	00	00	00	90	00	50	90	00	51	D0	8F	22	56
00000090	D0	8F	23	57	90	66	52	90	67	53	D1	50	53	1A	E4	90
000000A0	03	54	78	01	50	50	90	52	55	90	54	58	78	55	55	93
000000B0	01	55	13	00	B8	80	01	50	91	50	00	1F	C2	82	53	50
000000C0	11	C5	80	53	50	78	01	51	51	91	50	00	1F	D1	80	01
000000D0	51	82	01	54	91	54	00	1E	A2	D0	8F	24	66	90	51	86
000000E0	90	00	58	00	80	01	58	00	00	00	00	00	00	00	00	00
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000210	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Register Window:

Register	Value
R0	00000001
R1	00000002
R2	00000012
R3	00000005
R4	00000000
R5	00000000
R6	00000000
R7	00000000
R8	00000018
R9	00000000
RA	00000000
RB	00000000
RD	00000000
RE	00000000
RF	000000E4

Comment Window:

Логическое сложение op1 и op2, результат запоминается по адресу op2

Содержимое op1 логически умножается на инвертированное содержимое op1, результат запоминается по адресу op2

Окончание табл. 1.7

КОП	Операция	Флаги	Описание
		N Z V C	

Bottom Panel:

```

Имя файла: 000000E3
Файл: 00
Имя файла: HALT
Код: 0
  
```

Рисунок 2 – Результат