Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

НАПРАВЛЕНИЕ ПОДГОТОВКИ «09.03.04 Программная инженерия»

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Название: Изучение принципов работы микропроцессорного ядра RISC-V

Дисциплина: Архитектура ЭВМ

Студент	ИУ7-54Б		С. Д. Параскун
	Группа	Подпись, дата	И. О. Фамилия
Преподаватель			А. Ю. Попов
		Подпись, дата	И. О. Фамилия

Москва, 2021 г.

1. Задание №1

В данной работе будет выполнен 12 вариант.

Листинг 1.1 – Код рассматриваемой программы

```
  .section .text
        .globl _start;
        len = 8 #array length
        enroll = 4 #count of elems in one loop
        elem_sz = 4 #size of one element

_start:
        la x1, _x
        addi x20, x1, elem_sz*len #address of last elem
lp:
        lw x2, 0(x1)
        lw x3, 4(x1)
        add x31, x31, x2 #!
        add x31, x31, x3
        lw x4, 8(x1)
        lw x5, 12(x1)
        add x31, x31, x4
        add x31, x31, x5
        addi x1, x1, elem_sz*enroll
        bne x1, x20, lp
        addi x31, x31, 1
lp2: j lp2

        .section .data
_x:     .4byte 0x1
        .4byte 0x2
        .4byte 0x3
        .4byte 0x4
        .4byte 0x5
        .4byte 0x6
        .4byte 0x7
        .4byte 0x8
```

В регистре $x31$ в конце программы будет находится сумма элементов массива + 1, т.е. значение его будет 37. За один цикл обрабатывается 4 элемента, при этом сначала считываются 2, потом прибавляются к результату, потом считываются еще 2 и прибавляются.

Листинг 1.2 – Псевдокод рассматриваемой программы

```
#define len 8
#define enroll 4
#define elem_sz 4
int _x[ ]={1,2,3,4,5,6,7,8};
void _start() {
    int *x1 = _x;
    int x20 = x1 + len;

    do {
      int x2 = x1[0];
      int x3 = x1[1];
      x31 += x2;
      x31 += x3;
      int x4 = x1[2];
      int x5 = x1[3];
      x31 += x4;
      x31 += x5;
      x1 += enroll;
    } while(x1 != x20);
    x31++;
    while(1){}
}
```

Листинг 1.3 – Дизассемблированный листинг рассматриваемой программы

```
80000000 <_start>:
80000000:       00000097                auipc   x1,0x0
80000004:       03c08093                addi    x1,x1,60 # 8000003c <_x>
80000008:       02008a13                addi    x20,x1,32

8000000c <lp>:
8000000c:       0000a103                lw      x2,0(x1)
80000010:       0040a183                lw      x3,4(x1)
80000014:       002f8fb3                add     x31,x31,x2
80000018:       003f8fb3                add     x31,x31,x3
8000001c:       0080a203                lw      x4,8(x1)
80000020:       00c0a283                lw      x5,12(x1)
80000024:       004f8fb3                add     x31,x31,x4
80000028:       005f8fb3                add     x31,x31,x5
8000002c:       01008093                addi    x1,x1,16
80000030:       fd409ee3                bne     x1,x20,8000000c <lp>
80000034:       001f8f93                addi    x31,x31,1

80000038 <lp2>:
80000038:       0000006f                jal     x0,80000038 <lp2>
```

2. Задание №2

На рисунке ниже рассмотренна временная диаграмма стадий выборки и диспетчеризации команды 8000000c на 2-й итерации цикла. Выборка происходит в 27 такте, так так сигнал fetch_complete сброшен в 0, диспетчеризация заканчивается в 29 (заканчивается записью 0000a103 в instruction_table).

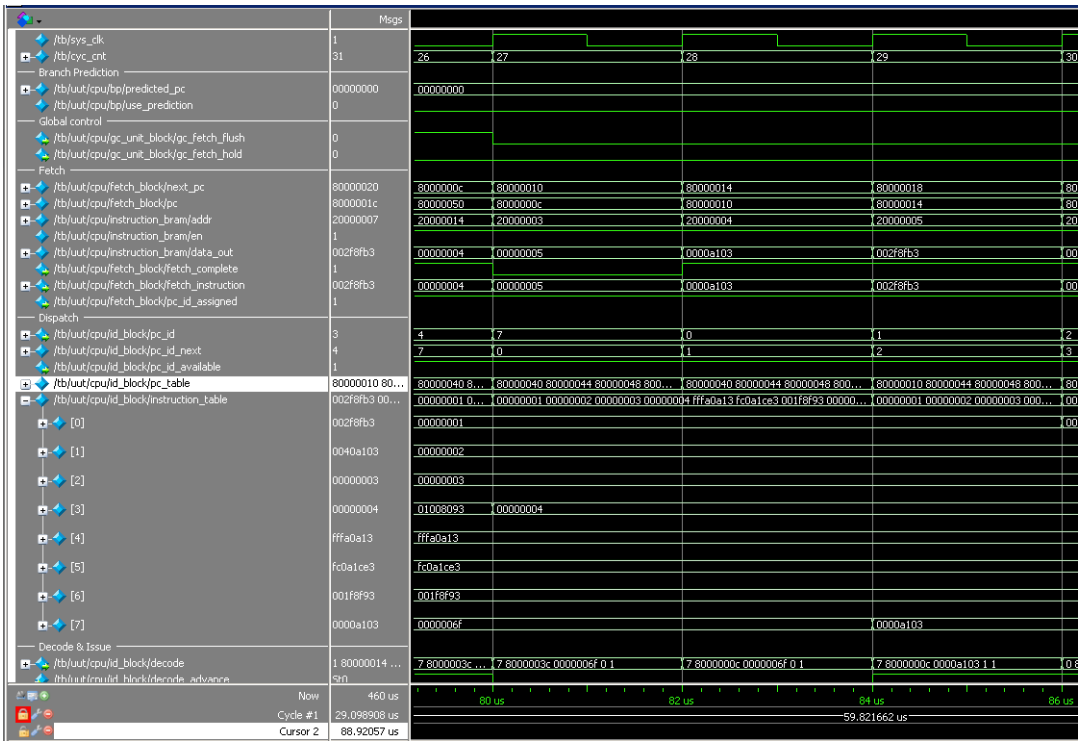


Рисунок 2.1 – Временная диаграмма выполнения стадий выборки и диспетчеризации команды 8000000c на 2-й итерации

3. Задание №3

На рисунке ниже рассмотренна временная диаграмма стадии декодирования и планирования на выполнение команды 80000018 на 2-й итерации цикла. В 34 такте происходит декодирование команды, в 35 она планируется на выполнение, возникает конфликт по регистру rs2, в 36 такте завершается выполнение предыдущей команды.

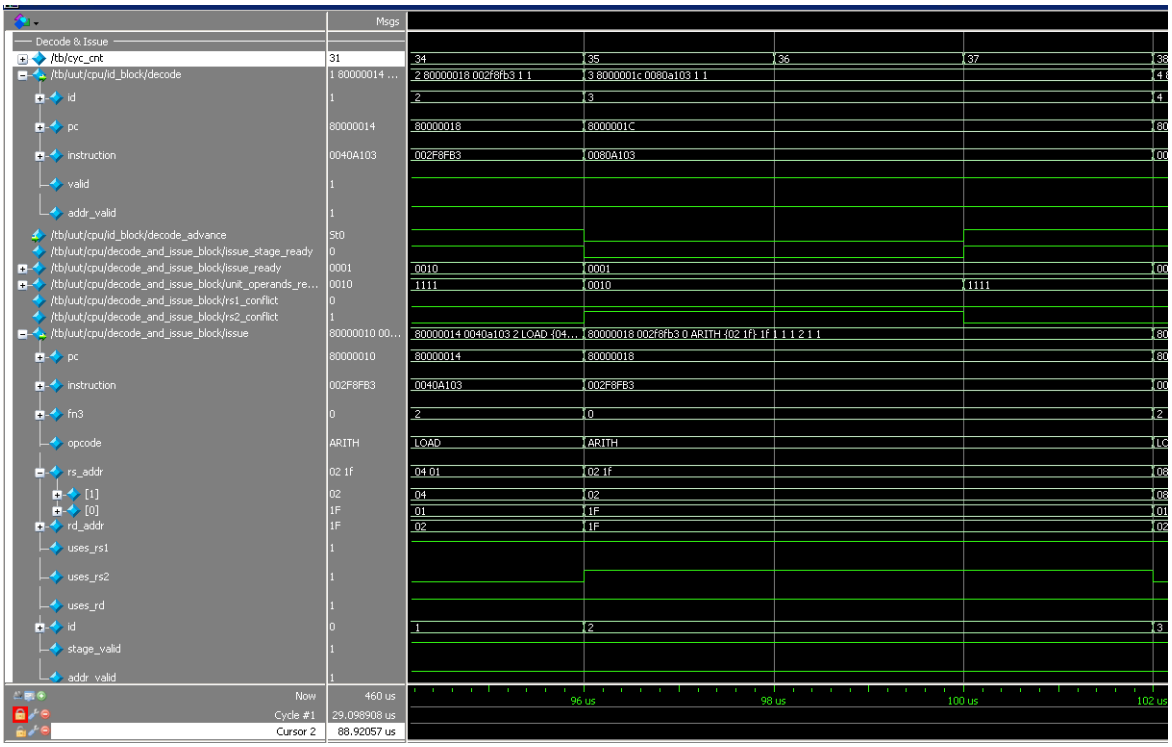


Рисунок 3.1 – Временная диаграмма выполнения стадий декодирования и планирования на выполнение команды 80000018 на 2-й итерации

4. Задание №4

На рисунке ниже рассмотренна временная диаграмма стадии выполнения команды 8000002c на 1-й итерации цикла. Видно, что в 23 такте данная команда поступает на выполнение, так как это арифметическая операция, она выполняется блоком ALU, в unit_wb[0]/id записывается 3 (как и у исследуемой команды), сигнал unit_wb[0]/done выставлен в 1. Следовательно выполнение данной команды завершается в этом же такте.

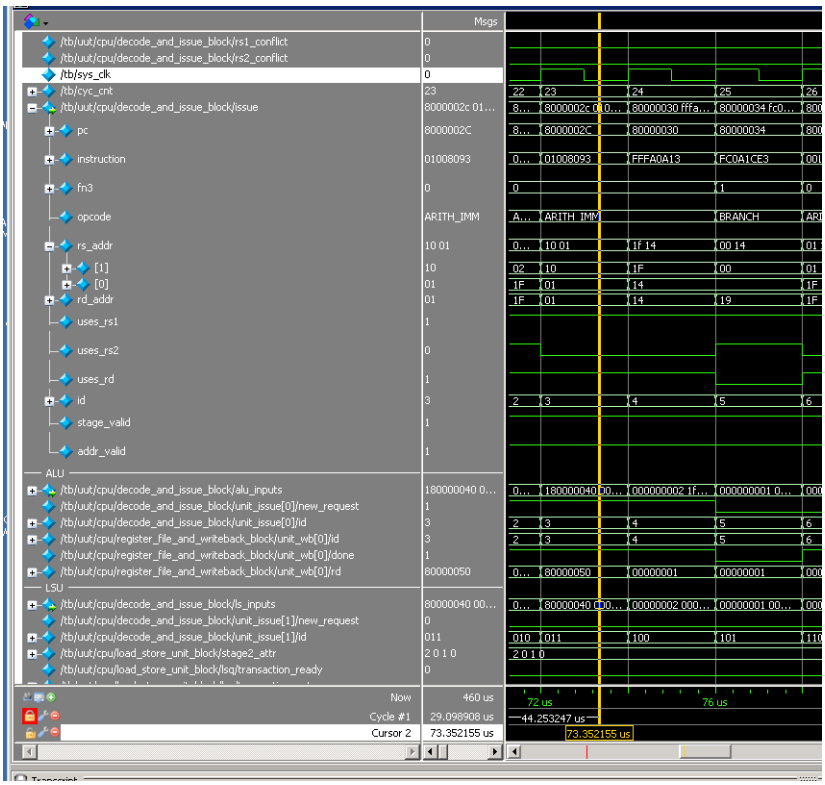


Рисунок 4.1 – Временная диаграмма выполнения стадии выполнения команды 8000002c на 1-й итерации

5. Задание №5

Для начала узнаем значение регистра $x31$, из рисунка ниже видно, что вычисленное в первом задании значение совпадает с хранящимся там

$$00000025(hex) = 37(dec) \tag{5.1}$$

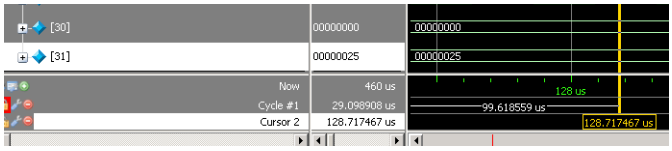


Рисунок 5.1 – Значение регистра x31 после выполнения программы

Символом #! в программе помечена команда addi x31, x31, x2. Рассмотрим стадии ее выполнения. На рисунке 5.1 представлены стадии выборки (6 такт), диспетчеризации (7 такт) и декодирования (8 такт).

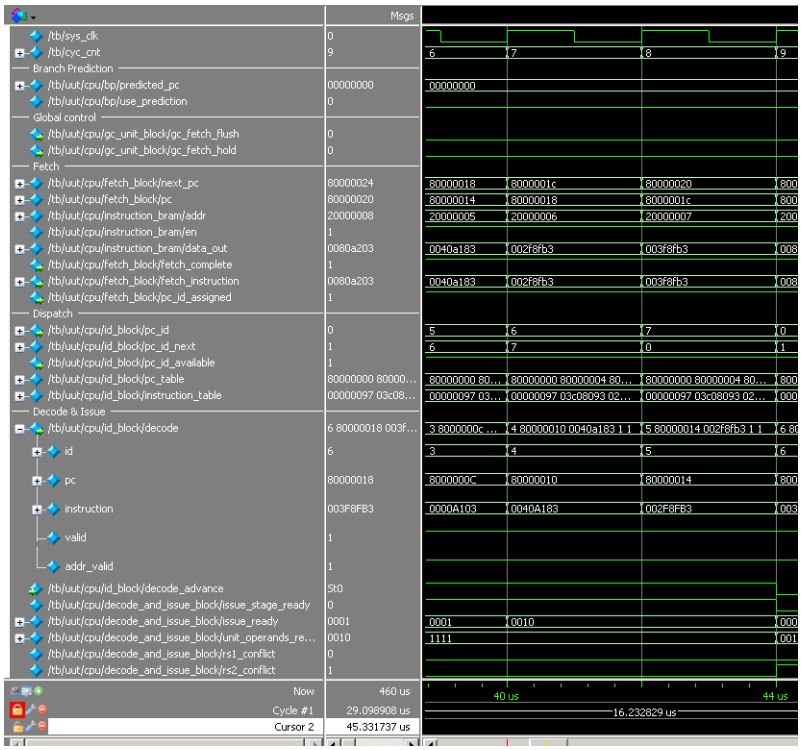


Рисунок 5.2 – Выборка, диспетчеризация и декодирование команды

В 9 такте команда планируется на выполнение, но так как сигнал rs2_conflict равен 1 возникает конфликт. В следующем такте этот конфликт разрешается, управление передается блоку ALU, о чем свидетельтсвует совпадение

issue/id и unit_wb[0]/id, они равны 5. В 10 такте выполнение завершается, о чем свидетельствует 1 на сигнале unit_wb[0]/done, и в следующем такте переходит к выполнению следующей команды.

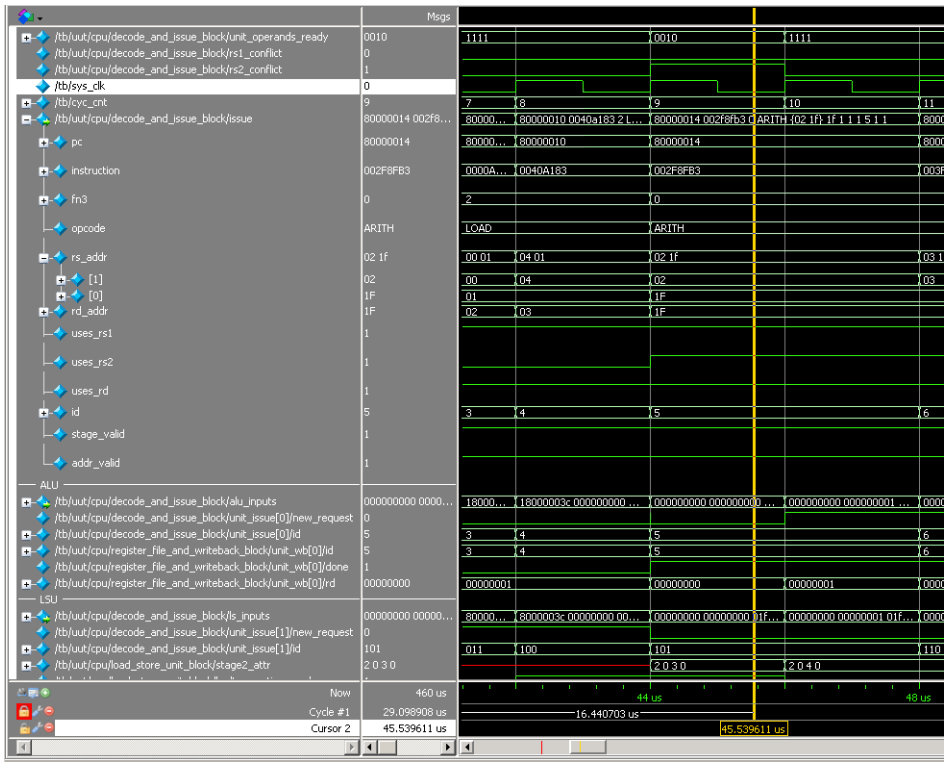


Рисунок 5.3 – Планирование на выполнение и выполнение команды

Ниже представленна временная диаграмма сигналов выполнения программы по варианту 12.

Адрес	Код команды	Команда	id	Номер такта																																												
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
80000000<_start>	00000097	auipc x1,0x0	0	F	ID	D	AL																																											
80000004	03c08093	addi x1,x1,60#8000003c<_x>	1		F	ID	D	AL																																										
80000008	00200a13	addi x20,x1,32	2			F	ID	D	AL																																									
8000000c<lp>	0000a103	lw x2,0(x1)	3				F	ID	D	M1	M2	M3																																				
80000010	0040a183	lw x3,4(x1)	4					F	ID	D	M1	M2	M3																																			
80000014	002f8fb3	add x31,x31,x2	5						F	ID	D	C	AL																																			
80000018	003f8fb3	add x31,x31,x3	6							F	ID	W	D	AL																																		
8000001c	0080a203	lw x4,8(x1)	7								F	ID	W	D	M1	M2	M3																															
80000020	00c0a283	lw x5,12(x1)	0									F	ID	W	D	M1	M2	M3																														
80000024	004f8fb3	add x31,x31,x4	1										F	ID	W	D	C	AL																														
80000028	005f8fb3	add x31,x31,x5	2											F	ID	W	W	D	AL																													
8000002c	01008093	addi x1,x1,16	3												F	ID	W	W	D	AL																												
80000030	fd409ee3	bne x1,x20,8000000c<lp>	4													F	ID	W	W	D	B																											
80000034	001f8f93	addi x31,x31,1	5														F	ID	W	W	D	X																										
80000038	0000006f	jal x0,80000038<lp2>	6															F	ID	W	W	DX																										
8000003c	00000001	<invalid operation>	7																F	ID	W	X																										
80000040	00000002	<invalid operation>	0																	F	ID	X																										
80000044	00000003	<invalid operation>	1																		F	X																										
80000048	00000004	<invalid operation>	2																			FX																										
8000000c<lp>	0000a103	lw x2,0(x1)	6																				F	ID	D	M1	M2	M3																				
80000010	0040a183	lw x3,4(x1)	7																					F	ID	D	M1	M2	M3																			
80000014	002f8fb3	add x31,x31,x2	0																						F	ID	D	C	Al																			
80000018	003f8fb3	add x31,x31,x3	1																							F	ID	W	D	AL																		
8000001c	0080a203	lw x4,8(x1)	2																								F	ID	W	D	M1	M2	M3															
80000020	00c0a283	lw x5,12(x1)	3																									F	ID	W	D	M1	M2	M3														
80000024	004f8fb3	add x31,x31,x4	4																										F	ID	W	D	C	AL														
80000028	005f8fb3	add x31,x31,x5	5																											F	ID	W	W	D	AL													
8000002c	01008093	addi x1,x1,16	6																												F	ID	W	W	D	AL												
80000030	fd409ee3	bne x1,x20,8000000c<lp>	7																													F	ID	W	W	D	B											
8000000c<lp>	0000a103	lw x2,0(x1)	0																														F	ID	W	W	D	X										
80000010	0040a183	lw x3,4(x1)	1																															F	ID	W	W	DX										
80000014	002f8fb3	add x31,x31,x2	2																																F	ID	W	X										
80000018	003f8fb3	add x31,x31,x3	3																																	F	ID	X										
8000001c	0080a203	lw x4,8(x1)	4																																		F	X										
80000020	00c0a283	lw x5,12(x1)	5																																			FX										
80000034	001f8f93	add x31,x31,1	1																																				F	ID	D	AL						
80000038	0000006f	jal x0,80000038<lp2>	2																																					F	ID	D	B					
8000003c	00000001	<invalid operation>	3																																						F	ID	D	X				
80000040	00000002	<invalid operation>	4																																							F	ID	DX				
80000044	00000003	<invalid operation>	5																																								F	X				
80000048	00000004	<invalid operation>	6																																									FX				
80000038<lp2>	0000006f	jal x0,8000003c<forever>	4																																										F	ID	D	B	
80000038<lp2>	0000006f	jal x0,8000003c<forever>	5																																											F	ID	D	B
Адрес	Код команды	Команда	id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
				Номер такта																																												

Рисунок 5.4 – Временная диаграмма сигналов выполнения программы

Из трассы видно, что выполнение программы заканчивается выполнением команды add x31, x31, 1 в 39 такте. Тоесть выполнение программы происходит с 1 по 39 такт. Из них планирование на выполнение не происходило в течение 16 тактов (7, 8, 12, 13, 19 - 24, 28, 29, 35 - 38). Это составляет 41%. Чтобы сократить время простоя можно провести оптимизацию путем перестановки команд.

В качестве оптимизации было выбрано перемещение строк загрузки данных друг к другу. Оптимизированный код представлен на листинге ниже.

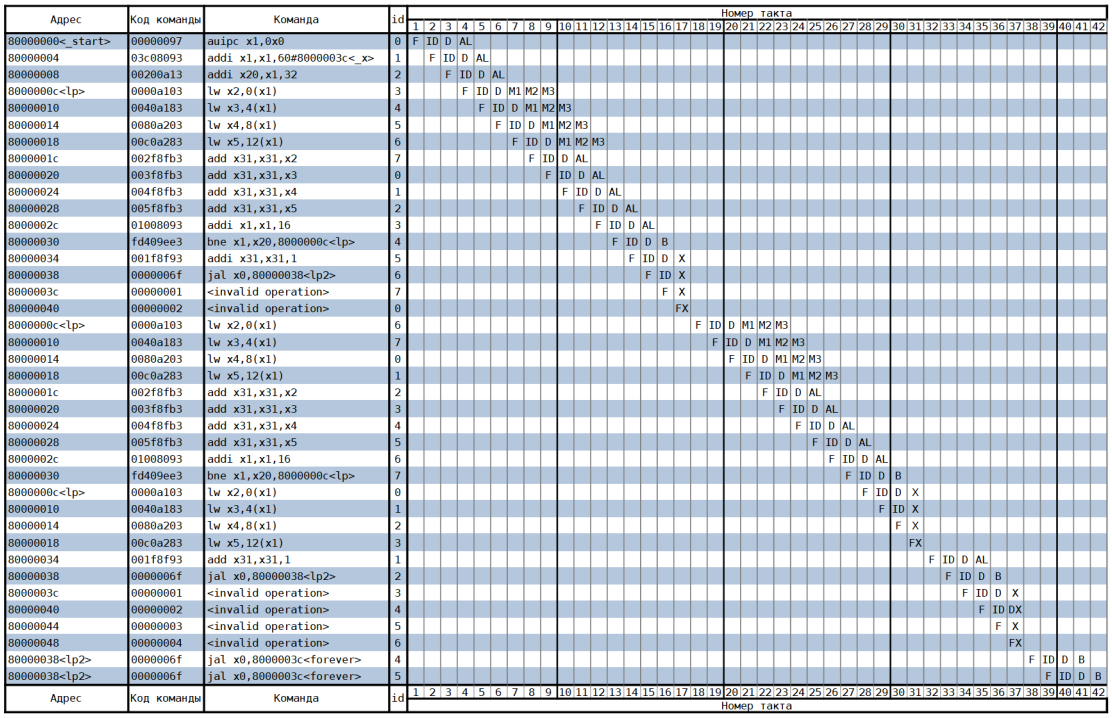
Листинг 5.1 – Код оптимизированной программы

```
  .section .text
        .globl _start;
        len = 8 #array length
        enroll = 4 #count of elems in one loop
        elem_sz = 4 #size of one element

_start:
        la x1, _x
        addi x20, x1, elem_sz*len #address of last elem
lp:
        lw x2, 0(x1)
        lw x3, 4(x1)
        lw x4, 8(x1)
        lw x5, 12(x1)
        add x31, x31, x2 #!
        add x31, x31, x3
        add x31, x31, x4
        add x31, x31, x5
        addi x1, x1, elem_sz*enroll
        bne x1, x20, lp
        addi x31, x31, 1
lp2: j lp2

        .section .data
_x:     .4byte 0x1
        .4byte 0x2
        .4byte 0x3
        .4byte 0x4
        .4byte 0x5
        .4byte 0x6
        .4byte 0x7
        .4byte 0x8
```

Временная диаграмма сигналов выполнения представленна ниже.



Адрес	Код команды	Команда	id	Номер такта 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
80000000<_start>	00000097	auipc x1,0x0	0	F	ID	D	AL																																						
80000004	03c08093	addi x1,x1,60#8000003c<_x>	1		F	ID	D	AL																																					
80000008	00200a13	addi x20,x1,32	2			F	ID	D	AL																																			
8000000c<lp>	0000a103	lw x2,0(x1)	3				F	ID	D	M1	M2	M3																																
80000010	0040a183	lw x3,4(x1)	4					F	ID	D	M1	M2	M3																															
80000014	0080a203	lw x4,8(x1)	5						F	ID	D	M1	M2	M3																													
80000018	00c0a283	lw x5,12(x1)	6							F	ID	D	M1	M2	M3																											
8000001c	002f8fb3	add x31,x31,x2	7								F	ID	D	AL																											
80000020	003f8fb3	add x31,x31,x3	0									F	ID	D	AL																										
80000024	004f8fb3	add x31,x31,x4	1										F	ID	D	AL																									
80000028	005f8fb3	add x31,x31,x5	2											F	ID	D	AL																								
8000002c	01008093	addi x1,x1,16	3												F	ID	D	AL																							
80000030	fd409ee3	bne x1,x20,8000000c<lp>	4													F	ID	D	B																					
80000034	001f8f93	addi x31,x31,1	5														F	ID	D	X																				
80000038	0000006f	jal x0,80000038<lp2>	6															F	ID	X																				
8000003c	00000001	<invalid operation>	7																F	X																				
80000040	00000002	<invalid operation>	0																	FX																				
8000000c<lp>	0000a103	lw x2,0(x1)	6																		F	ID	D	M1	M2	M3														
80000010	0040a183	lw x3,4(x1)	7																			F	ID	D	M1	M2	M3													
80000014	0080a203	lw x4,8(x1)	0																				F	ID	D	M1	M2	M3												
80000018	00c0a283	lw x5,12(x1)	1																					F	ID	D	M1	M2	M3											
8000001c	002f8fb3	add x31,x31,x2	2																						F	ID	D	AL												
80000020	003f8fb3	add x31,x31,x3	3																							F	ID	D	AL											
80000024	004f8fb3	add x31,x31,x4	4																								F	ID	D	AL										
80000028	005f8fb3	add x31,x31,x5	5																									F	ID	D	AL									
8000002c	01008093	addi x1,x1,16	6																										F	ID	D	AL								
80000030	fd409ee3	bne x1,x20,8000000c<lp>	7																											F	ID	D	B							
8000000c<lp>	0000a103	lw x2,0(x1)	0																												F	ID	D	X						
80000010	0040a183	lw x3,4(x1)	1																													F	ID	X						
80000014	0080a203	lw x4,8(x1)	2																														F	X						
80000018	00c0a283	lw x5,12(x1)	3																															FX						
80000034	001f8f93	add x31,x31,1	1																																F	ID	D	AL		
80000038	0000006f	jal x0,80000038<lp2>	2																																	F	ID	D	B	
8000003c	00000001	<invalid operation>	3																																		F	ID	D	X	
80000040	00000002	<invalid operation>	4																																			F	ID	DX	
80000044	00000003	<invalid operation>	5																																				F	X	
80000048	00000004	<invalid operation>	6																																					FX	
80000038<lp2>	0000006f	jal x0,8000003c<forever>	4																																						F	ID	D	B
80000038<lp2>	0000006f	jal x0,8000003c<forever>	5																																							F	ID	D	B
Адрес	Код команды	Команда	id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
				Номер такта																																								

Рисунок 5.5 – Временная диаграмма сигналов выполнения оптимизированной программы

Из трассы видно, что выполнение программы заканчивается выполнением команды add x31, x31, 1 в 35 такте. Тоесть выполнение программы происходит с 1 по 35 такт. Из них планирование на выполнение не происходило в течение 12 тактов (7, 8, 17 - 22, 31 - 34). Это составляет 31%. Удалось сократить простой на 10%.