Министерство науки и высшего образования Российской Федерации



Федеральное государственное вюджетное образовательное учреждение высшего образования Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет) $(M\Gamma T Y \text{ им. H.Э. Баумана})$

ФАКУЛЬТЕТ	«Информатика и системы управления»
КАФЕДРА	«Программное обеспечение ЭВМ и информационные технологии»
НАПРАВЛЕНІ	ИЕ ПОДГОТОВКИ «09.03.04 Программная инженерия»

ОТЧЕТ по лабораторной работе №2

Название:	Изучение прин	ципов работы микропроцессор	ного ядра RISC-V
Дисциплина:		Архитектура ЭВМ	
Студент	<u>ИУ7-54Б</u> Группа	Подпись, дата	С. Д. Параскун и. о. Фамилия
Преподаватель		—————————————————————————————————————	А. Ю. Попов и. о. Фамилия

Цели работы

Основной целью работы является ознакомление с принципами функционирования, построения и особенностями архитектуры суперскалярных конвейерных микропроцессоров. Дополнительной целью работы является знакомство с принципами проектирования и верификации сложных цифровых устройств с использованием языка описания аппаратуры SystemVerilog и ПЛИС.

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
10 lp:
           lw x2, 0(x1)
11
           lw x3, 4(x1)
12
           add x31, x31, x2 #!
13
           add x31, x31, x3
           lw x4, 8(x1)
15
           lw x5, 12(x1)
16
           add x31, x31, x4
           add x31, x31, x5
18
           addi x1, x1, elem_sz*enroll
19
           bne x1, x20, lp
20
           addi x31, x31, 1
22 lp2: j lp2
23
           .section .data
          .4byte 0x1
25
           .4byte 0x2
26
           .4byte 0x3
           .4byte 0x4
28
           .4byte 0x5
29
           .4byte 0x6
           .4byte 0x7
31
           .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

```
1 #define len 8
  #define enroll 4
3 #define elem_sz 4
4 int _x[]={1,2,3,4,5,6,7,8};
  void _start() {
      int *x1 = _x;
      int x20 = x1 + len;
      do {
9
        int x2 = x1[0];
         int x3 = x1[1];
11
        x31 += x2;
12
         x31 += x3;
13
         int x4 = x1[2];
14
        int x5 = x1[3];
15
        x31 += x4;
        x31 += x5;
17
        x1 += enroll;
18
      } while(x1 != x20);
      x31++;
20
      while (1) {}
21
22 }
```

Листинг 1.3 – Дизассемблированный листинг рассматриваемой

```
программы
 80000000 <_start>:
2 80000000:
                   00000097
                                                      x1,0x0
                                             auipc
3 80000004:
                   03 c08093
                                                      x1,x1,60 # 8000003c <_x>
                                             addi
  80000008:
                   02008a13
                                             addi
                                                      x20,x1,32
6 8000000c <1p>:
 8000000c:
                   0000a103
                                             lw
                                                      x2,0(x1)
8 80000010:
                   0040a183
                                                      x3,4(x1)
                                             lw
9 80000014:
                   002f8fb3
                                             add
                                                      x31,x31,x2
10 80000018:
                   003f8fb3
                                             add
                                                      x31,x31,x3
11 8000001c:
                   0080a203
                                                      x4,8(x1)
                                             lw
12 80000020:
                                                      x5,12(x1)
                   00c0a283
                                             lw
                                                      x31,x31,x4
13 80000024:
                   004f8fb3
                                             add
14 80000028:
                                                      x31,x31,x5
                   005f8fb3
                                             add
15 8000002c:
                   01008093
                                                      x1,x1,16
                                             addi
16 80000030:
                                                      x1,x20,8000000c <lp>
                   fd409ee3
                                             bne
17 80000034:
                                                      x31,x31,1
                   001f8f93
                                             addi
19 80000038 <1p2>:
20 80000038:
                   000006f
                                             jal
                                                      x0,80000038 <1p2>
```

На рисунке ниже рассмотренна временная диаграмма стадий выборки и диспетчеризации команды 8000000с 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).

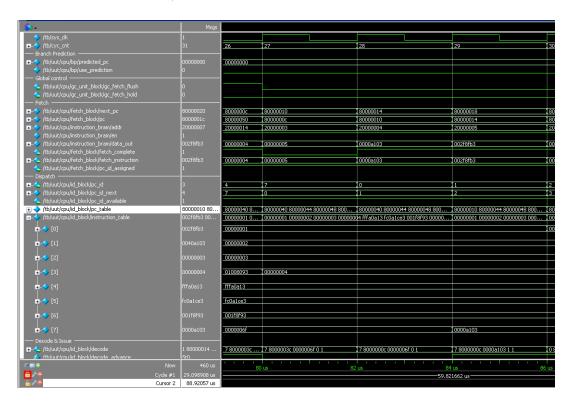


Рисунок 2.1 – Временная диаграмма выполнения стадий выборки и диспетчеризации команды 8000000с на 2-й итерации

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.

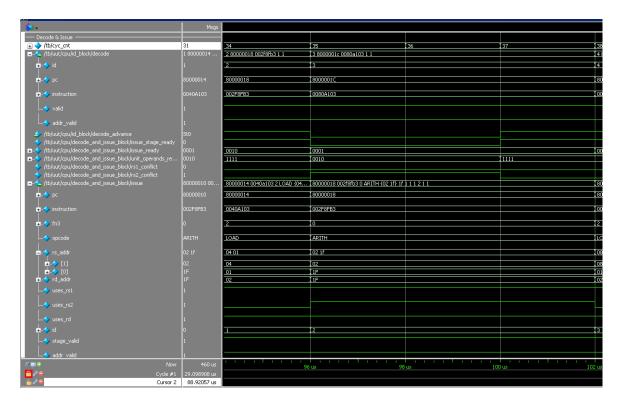


Рисунок 3.1 – Временная диаграмма выполнения стадий декодирования и планирования на выполнение команды 80000018 на 2-й итерации

На рисунке ниже рассмотренна временная диаграмма стадии выполнения команды 8000002с 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.

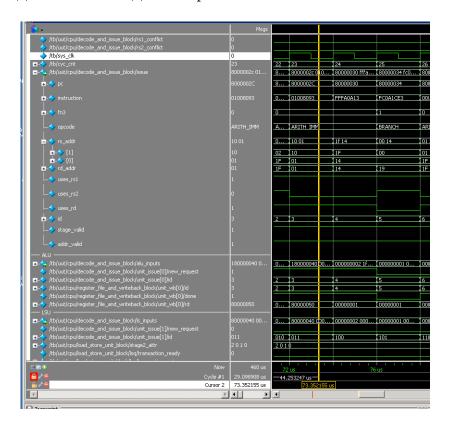


Рисунок 4.1 – Временная диаграмма выполнения стадии выполнения команды 8000002с на 1-й итерации

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 х31 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).

1.	Msgs				
♦ /tb/sys_clk	0				
⊨ √ /tb/cyc_cnt	9	6	7	[8	=
├� /tb/uut/cpu/bp/predicted_pc	00000000	00000000			
/tb/uut/cpu/bp/use prediction	0				
- Global control					
/tb/uut/cpu/gc_unit_block/gc_fetch_flush	0				
/tb/uut/cpu/gc_unit_block/gc_fetch_hold	0				
Fetch —					
	80000024	80000018	18000001c	180000020	
	80000020	80000014	180000018	18000001c	
- /tb/uut/cpu/instruction bram/addr	20000008	20000005	20000006	20000007	
4 /tb/uut/cpu/instruction bram/en					
	0080a203	0040a183	1002f8fb3	1003f8fb3	
4 /tb/uut/cpu/fetch block/fetch complete					
-4 /tb/uut/cpu/fetch_block/fetch_instruction	0080a203	0040a183	1002f8fb3	I 003f8fb3	
4 /tb/uut/cpu/fetch block/pc id assigned					Ī
Dispatch —					
-4 /tb/uut/cpu/id_block/pc_id	0	5	6	17	
- /tb/uut/cpu/id block/pc id next		6	7	lo	Ē
4 /tb/uut/cpu/id block/pc id available					
⊢/tb/uut/cpu/id_block/pc_table	80000000 80000	80000000 80	. 180000000 80000004 80	180000000 80000004 80	
- /tb/uut/cpu/id block/instruction table	00000097 03c08		. 100000097 03c08093 02		
- Decode & Issue					
-4. /tb/uut/cpu/id_block/decode	6 80000018 003f	3 8000000c	I 4 80000010 0040a183 1 1	I 5 80000014 002f8fb3 1 1	
I		3	14	-	
	P	3	4	5	
■-	80000018	8000000C	[80000010	[80000014	
•	003F8FB3	0000A103	I 0040A183	1002F8FB3	
└→ valid					
addr_valid					
/b/uut/cpu/id block/decode advance	StO				
tb/uut/cpu/decode_and_issue_block/issue_stage_ready	n -				
tb/uut/cpu/decode and issue block/issue ready	0001	0001	[0010		
tb/uut/cpu/decode and issue block/unit operands re		1111	,00.0		
/tb/uut/cpu/decode_and_issue_block/rs1_conflict	0				Ē
/tb/uut/cpu/decode and issue block/rs2 conflict	1				
. Something the state of the s	460 us	1 1 1	10 us	1 1 1 1	44
/ ⊖ Cycle #1	29.098908 us			32829 us	
Cursor 2	_		10.2		

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.

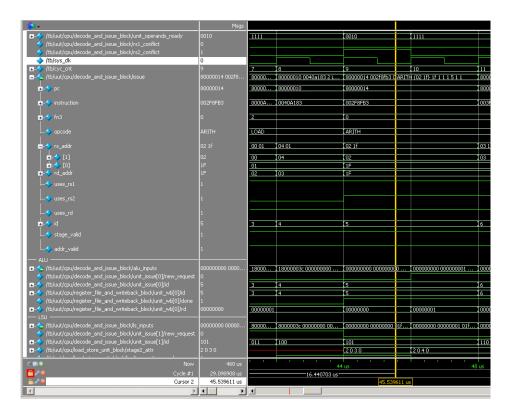


Рисунок 5.3 – Планирование на выполнение и выполнение команды

Ниже представленна временная диаграмма сигналов выполнения программы по варианту 12.

Адрес	Код команды	Команда	3.4																			lom	ep '	так	ста																		
- 14			id					6	7	8	9 1	0 11	1 12	13	14 1	15	6 1	7 18	3 19	20	21	22	23	24	25 2	26 2	7 2	8 29	36	31	32	33	34	35 3	36 3	7 38	39	40	41 4	12 4	13 4	4 4	15 4
80000000<_start>	00000097	auipc x1,0x0				D A					Т																															П	
80000004	03c08093	addi x1,x1,60#8000003c<_x>	1			ED E					\perp																							\perp				Ш		\perp	\perp		
80000008	00200a13	addi x20,x1,32	2			FI	D	AL			П									ı									Н														
8000000c <lp></lp>	0000a103	lw x2,0(x1)	3			F	ID	D	М1	M2 N	13																																
80000010	0040a183	lw x3,4(x1)	4				F	ID	D	M1 N	12 M	3								ı									П														
80000014	002f8fb3	add x31,x31,x2	5	П			Т	F	ID	D	C A	L	Т				Т	Т	П	Г				П	П			Т	Т					Т	Т	Т		П	П	Т	Т	Т	Т
80000018	003f8fb3	add x31,x31,x3	6						F	ID	W E) AL																															
8000001c	0080a203	lw x4,8(x1)	7	П		Т	Т			F J	DV	/ D	M1	M2	мз		Т	Т	Т					П	П		Т	Т	Т	П	Г			Т	Т	Т		П		Т	Т	Т	Т
80000020	00c0a283	lw x5,12(x1)	0								FI	D W	D	М1	M2 N	13																											
80000024	004f8fb3	add x31,x31,x4	1	П	╛	т	Т			т	F	10	W	D	C A	AL.	Т	Т	т				╛	П	т	т	т	Т	Т	Т	г			т	т			П		т	т	т	т
80000028	005f8fb3	add x31,x31,x5	2												W		L																										
8000002c	01008093	addi x1,x1,16	3		7	т	т			_	т	1			W				т				7	7	_				т					т				П	-	т	т	т	т
80000030	fd409ee3	bne x1,x20,8000000c <lp></lp>	4												ID																												
80000034	001f8f93	addi x31,x31,1	5								т					D V													т					-						т	7		т
80000034	00010193	jal x0,80000038 <lp2></lp2>	6													FI													L														
8000003c	00000001	<pre><invalid operation=""></invalid></pre>	7															D W											П														٠
80000030	00000001	<pre><invalid operation=""></invalid></pre>	0								-					,		: II																					_				
			_								-						1	F	_										н					-						4	+		4
80000044	00000003	<pre><invalid operation=""></invalid></pre>	1								-							F																_					_	-	_		4
80000048	00000004	<invalid operation=""></invalid>	2																FX	_														4						4	4		4
8000000c <lp></lp>	0000a103	lw x2,0(x1)	6			_					-	_				_	_	-		F	ID					_	_		н					_	_			Ш	_	4	4	_	4
80000010	0040a183	lw x3,4(x1)	7								1										F				M2 I															4			
80000014	002f8fb3	add x31,x31,x2	0			_				_	_					_	_	_	_	ш	ш				C		_		_			Ш		_	_		ш	Ш	_	_	_	_	_
80000018	003f8fb3	add x31,x31,x3	1																							D A																	
8000001c	0080a203	lw x4,8(x1)	2								\perp																	1 M2						\perp				Ш		\perp	\perp		\perp
80000020	00c0a283	lw x5,12(x1)	3								н									ı								M)															
80000024	004f8fb3	add x31,x31,x4	4																							FΙ	D V	۷ D	C	AL													
80000028	005f8fb3	add x31,x31,x5	5																							F	FI	D W															
8000002c	01008093	addi x1,x1,16	6	П		Т	Т			\neg	Т	Т	Т			Т	Т	Т	П					П	П	Т	F	- II	W	W	D	AL		Т	Т	Т		П	П	Т	Т	Т	Т
80000030	fd409ee3	bne x1,x20,8000000c <lp></lp>	7																									F	10	W	W	D	В										
8000000c <lp></lp>	0000a103	lw x2,0(x1)	0	П	П	Т	Т	П		Т	Т	Т	Т	П	П	Т	Т	Т	Т	Г			П	П	Т	Т	Т	Т	F	ID	w	w	D	Х	Т	Т		П	П	Т	Т	Т	Т
80000010	0040a183	lw x3,4(x1)	1																											F	ID	W	W	DX									
80000014	002f8fb3	add x31.x31.x2	2		7	т	т			_	т	Т			П		т	т	т				7	7	т				т		F		w							т	т	т	т
80000018	003f8fb3	add x31,x31,x3	3																														ID										
8000001c	0080a203	lw x4,8(x1)	4								т																		Т				F							т	т	т	т
80000020	00c0a283	lw x5,12(x1)	5																															FX									
80000034	001f8f93	add x31,x31,1	1								т																		т						FΤ	D D	ΔI			т	7	7	т
80000034	00010195	ial x0,80000038 <lp2></lp2>	2																																	= ID							
8000003c	00000001	<pre><invalid operation=""></invalid></pre>	3								т																		Н						т.			D	V	-			7
80000030	00000001	<pre><invalid operation=""></invalid></pre>	4								-																									1	F	ID			-		
80000040	00000002	<pre><invalid operation=""> <invalid operation=""></invalid></invalid></pre>	5																										Н					-			r	F			-		٠
						-					-					-										-								_					FX	-	4	-	4
80000048	00000004	<invalid operation=""></invalid>	6																																								
80000038 <lp2></lp2>	0000006f	jal x0,8000003c <forever></forever>	4								-																		L					_					_		D D		
80000038 <lp2></lp2>	0000006f	jal x0,8000003c <forever></forever>	5				4								يليا						Щ								ļ.										بلب		F I		
Адрес	Код команды	Команда	lid	1	2	3 4	5	6	7	8	9 [1	$\theta 11$	L 12	13	14	15	6 1	/ 18	3 19	20	21			24	25 2 ста	26 2	1 2	8 29) [36	31	32	33	34	<u>35 3</u>	36 3	/ 38	39	40	41 4	12 4	3 4	4 4	5 4

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
10 lp:
           lw x2, 0(x1)
11
           lw x3, 4(x1)
12
           lw x4, 8(x1)
13
           lw x5, 12(x1)
14
           add x31, x31, x2 #!
15
           add x31, x31, x3
           add x31, x31, x4
17
           add x31, x31, x5
18
           addi x1, x1, elem_sz*enroll
           bne x1, x20, lp
20
           addi x31, x31, 1
21
22 1p2: j 1p2
23
           .section .data
24
           .4byte 0x1
25
  _x:
           .4byte 0x2
26
           .4byte 0x3
^{27}
           .4byte 0x4
28
           .4byte 0x5
29
           .4byte 0x6
30
           .4byte 0x7
31
           .4byte 0x8
32
```

Временная диаграмма сигналов выполнения представленна ниже.

Адрес	Код команды	Команда	id		0				-	0	o la	0 4	0 00	ala c		4.5	4.0		100		мер				lor	0.0	0.7	00	00	20	24	201	no lo		E lo	o lo:	7 0	, lac	al e	0 4	140
80000000< start>	00000097	auipc x1,0x0				3 4 D AI		6	7	8	9 1	0 1	1 12	2 13	14	15	16	17	18 1	92	0 2	1 2	2 2	3 24	125	26	27	28	29	30	31	32 3	33 3	4 3	5 3	6 3	7 38	3 39	9 41	0 4:	42
_			0			D AI														-					Н						-	-						-	н		
80000004	03c08093	addi x1,x1,60#8000003c<_x>	1			F II					-	-		-					-	4		-	-		-	Н					_	-			-	-	+	-	-	-	-
80000008	00200a13	addi x20,x1,32	2				ID													н											9	4							н		
8000000c <lp></lp>	0000a103	lw x2,0(x1)	3			-														4			-									_					+	-	-		
80000010	0040a183	lw x3,4(x1)	4				F			M1										н																			н		
80000014	0080a203	lw x4,8(x1)	5					F		D										4		-	_								_	_				_	_	_	_	-	_
80000018	00c0a283	lw x5,12(x1)	6										2 M	3						н												4							н		
8000001c	002f8fb3	add x31,x31,x2	7		_		_			F I				_					_	4	_	_	_		_	_					_	4	_	_	-	_	_	┺	1	_	_
80000020	003f8fb3	add x31,x31,x3	0) Al							-												4							н		
80000024	004f8fb3	add x31,x31,x4	1		_		_				- 1		D D						_	4	-	-	-		_	_					_	_	_		-	-	_	_	1	-	_
80000028	005f8fb3	add x31,x31,x5	2									F	11							1																			н		
8000002c	01008093	addi x1,x1,16	3								4		F		D					4											_	_					_	_	L		_
80000030	fd409ee3	bne x1,x20,8000000c <lp></lp>	4											F	ID					1																			L		
80000034	001f8f93	addi x31,x31,1	5		_						_	_			F		D		_	4		_	_								_	_	_	_	_	_	_	┸	_	_	\perp
80000038	0000006f	jal x0,80000038 <lp2></lp2>	6													F	ID	_		1																					
8000003c	00000001	<invalid operation=""></invalid>	7		_				ш		_	_		_				X	_	4	_	_	_		ш						_	_		_	_	_	_	┸	Т	_	┸
80000040	00000002	<invalid operation=""></invalid>	0															FX		П																					
8000000c <lp></lp>	0000a103	lw x2,0(x1)	6								_								F I												_	_					┸	┸	Т		┸
80000010	0040a183	lw x3,4(x1)	7																		D D																				
80000014	0080a203	lw x4,8(x1)	0																	_ !	= II										_	_					┸	┸	L		┸
80000018	00c0a283	lw x5,12(x1)	1																		F		D D																		
8000001c	002f8fb3	add x31,x31,x2	2																			F	11																		
80000020	003f8fb3	add x31,x31,x3	3																				F	II	D	AL													П		
80000024	004f8fb3	add x31,x31,x4	4																					F	10																
80000028	005f8fb3	add x31,x31,x5	5																						F	ID													П		
8000002c	01008093	addi x1,x1,16	6																							F	ID														
80000030	fd409ee3	bne x1,x20,8000000c <lp></lp>	7																	П							F		D										Н		
8000000c <lp></lp>	0000a103	lw x2,0(x1)	0																									F	ID												
80000010	0040a183	lw x3,4(x1)	1								П									н									F										Н		
80000014	0080a203	lw x4,8(x1)	2								П									П										F	X	П							П		
80000018	00c0a283	lw x5,12(x1)	3																	П											FX								П		
80000034	001f8f93	add x31,x31,1	1				П				Т	Т	Т	Т						Т		Т	Т	Т	Г				П		П	F]				Т	Т	Т	Т	Т	Т
80000038	0000006f	jal x0,80000038 <lp2></lp2>	2																														FΙ	D E	В	3					
8000003c	00000001	<invalid operation=""></invalid>	3		П		П				Т	Т	Т	Т		П			Т	Т	Т	Т	Т	Т	Т				П		Т	Т	F	I	D D	X	(Т	Т	Т	Т
80000040	00000002	<invalid operation=""></invalid>	4																															F	I	D D	X				
80000044	00000003	<invalid operation=""></invalid>	5			T					Т	Т	T						Т	Т	T	Т	T	Г								Т	T	T	F	×	(Г	T	Г
80000048	00000004	<invalid operation=""></invalid>	6																																	F	X		L		
80000038<1p2>	0000006f	jal x0,8000003c <forever></forever>	4								Т	Т	T							Т		Τ	T	Г										T	Т	T	F	ID	D	В	
80000038 <lp2></lp2>	0000006f	jal x0,8000003c <forever></forever>	5																																			F			В
Адрес	Код команды	Команда	id	1	2	3 4	5	6	7	8	9 1	0 1	1 12	2 13	14	15	16	17	18 1						25	26	27	28	29	30	31	32 3	33 3	4 3	5 3	6 3	7 38	39	9 41	0 4:	L 42
Адрес	под коминды	Команда	10		_	_		_												Н	мер	Т	акт	a	_					_	_	_					_	_			

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%.