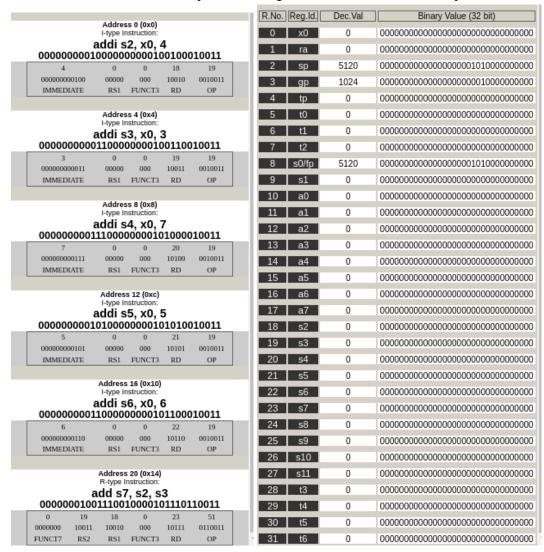
Roteiro 8 - Laboratório de Organização e Arquitetura de Computadores

Aluno: Tarso Jabbes Lima de Oliveira

RESPOSTAS PARA O PROGRAMA 1

1.2.a

Conteúdo da memória de instruções e dos registradores no início da execução:



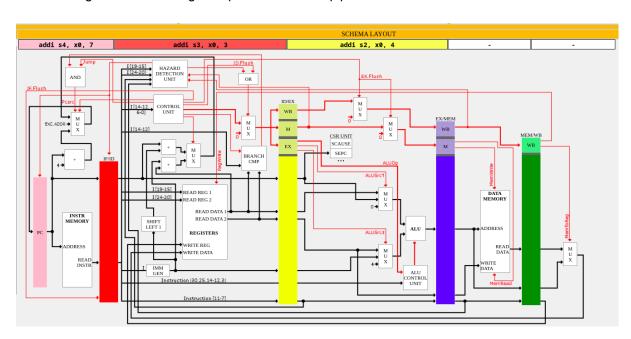
Podemos observar que os únicos valores nos registradores ao início são o de Stack Pointer, Group Pointer e o s0/fp. Além disso, as instruções foram carregadas corretamente na memória de instruções.

Conteúdo da memória de instruções e dos registradores ao final da execução:

				•				
		s 0 (0x0) struction:			R.No.	Reg.Id.	Dec.Val	Binary Value (32 bit)
	addi sa	2, x0,	4					
0000000001					0	х0	0	000000000000000000000000000000000000000
4 000000000100	00000	000	18 10010	19 0010011	1	ra	0	000000000000000000000000000000000000000
IMMEDIATE		FUNCT3	RD	OP	2	sp	5120	00000000000000000001010000000
					3	gp	1024	000000000000000000000000000000000000000
		s 4 (0x4) struction:			4	tp	0	000000000000000000000000000000000000000
	addi s		3		5	tO	0	000000000000000000000000000000000000000
0000000000	0110000	000010	001100		6	t1	0	000000000000000000000000000000000000000
3	00000	000	19 10011	19 0010011	7	t2	0	000000000000000000000000000000000000000
IMMEDIATE		FUNCT3	RD	OP	8	s0/fp	5120	00000000000000000001010000000
					9	s1	0	000000000000000000000000000000000000000
		s 8 (0x8) struction:			10	a0	0	000000000000000000000000000000000000000
	addi s		7		11	a1	0	000000000000000000000000000000000000000
0000000000				19	12	82	0	000000000000000000000000000000000000000
000000000111	00000	000	20 10100	0010011	13	a3	0	000000000000000000000000000000000000000
IMMEDIATE	RS1	FUNCT3	RD	OP	14	84	0	000000000000000000000000000000000000000
					15	a5	0	000000000000000000000000000000000000000
		s 12 (0xc) struction:			16	a6	0	000000000000000000000000000000000000000
	addi s				10			
0000000001			010100 21		17	a7	0	000000000000000000000000000000000000000
5 000000000101	00000	000	10101	19 0010011	18	s2	4	000000000000000000000000000000000000000
IMMEDIATE	RS1	FUNCT3	RD	OP	19	s3	3	000000000000000000000000000000000000000
					20	s4	7	000000000000000000000000000000000000000
		16 (0x10) struction:	J		21	s5	5	000000000000000000000000000000000000000
	addi se	6, x0,	6		22	s6	6	000000000000000000000000000000000000000
000000000			011000 22	19	23	s7	7	000000000000000000000000000000000000000
	0	000	10110	0010011	24	s8	0	000000000000000000000000000000000000000
	00000		RD	OP	25	s9	0	000000000000000000000000000000000000000
000000000110 IMMEDIATE	00000 RS1	FUNCT3						
00000000110	RS1				26	s10	0	000000000000000000000000000000000000000
00000000110	RS1	20 (0x14)			26 27	s10 s11	0	
00000000110 IMMEDIATE	Address R-type Inc	20 (0x14) histruction: ', S2, S	3			s11	-	000000000000000000000000000000000000000
00000000110 IMMEDIATE	Address R-type In add s7 0111001	20 (0×14) estruction: ', S2, S L00001(3 011101		27 28	s11 t3	0	000000000000000000000000000000000000000
00000000110 IMMEDIATE	Address R-type In add s7, 0111001	20 (0x14) histruction: ', S2, S	3	10011 51 0110011	27	s11	0	00000000000000000000000000000000000000

Ao final da execução, os registradores s2, s3, s4, s5, s6 e s7 sofreram alterações.

1.2.b Passagem em três estágios representativos do pipeline



Podemos observar que a primeira instrução está no estágio 3 do pipeline (estágio de execução), a segunda instrução está no estágio 2 (estágio de decodificação) e a terceira instrução está no estágio 1 (estágio de *fetch*).

1.2.c

EXECUTION TABLE													
FULL LOOPS 🗸	CPU Cycles												
Instruction	1	2	3	4	5	6	7	8	9	10			
addi s2, x0, 4	F	D	X	M	W								
addi s3, x0, 3		F	D	X	M	W							
addi s4, x0, 7			F	D	X	M	W						
addi s5, x0, 5				F	D	X	M	W					
addi s6, x0, 6					F	D	X	M	W				
add s7, s2, s3						F	D	X	M	W			

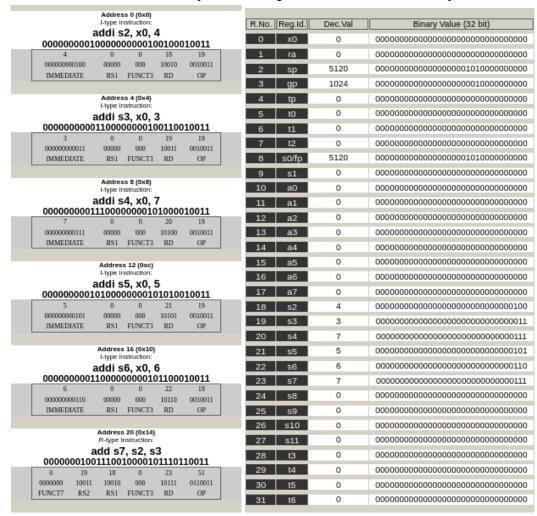
A tabela mostra que não houve conflito na execução das instruções.

1.2.d Ao total, foram gastos 10 ciclos de CPU para que o programa fosse executado.

2.2.aConteúdo da memória de instruções e dos registradores no início da execução:

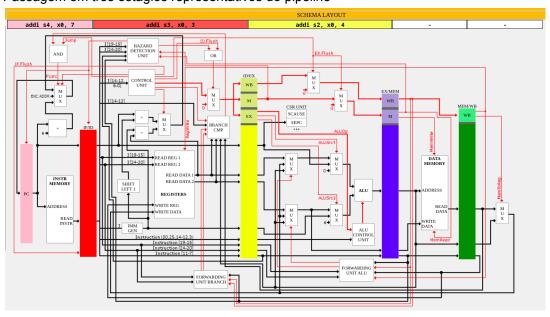
				R.No.	Den Id	Dec.Val	Binary Value (32 bit)
	Address 0 (
	addi s2, x			0	x0	0	000000000000000000000000000000000000000
	1000000000		010011	1	ra	0	000000000000000000000000000000000000000
4	0	0 18	19	2	sp	5120	000000000000000000101000000000
000100		000 10010	0010011	3	gp	1024	000000000000000000000000000000000000000
EDIATE	RS1 FUN	NCT3 RD	OP	4	tp	0	000000000000000000000000000000000000000
	Address 4 (5	t0	0	000000000000000000000000000000000000000
	addi s3, x			6	t1	0	000000000000000000000000000000000000000
	011000000		010011	7	t2	0	000000000000000000000000000000000000000
3		0 19	19	8	s0/fp	5120	000000000000000000010100000000
00000000011 MMEDIATE		000 10011 NCT3 RD	0010011 OP	9	s1	0	000000000000000000000000000000000000000
	KDI POI		01	10	a0	0	000000000000000000000000000000000000000
	Address 8 (11	a1	0	000000000000000000000000000000000000000
	addi s4, 2			11			
	1110000000		010011	12	a2	0	000000000000000000000000000000000000000
7		0 20	19	13	a3	0	000000000000000000000000000000000000000
00000111 MEDIATE		000 10100 NCT3 RD	0010011 OP	14	a4	0	000000000000000000000000000000000000000
	KDI TO	nors no	<u> </u>	15	a5	0	000000000000000000000000000000000000000
	Address 12			16	a6	0	000000000000000000000000000000000000000
	addi s5, x			17	a7	0	000000000000000000000000000000000000000
	101000000	,	010011	18	s2	0	000000000000000000000000000000000000000
5		0 21	19	19	s3	0	000000000000000000000000000000000000000
00000000101 MMEDIATE		000 10101 NCT3 RD	0010011 OP	20	s4	0	000000000000000000000000000000000000000
				21	s5	0	000000000000000000000000000000000000000
	Address 16 ((0x10) ction:		22	s6	0	000000000000000000000000000000000000000
	addi s6, x			23	s7	0	000000000000000000000000000000000000000
000000001	110000000					0	
	0	0 22 000 10110	19 0010011	24	s8		000000000000000000000000000000000000000
6	onono o		OP	25	s9	0	000000000000000000000000000000000000000
0000000110		NCT3 RD					
0000000110		NCT3 RD	O.	26	s10	0	000000000000000000000000000000000000000
00000000110	RS1 FUN	(0x14)	O.	26 27	s10 s11	0	
00000000110 IMMEDIATE	Address 20 (R-type Instru	(0x14) action:	or .	26 27 28			000000000000000000000000000000000000000
00000000110 MMEDIATE	RS1 FUN	(0x14) action: 62, S3		27	s11	0	000000000000000000000000000000000000000
0000000110 MMEDIATE	Address 20 (R-type Instru add s7, s: 011100100	(0x14) action: 62, S3		27 28	s11 t3	0	000000000000000000000000000000000000

Como não há hazards, os resultados permaneceram os mesmo com o Forwarding ativo. Conteúdo da memória de instruções e dos registradores ao final da execução:



Os mesmos valores encontrados com o Forwarding desativados foram encontrados agora.

2.2.b Passagem em três estágios representativos do pipeline



Encontramos os mesmos resultados que anteriormente: a primeira instrução está no estágio 3 do pipeline (estágio de execução), a segunda instrução está no estágio 2 (estágio de decodificação) e a terceira instrução está no estágio 1 (estágio de *fetch*).

2.2.c

EXECUTION TABLE												
FULL LOOPS 🗸		CPU Cycles										
Instruction	1	2	3	4	5	6	7	8	9	10		
addi s2, x0, 4	F	D	X	M	W							
addi s3, x0, 3		F	D	X	M	W						
addi s4, x0, 7			F	D	X	M	W					
addi s5, x0, 5				F	D	X	M	W				
addi s6, x0, 6					F	D	X	M	W			
add s7, s2, s3						F	D	X	M	W		

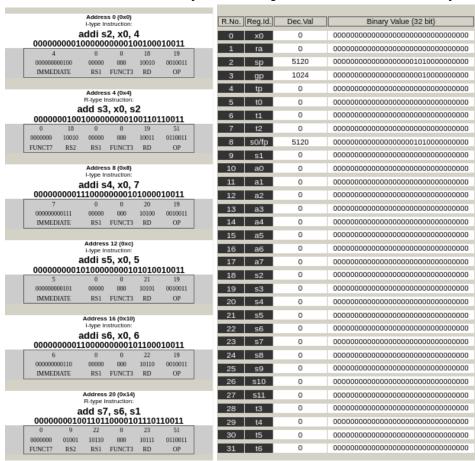
A tabela de execução mostra que não houve conflito durante a execução do programa 1.

2.2.d Ao total, também foram necessários 10 ciclos da CPU.

RESPOSTAS PARA O PROGRAMA 2

1.2.a

Conteúdo da memória de instruções e dos registradores no início da execução:



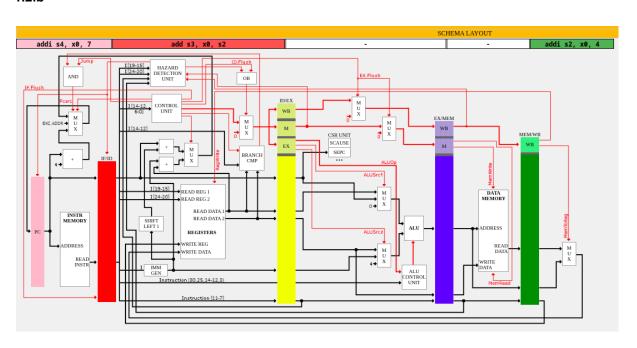
Podemos observar que os únicos valores nos registradores ao início são o de Stack Pointer, Group Pointer e o s0/fp. Além disso, as instruções foram carregadas corretamente na memória de instruções.

Conteúdo da memória de instruções e dos registradores ao final da execução:

		s 0 (0x0) struction:			R No	Reg.Id.	Dec.Val	Binary Value (32 bit)
	addi s		4				0	000000000000000000000000000000000000000
000000000				10011	0	x0		
4	0	0	18	19	1	ra	0	000000000000000000000000000000000000000
000000000100 IMMEDIATE	00000 RS1	000 FUNCT3	10010 RD	0010011 OP	2	sp	5120	00000000000000000010100000000
IMMEDIALE	K51	PUNCIS	KD	OF	3	gp	1024	000000000000000000000000000000000000000
		s 4 (0x4)			4	tp	0	000000000000000000000000000000000000000
		nstruction:	2		5	t0	0	000000000000000000000000000000000000000
00000010	add s3			110011	6	t1	0	000000000000000000000000000000000000000
0 18	0	0	19	51	7	t2	0	000000000000000000000000000000000000000
000000 10010		000	10011	0110011	8	s0/fp	5120	000000000000000000010100000000
JNCT7 RS2	RS1	FUNCT3	RD	OP	9	s1	0	000000000000000000000000000000000000000
	Addres	s 8 (0x8)			10	a0	0	000000000000000000000000000000000000000
		struction:					0	000000000000000000000000000000000000000
00000000	addi s			110011	11	a1		
7	0	0	20	19	12	a2	0	000000000000000000000000000000000000000
000000000111	00000	000	10100	0010011	13	a3	0	000000000000000000000000000000000000000
MMEDIATE	RS1	FUNCT3	RD	OP	14	a4	0	000000000000000000000000000000000000000
	Address	s 12 (0xc)			15	a5	0	000000000000000000000000000000000000000
	I-type In	struction:			16	a6	0	000000000000000000000000000000000000000
	addi s				17	a7	0	000000000000000000000000000000000000000
00000000	1010000	000010	21	19	18	s2	4	000000000000000000000000000000000000000
000000000101	00000	000	10101	0010011	19	s3	4	000000000000000000000000000000000000000
IMMEDIATE	RS1	FUNCT3	RD	OP	20	s4	7	000000000000000000000000000000000000000
		10 (010)			21	s5	5	000000000000000000000000000000000000000
		16 (0x10) struction:			22	s6	6	000000000000000000000000000000000000000
	addi s				23	s7	6	000000000000000000000000000000000000000
00000000	1100000	000010	22	19	24	s8	0	000000000000000000000000000000000000000
000000000110	00000	000	10110	0010011	25	so s9	0	
IMMEDIATE	RS1	FUNCT3	RD	OP				000000000000000000000000000000000000000
					26	s10	0	000000000000000000000000000000000000000
		20 (0x14) estruction:			27	s11	0	000000000000000000000000000000000000000
	add s7				28	t3	0	000000000000000000000000000000000000000
000000001					29	t4	0	000000000000000000000000000000000000000
0 9 00000 01001	22 10110	000	23 10111	51 0110011	30	t5	0	000000000000000000000000000000000000000
CT7 RS2	RS1	FUNCT3	RD	OP	31	t6	0	000000000000000000000000000000000000000

Ao final da execução, os registradores s2, s3, s4, s5, s6 e s7 sofreram alterações.

1.2.b



Como o Forwarding está desativado, só chegamos na etapa de 3 instruções acontecendo simultaneamente bem depois do que no problema 1, já que acontecem alguns conflitos e acontece o "stall" para lidá-los.

1.2.c

EXECUTION TABLE														
FULL LOOPS 🔻		CPU Cycles												
Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14
addi s2, x0, 4	F	D	X	M	W									
add s3, x0, s2		F	-	-	D	X	M	W						
addi s4, x0, 7					F	D	Х	M	W					
addi s5, x0, 5						F	D	X	M	W				
addi s6, x0, 6							F	D	X	M	W			
add s7, s6, s1								F	-	-	D	X	M	W

A tabela de execução mostra que houve conflito na execução da segunda e da sexta instrução. Como o Forwarding estava desativado, a execução dessas instruções foi pausada até que os conflitos fossem resolvidos.

1.2.d Ao total. foram necessários 14 ciclos de CPU.

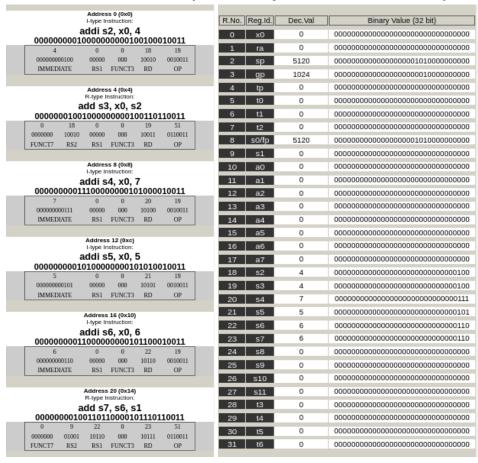
2.2.a

Conteúdo da memória de instruções e dos registradores no início da execução:

	Address I-type Inst				R.No.	Reg.Id.	Dec.Val	Binary Value (32 bit)
	addi s2		1		0	x0	0	000000000000000000000000000000000000000
0000000001					1	ra	0	000000000000000000000000000000000000000
4	00000	0	18 10010	19 0010011	2	SD	5120	0000000000000000000101000000000
MMEDIATE		FUNCT3	RD	OP	3	gp gp	1024	000000000000000000000000000000000000000
					4	tp	0	000000000000000000000000000000000000000
	Address R-type Ins				5	tO	0	000000000000000000000000000000000000000
á	add s3,		2			t1	0	000000000000000000000000000000000000000
00000100					6			
0 18 0000 10010	00000	0	19 10011	51 0110011	7	t2	0	000000000000000000000000000000000000000
NCT7 RS2		FUNCT3	RD	OP	8	s0/fp	5120	0000000000000000000101000000000
					9	s1	0	000000000000000000000000000000000000000
	Address I-type Inst				10	a0	0	000000000000000000000000000000000000000
	addi s4		7		11	a1	0	000000000000000000000000000000000000000
0000000001					12	a2	0	000000000000000000000000000000000000000
7	00000	000	20 10100	19 0010011	13	a3	0	000000000000000000000000000000000000000
MEDIATE		FUNCT3	RD	OP	14	a4	0	000000000000000000000000000000000000000
					15	a5	0	000000000000000000000000000000000000000
	Address : I-type Inst				16	a6	0	000000000000000000000000000000000000000
	addi s5		5		17	a7	0	000000000000000000000000000000000000000
00000001					18	s2	0	000000000000000000000000000000000000000
5 000000000101	00000	0	21 10101	19 0010011	19	s3	0	000000000000000000000000000000000000000
IMMEDIATE		FUNCT3	RD	OP	20	s4	0	000000000000000000000000000000000000000
					21	s5	0	000000000000000000000000000000000000000
	Address 1 I-type Inst	truction:			22		0	000000000000000000000000000000000000000
	addi s6		5			s6		
00000001					23	s7	0	000000000000000000000000000000000000000
6	00000	000	22 10110	19 0010011	24	s8	0	000000000000000000000000000000000000000
MMEDIATE		FUNCT3	RD	OP	25	s9	0	000000000000000000000000000000000000000
					26	s10	0	000000000000000000000000000000000000000
	Address 2 R-type Ins	20 (0x14)			27	s11	0	000000000000000000000000000000000000000
á	add s7.		1		28	t3	0	000000000000000000000000000000000000000
0000000010	0110110	000Ó10	11101		29	t4	0	000000000000000000000000000000000000000
0 9 00000 01001	22 10110	0	23 10111	51 0110011	30	t5	0	000000000000000000000000000000000000000
CT7 RS2		FUNCT3	RD	OP OF	31	t6	0	000000000000000000000000000000000000000

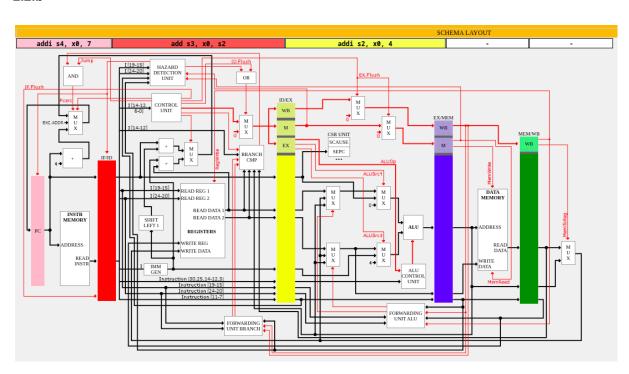
Encontramos os mesmo valores quando o Forwarding estava desativado.

Conteúdo da memória de instruções e dos registradores ao final da execução:



Novamente, os mesmos valores encontrados quando o Forwarding estava desativado.

2.2.b



Com o Forwarding ativo, conseguimos chegar bem antes no estado em que 3 instruções estão sendo executadas em estágios diferentes.

2.2.c

EXECUTION TABLE													
FULL LOOPS 💙	CPU Cycles												
Instruction	1	2	3	4	5	6	7	8	9	10			
addi s2, x0, 4	F	D	Х	M	W								
add s3, x0, s2		F	D	Х	M	W							
addi s4, x0, 7			F	D	Х	M	W						
addi s5, x0, 5				F	D	X	M	W					
addi s6, x0, 6					F	D	X	M	W				
add s7, s6, s1						F	D	X	M	W			

É possível ver que com o Forwarding ativo não há mais a pausa na execução de um pipeline quando temos um stall. Isso reduziu a quantidade de ciclos de CPU de precisamos para executar as instruções.

2.2.d Ao total, foram necessários 10 ciclos de CPU.