CS224

Lab 4

Section 3

Muhammet Abdullah Koç

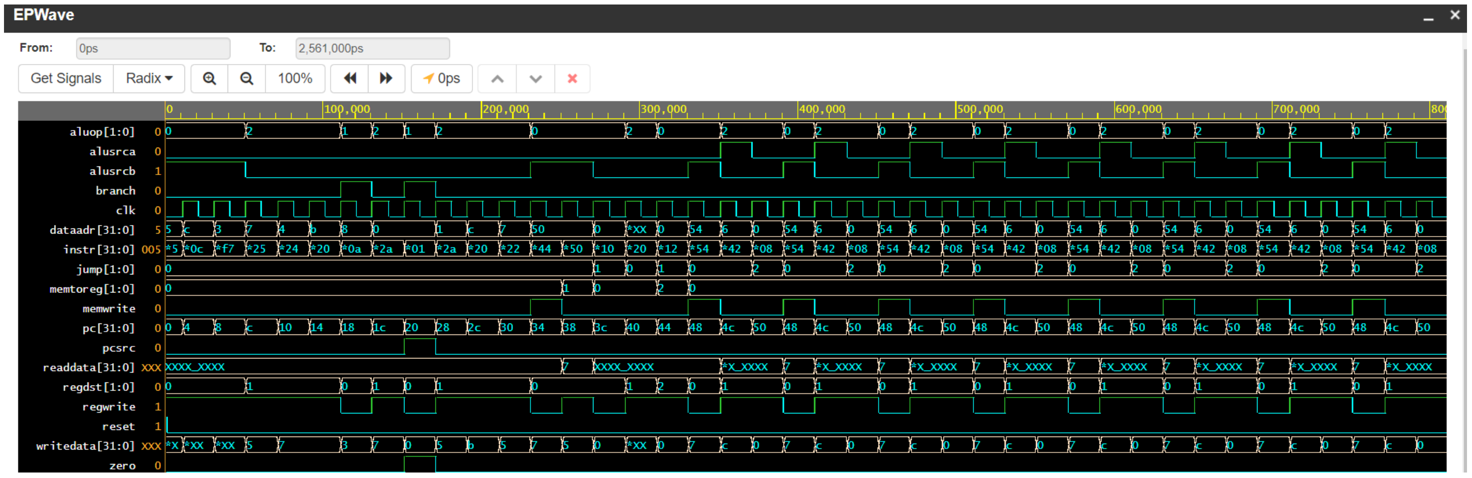
21802832

PART 1

a)

Location	**Machine Instruction (Hex)**	**Assembly Language Equivalent**
00	0x20020005	addi $v0, $zero, 5
04	0x2003000c	addi $v1, $zero, 12
08	0x2067fff7	addi $a3, $v1, -9
0c	0x00e22025	or $a0, $a3, $v0
10	0x00642824	and $a1, $v1, $a0
14	0x00a42820	add $a1, $a1, $a0
18	0x10a7000a	beq $a1, $a3, 10
1c	0x0064202a	slt $a0, $v1, $a0
20	0x10800001	beq $a0, $zero, 1
24	0x20050000	addi $a1, $zero, 0
28	0x00e2202a	slt $a0, $a3, $v0
2c	0x00853820	add $a3, $a0, $a1
30	0x00e23822	sub $a3, $a3, $v0
34	0xac670044	sw $a3, 68($v1)
38	0x8c020050	lw $v0, 80($zero)
3c	0x08000010	j 0x0000010
40	0x001f6020	add $t4, $zero, $ra
44	0x0c000012	jal 0x0000012
48	0xac020054	sw $v0, 84($zero)
4c	0x00039042	srl $s2, $v1, 1
50	0x03E00008	jr $ra

e)



f)

i) Writedata corresponds to data that will be written to data memory. It comes from register file and goes to data memory.

ii) Because early instructions are not R-type instructions. They are I-type instructions. Therefore, writedata is undefined.

iii) Readdata is used for only loading and storing operations (lw and sw), and their memtoreg value is 01. Therefore, readdata is undefined in most of the program.

iv) Data addr corresponds to ALU result in an R-type instruction. This result is written to the register file.

v) Dataaddr is undefined when $ra is in an arithmetic operation such as add, addi, sub, shift etc. Because $ra holds an address, and address exceeds the number range of arithmetic operations. Dataaddr becomes undefined.

g)

i) No, because we can implement srlv without any change. If we set ALUSrcA = 0, shift amount would come from register. Other parameters would be the same with srl instruction.

ii) I would modify ALU module. I would add a new case in ALU, and set result to b << a.

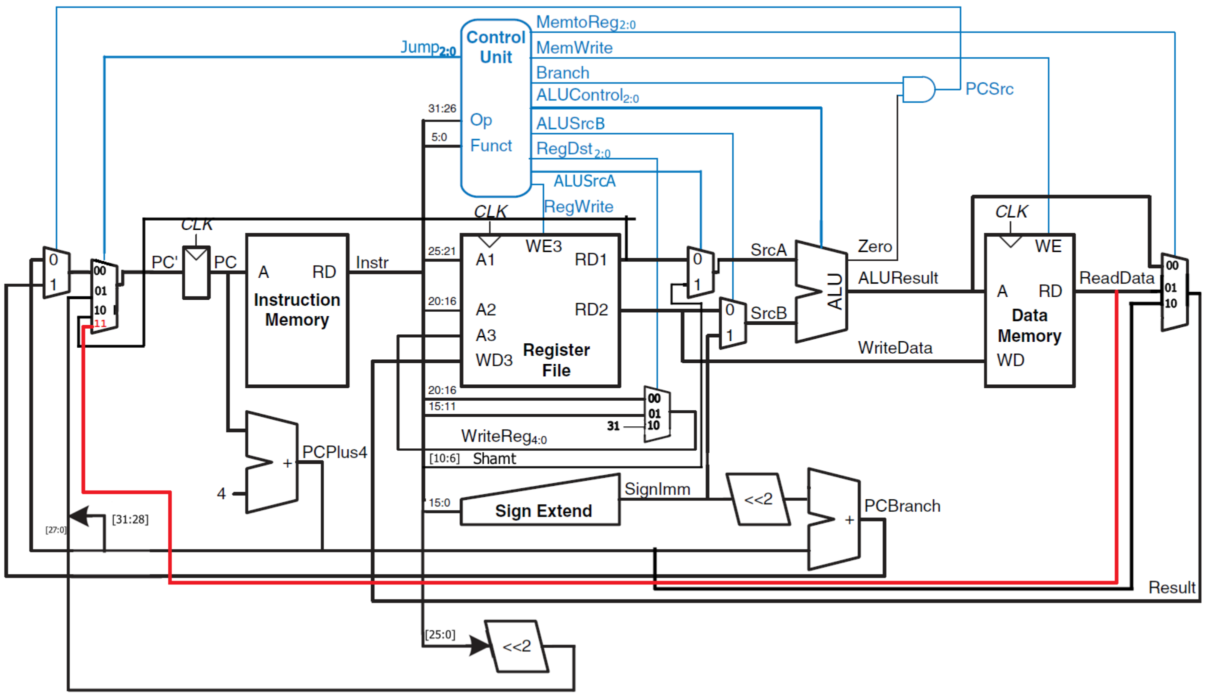
PART 2

a) IM[PC]

R[rt] $\leftarrow$ PC + 4

PC $\leftarrow$ M[R[rs] + SignExtImm]

b)



c)

Instruction	Opcode	RegWrite	RegDst	ALUSrcA	ALUSrcB	Branch	MemWrite	MemToReg	ALUOp	Jump
R-type	000000	1	01	0	0	0	0	00	10	00
srl	000000	1	01	1	0	0	0	00	10	00
lw	100011	1	00	0	1	0	0	01	00	00
sw	101011	0	X	0	1	0	1	XX	00	00
beq	000100	0	X	0	0	1	0	01	01	00
addi	001000	1	00	0	1	0	0	00	00	00
j	000010	0	X	X	X	X	0	XX	XX	01
jal	000011	1	10	X	X	X	0	10	XX	01
jr	000000	1	01	0	0	0	0	00	10	10
jalm	100111	1	00	0	1	0	0	10	00	11