CS224

Lab 4

Section 001

Metehan Saçakçı

21802788

Part 1:

a)

Location	Instruction (Hex)	Equivalent Assembly Language
00	0x2002 0005	addi $v0, $zero, 5
04	0x2003 000C	addi $v1, $zero, 12
08	0x2067 FFF7	addi $a3, $v1, -9
0C	0x00E2 2025	or $a0, $a3, $v0
10	0x0064 2824	and $a1, $v1, $a0
14	0x00A4 2820	add $a1, $a1, $a0
18	0x10A7 000A	beq $a1, $a3, 10
1C	0x0064 202A	slt $a0, $v1, $a0
20	0x1080 0001	beq $a0, $zero, 1
24	0x2005 0000	addi $a1, $zero, 0
28	0x00E2 202A	slt $a0, $a3, $v0
2C	0x0085 3820	add $a3, $a0, $a1
30	0x00E2 3822	sub $a3, $a3, $v0
34	0xAC67 0044	sw $a3, 68($v1)
38	0x8C02 0050	lw $v0, 80($zero)
3C	0x0800 0010	j 0x0000010
40	0x001F 6020	add $t4, $zero, $ra
44	0x0C00 0012	jal 0x0000012
48	0xAC02 0054	sw $v0, 84($zero)
4C	0x0003 9042	srl $s2, $v1, 1
50	0x03E0 0008	jr $ra

Figure 1.

e) Screenshot (Figure 2) of the generated waveform can be found below:

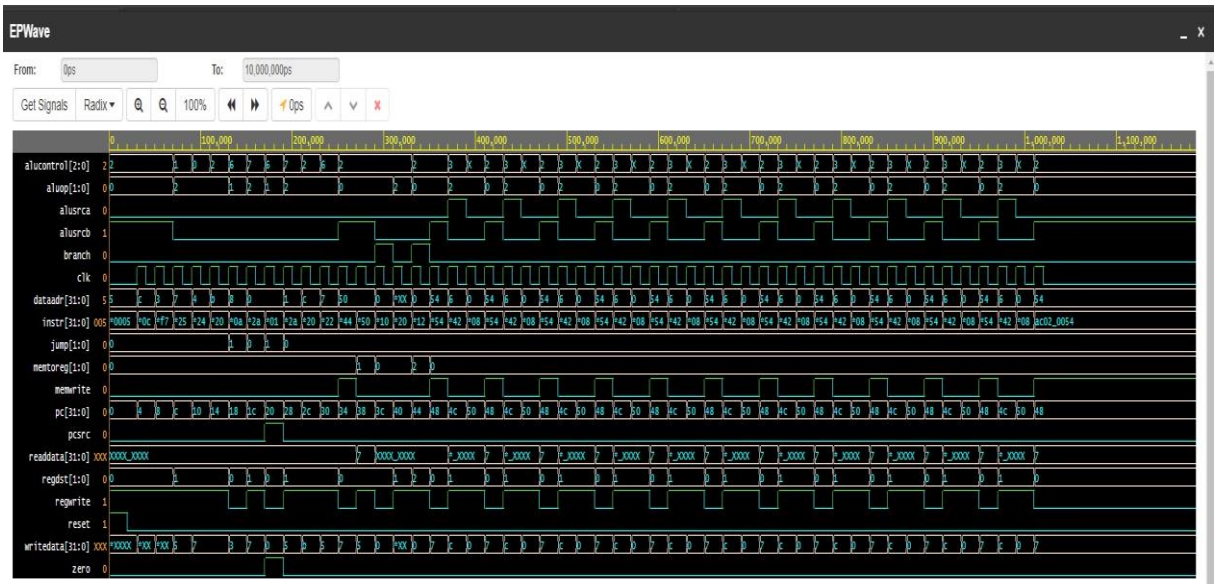

Figure 2.

f)

i) Firstly, it should be seen that “RD2” of the Register File is directly connected to the “Write Data” part of the Data Memory. Since, RD2 contains RF[RT] when a R-type instruction is processed, “Write Data” also will carry RF[RT] as well.

ii) When instruction memory is checked, it can be realized that first three instruction of the system are I-type instructions. In those cases, RF[RT] will be our destination and that is why, RF[RT] will be passed to A3 so “Write Data” which is connected with the RD2 will not be updated and also will be seen as undefined.

iii) Read Data can be defined as the data which is located on A of the data memory. However, A is also connected with ALU system which is used for many instructions. The result that comes from ALU cannot always be a valuable address which is located in the data memory. It is normal to see unknown result for “Read Data” when a data which is not created as address of data memory reached the A.

iv) According to the given SystemVerilog code of the Original12 and “mips” module implementation, dataadr corresponds to aluout which is operational for R-type instructions.

v) ALU unit designed to make operation according to given 3 bit alucont value. However, if an alucont value which is not correspond anything in the system is given, dataadr which is also the result of the aluout can be undefined.

g)

i) For srlv, the idea is making that: RF[rd] = RF[rs] >> RF[rt]. To make this I need a right logical shifter. If RD1 is connected to shifter as value which will be shifted and RD2 is also connected to the shifter as shift amount, the result can be passed to the WD3. Also because rd will be on the A3 and WE3 will be 1, operation will be completed. But, in the end, I believe I need new shift operation which can be done inside the ALU for this.

ii) For sll, the idea is making that: R[rd] = R[rt] << shamt. To make this ALU unit can be modified for accomplishing the left shifting. ALU unit designed to make operation according to given 3 bit alucont value. An unique 3 bit alucont value can be given for this purpose.

Part 2:

a) DM[RF[rs] + signExtImm] = RF[rt] (I-TYPE)
RF[rs] = RF[rs] + 4
PC = PC + 4

b)

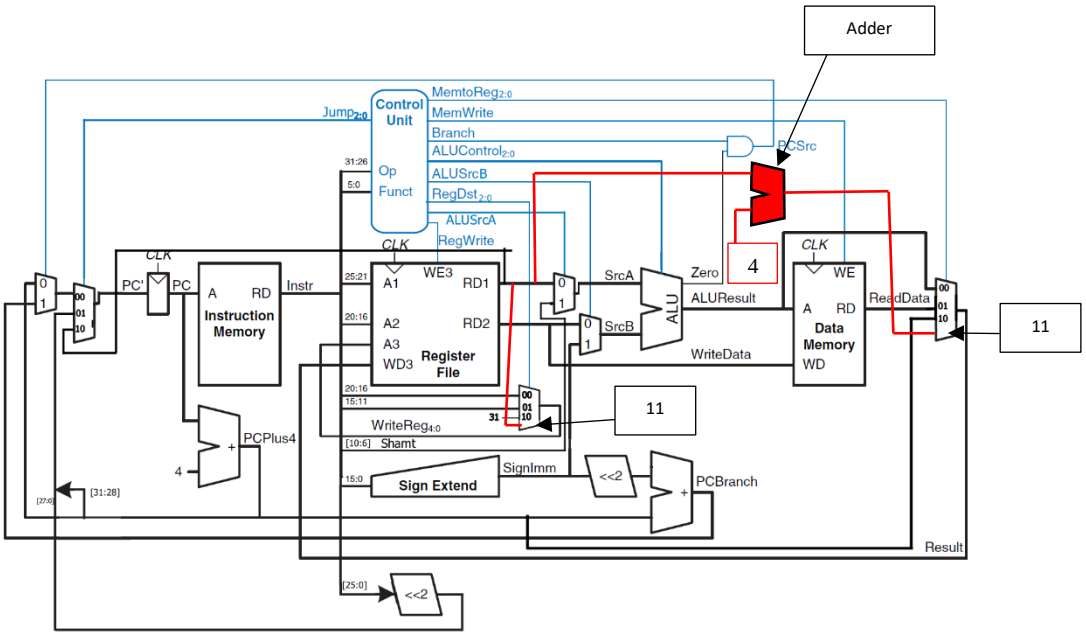

Figure 3.

c)

Insturction	Opcode	RegWrite	RegDst	ALUSrcA	ALUSrcB	Branch	MemWrite	MemToReg	ALU OP	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
sw+	000001	1	11	0	1	0	1	11	00	00

Figure 4.