CS224 Lab - 04

Preliminary Report

Mert Fidan - 22101734 - Section 3

- **PART I**

a-) Machine Instructions in imem module

Machine Code	PC Value	Instruction
0x2014fff6	0x00	addi $s4 $zero 0xFFF6
0x20090007	0x04	addi $t1 $zero 0x0007
0x22820003	0x08	addi $v0 $s4 0x0003
0x01342025	0x0c	or $a0 $t1 $s4
0x00822824	0x10	and $a1 $a0 $v0
0x00a42820	0x14	add $a1 $a1 $a0
0x1045003d	0x18	beq $v0 $a1 0x003D
0x0054202a	0x1c	slt $a0 $v0 $s4
0x10040001	0x20	beq $zero $a0 0x0001
0x00002820	0x24	add $a1 $zero $zero
0x0289202a	0x28	slt $a0 $s4 $t1
0x00853820	0x2c	add $a3 $a0 $a1
0x00e23822	0x30	sub $a3 $a3 $v0
0xac470057	0x34	sw $a3 0x0057 $v0

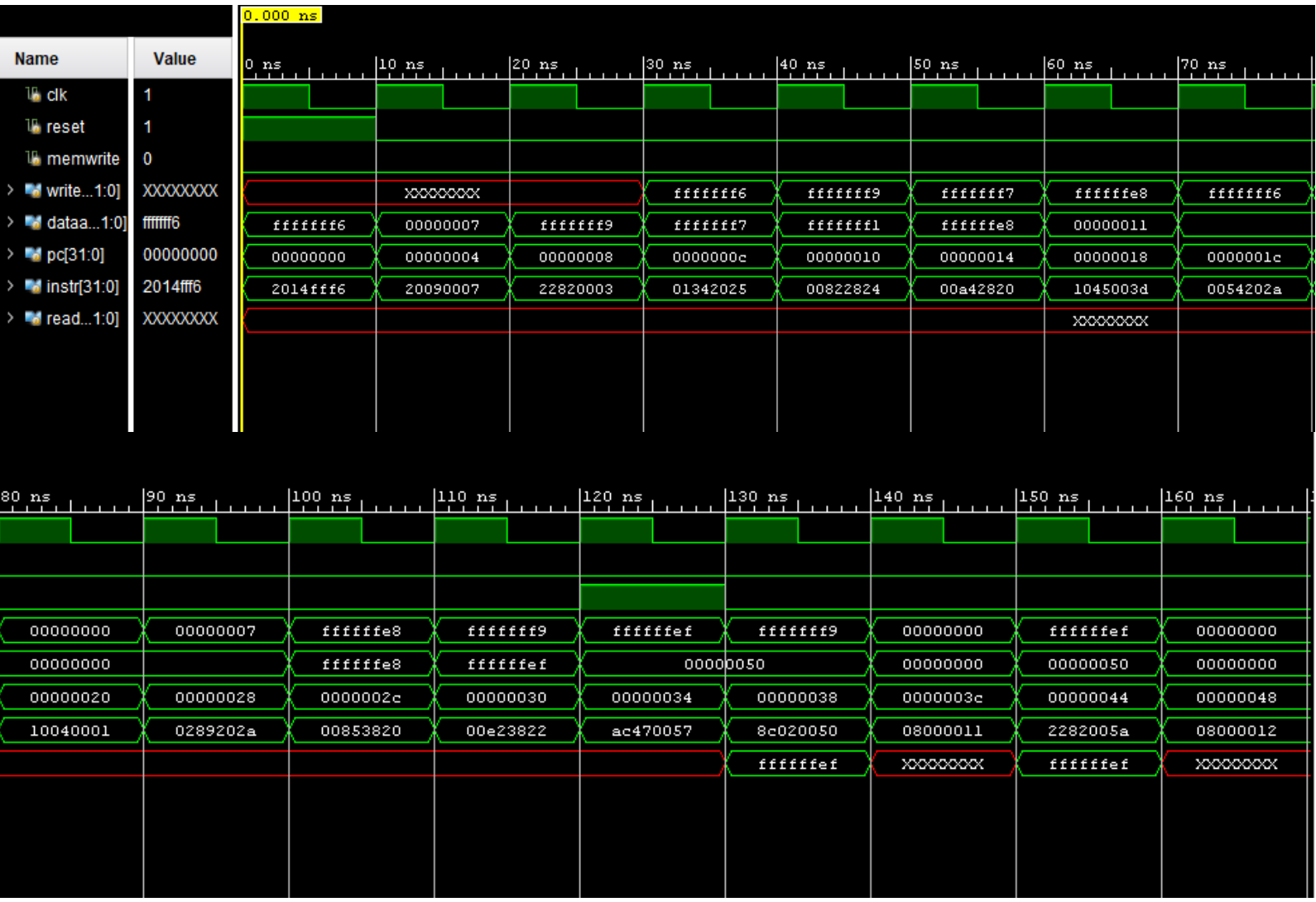
0x8c020050	0x38	**lw $v0 0x0050 $zero**
0x08000011	0x3c	**j 0x0000011**
0x20020001	0x40	**addi $v0 $zero 0x0001**
0x2282005a	0x44	**addi $v0 $s4 0x005A**
0x08000012	0x48	**j 0x0000012**

Instruction	Opcode	RegWrite	RegDst	ALUSrc	Branch	MemWrite	MemToReg	ALUOp	Jump
R-type	000000	1	1	0	0	0	0	10	0
lw	100011	1	0	1	0	0	1	00	0
sw	101011	0	X	1	0	1	X	00	0
beq	000100	0	X	0	1	0	X	01	0
addi	001000	1	0	1	0	0	0	00	0
j	000010	0	X	X	X	0	X	XX	1

Table 1: Main Decoder for Original10

ALUOp	Funct	ALUControl
00	X	010 (add)
01	X	110 (subtract)
1X	100000 (add)	010 (add)
1X	100010 (sub)	110 (subtract)
1X	100100 (and)	000 (and)
1X	100101 (or)	001 (or)
1X	101010 (slt)	111 (set less than)

Table 2: ALU Decoder for Original10



e-) Questions

i) In an R-type instruction what does writedata correspond to?

- It corresponds to *rt* field of the instruction

ii) Why is writedata undefined for some of the early instructions in the program?

- Since the first three instructions are I-type, *ALUSrc* is 1 and *SrcB* of ALU is chosen as *SignExtImm*.Thus, *writedata* receives no input for these instructions.

iii) In which instructions memwrite becomes 1?

- In *sw* instruction and new instruction *spc*

iv) Why is dataaddr 0xffffffe8 when PC is 0x14?

- Since *dataadr* corresponds to the ALU result and the result of addition for the values in $a1 and $a0 is *0xffffffe8*, *dataadr* becomes *0xffffffe8* for the instruction in *PC = 0x14*, which is *add $a1 $a1 $a0*

v) In the example program, when is the output of readdata defined and why?

- It is defined for the instructions that read data from the data memory, in this case for the occurrences of *lw* instruction. Since the first *lw* appears at *PC = 0x38*, *readdata* is firstly defined at that instance.

f-) Modified ALU Module

```
module alu(input  logic [31:0] a, b,
           input  logic [4:0] shamt,
           input  logic [2:0]  alucont,
           output logic [31:0] result,
           output logic zero);

    always_comb
        case(alucont)
            3'b010: result = a + b;
            3'b011: result = a ^ b; // a XOR b
            3'b100: result = (a << shamt) | (a >> (32 - shamt)); // for ROL
            3'b110: result = a - b;
            3'b000: result = a & b;
            3'b001: result = a | b;
            3'b111: result = (a < b) ? 1 : 0;
            default: result = {32{1'bx}};
        endcase

    assign zero = (result == 0) ? 1'b1 : 1'b0;
endmodule
```

- **PART II**

a-) RTL Expressions for *spc* and *rol*

IM[PC]

rol: RF[rd] ← [(RF[rs] << shamt) OR (RF[rs] >> 32 - shamt)]

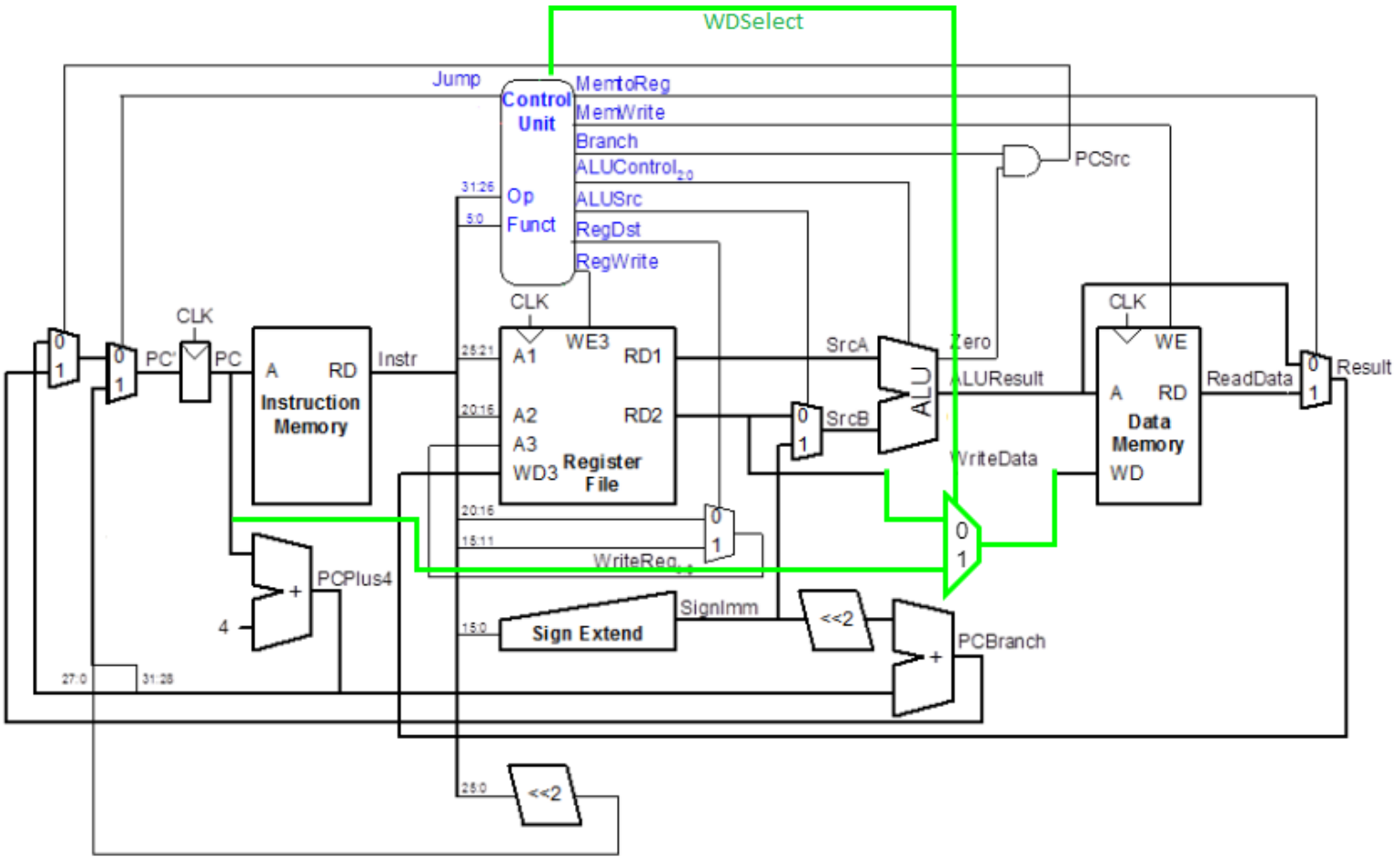
PC ← PC + 4

IM[PC]

spc: M[RF[rs] + SignExtImm] ← PC

PC ← PC + 4

b-) Datapath Changes



c-) Main Decoder and ALU Decoder Changes

Instruction	Opcode	RegWrite	RegDst	ALUSrc	Branch	MemWrite	MemToReg	ALUOp	Jump
R-type	000000	1	1	0	0	0	0	10	0
lw	100011	1	0	1	0	0	1	00	0
sw	101011	0	X	1	0	1	X	00	0
beq	000100	0	X	0	1	0	X	00	0
addi	001000	1	0	1	0	0	0	01	0
j	000010	0	X	X	0	X	X	XX	1
spc	000001	0	X	1	0	1	X	11	0

ALUOp	Funct	ALUControl	WDSelect
00	X	010 (add)	0
01	X	110 (subtract)	0
10	100000 (add)	010 (add)	0
10	100010 (sub)	110 (subtract)	0
10	100100 (and)	000 (and)	0
10	100101 (or)	001 (or)	0
10	101010 (slt)	111 (set less than)	0
10	000000(rol)	100(rol)	0
11	X	010(add for spc)	1