CS224
Section No.: 2
Spring 2019
Lab No. 4
Munib Emre Sevilgen / 21602416

Program List:
#f – The MIPS test program to test the new instruction in MIPS assembly language
#g – The modified maindec module of Complete MIPS Model to add the new instructions

#b

Location	Machine Instruction (in hex)	Assembly Language Equivalent
0x00000000	0x20020005	addi $v0, $zero, 5
0x00000004	0x2003000c	addi $v1, $zero, 12
0x00000008	0x2067fff7	addi $a3, $v1, -9
0x0000000c	0x00e22025	or $a0, $a3, $v0
0x00000010	0x00642824	and $a1, $v1, $a0
0x00000014	0x00a42820	add $a1, $a1, $a0
0x00000018	0x10a7000a	beq $a1, $a3, 0x0000000a
0x0000001c	0x0064202a	slt $a0, $v1, $a0
0x00000020	0x10800001	beq $a0, $zero, 0x00000001
0x00000024	0x20050000	addi $a1, $zero, 0
0x00000028	0x00e2202a	slt $a0, $a3, $v0
0x0000002c	0x00853820	add $a3, $a0, $a1
0x00000030	0x00e23822	sub $a3, $a3, $v0
0x00000034	0xac670044	sw $a3, 68($v1)
0x00000038	0x8c020050	lw $v0, 80($zero)
0x0000003c	0x08000011	j 0x00000044
0x00000040	0x20020001	addi $v0, $zero, 1
0x00000044	0xac020054	sw $v0, 84($zero)
0x00000048	0x08000012	j 0x00000048

#c

nop:

IM[PC]

PC ← PC + 4

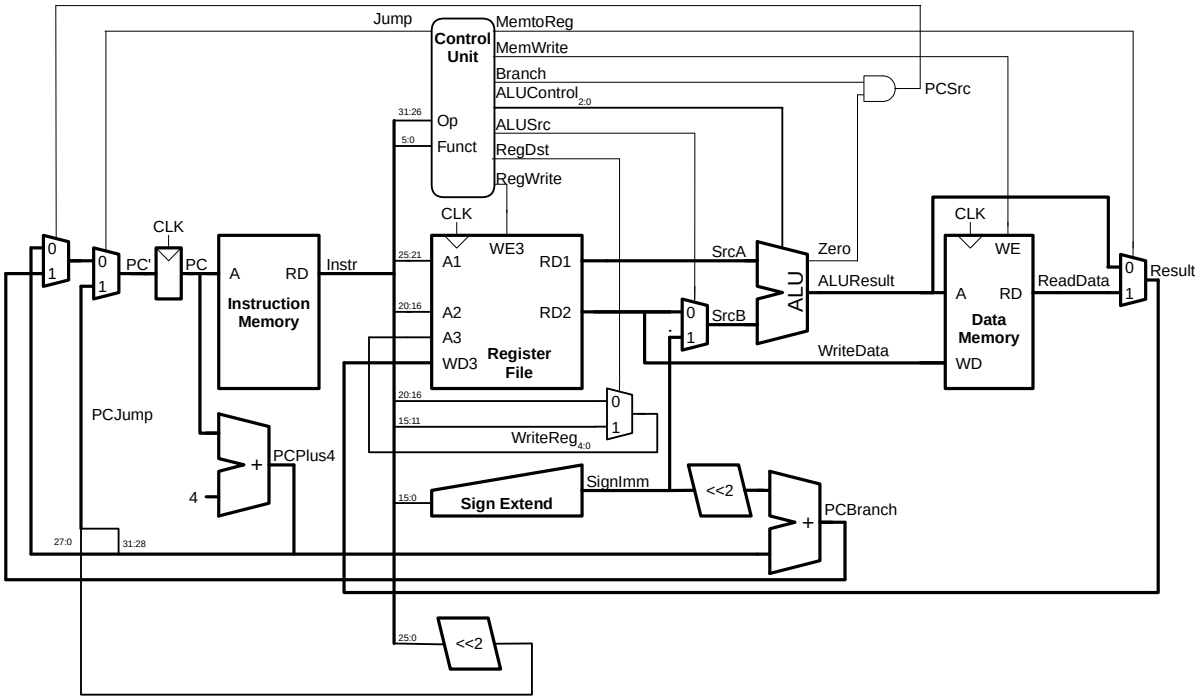
subi:

IM[PC]

R[rt] ← R[rs] - SignExtImm

PC ← PC + 4

#d



Since the instructions nop and subi is possible without changing or adding anything to the datapath, the base datapath is placed directly.

#e

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
nop	000001	0	X	X	0	0	X	XX	0
subi	010000	1	0	1	0	0	0	01	0

#f

```
        .text
main:
        # Test for new instructions
        nop

        addi $t0, $zero, 3
        addi $t1, $zero, 3
        addi $t0, $t0, -2
        subi $t1, $t1, 2

        bne $t0, $t1, subiError

        addi $t2, $zero, 1

        j continue

        subiError:
                move $t2, $zero
continue:
        # Test for other instruction that is
        # taken from the given system verilog file
        addi $v0, $zero, 5
        addi $v1, $zero, 12
        addi $a3, $v1, -9
        or $a0, $a3, $v0
        and $a1, $v1, $a0
        add $a1, $a1, $a0
        beq $a1, $a3, c
        slt $a0, $v1, $a0
        beq $a0, $zero, a
        addi $a1, $zero, 0
        a:
        slt $a0, $a3, $v0
        add $a3, $a0, $a1
        sub $a3, $a3, $v0
        sw $a3, 68($v1)
        lw $v0, 80($zero)
        j c
        addi $v0, $zero, 1
        c:
        sw $v0, 84($zero)
        d:
        j d
```

#g

Only the maindec module should be changed in order to make the nop and subi part of the single cycle MIPS processor's instruction set.

```
module maindec (input logic[5:0] op,
                output logic memtoreg, memwrite, branch,
                output logic alusrc, regdst, regwrite, jump,
                output logic[1:0] aluop );
  logic [8:0] controls;

  assign {regwrite, regdst, alusrc, branch, memwrite,
          memtoreg,  aluop, jump} = controls;

 always_comb
  case(op)
    6'b000000: controls <= 9'b110000100; // R-type
    6'b100011: controls <= 9'b101001000; // LW
    6'b101011: controls <= 9'b001010000; // SW
    6'b000100: controls <= 9'b000100010; // BEQx
    6'b001000: controls <= 9'b101000000; // ADDI
    6'b000010: controls <= 9'b000000001; // J
    6'b000001: controls <= 9'b000000000; // NOP
    6'b010000: controls <= 9'b101000010; // SUBI
    default:   controls <= 9'bxxxxxxxxx; // illegal op
  endcase
endmodule
```