Cs224 Section01
Lab04
Osman Buğra Aydın
21704100
Spring 2020

B-) Instructions

Hex Location	Machine Instruction	Assembly Equivalent
00	0x h20020005	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	and $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	0x h08000011	j 0x0000011
40	0x20020001	addi $v0, $zero, 1
44	0xac020054	sw $v0, 84($0)
48	0x08000012	j 0x0000012

C-) Rtl Expressions

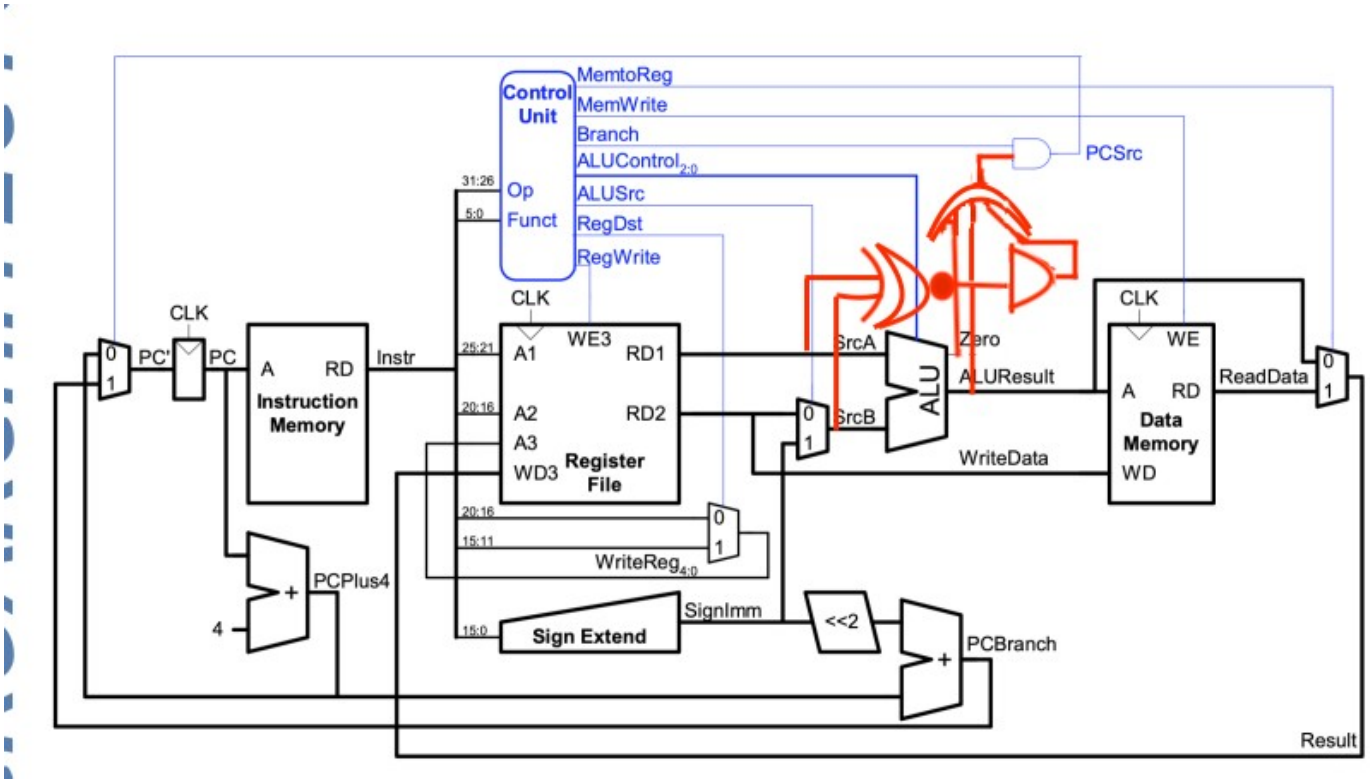
Expressions for ble

```
IM[PC]
if( RF[rs] == RF[rt])
      PC <-- 4 + SignExt(immed) << 2
else if ( RF[rt] < RF[rs] )
      PC <-- 4 + SignExt(immed) << 2
else
      PC <-- PC + 4
```

Expressions for subi

```
IM[PC]
RF[ rt ] <-- RF[ rs ] - SignExt(immed)
PC <-- PC + 4
```

D-) Datapath


Control Unit
MemtoReg
MemWrite
Branch
ALUControl2:0
ALUSrc
RegDst
RegWrite
Op
Funct
31:26
5:0
PCSrc
CLK
PC'
PC
A
RD
Instruction Memory
Instr
25:21
20:16
A1
A2
A3
WD3
WE3
RD1
RD2
Register File
SrcA
SrcB
ALU
Zero
ALUResult
WriteData
WE
Data Memory
WD
ReadData
20:16
15:11
WriteReg4:0
PCPlus4
4
15:0
Sign Extend
SignImm
<<2
PCBranch
Result

E-) Control Signals

Instructions	RegWrite	Reg Dst	AluSrc	Branch	Mem Write	MemReg	AluOp 2:0	Jump
lw	1	0	1	0	0	1	0 1 0	0
sw	0	X	1	0	1	X	0 1 0	0
beq	0	X	0	1	0	X	1 1 0	0
add	1	1	0	0	0	0	0 1 0	0
sub	1	1	0	0	0	0	1 1 0	0
or	1	1	0	0	0	0	0 0 1	0
slt	1	1	0	0	0	0	1 1 1	0
addi	1	0	1	0	0	0	0 1 0	0
j	0	X	X	0	0	X	X	1
and	1	1	0	0	0	0	0 0 0	0
ble	0	X	0	1	0	0	1 1 1	0
subi	1	0	1	0	0	0	1 1 0	0

F-) Mips Code

```
	.text

	.text

#------------------------------Code For Ble Instruction--------------------------------

addi	$t0, $zero, 45 #first variable
addi	$t1, $zero, 117 #second variable

if:	# if( variable1 < variable2 ) branch done

	slt	$t2, $t0, $1

	beq	$t2, 0, done

elseIf: # if( variable1 == variable2 ) branch done

	beq	$t0, $t1, done

	j	countinueCode

done:
	addi $v0, $zero, 1
	addi $a0, $zero, 1
	syscall
```

countinueCode: #Your program countinues here. It writes 1 to show it works correctly.

#------------------------------End For Ble Instruction--------------------------------

#------------------------------Code For Subi Instruction--------------------------------

```
addi	$t0, $zero, 42 #first variable
addi	$t1, $zero, 5 #second variable

sub	$t3, $t0, $t1 #subtraction part


check:	add $a0, $zero, $t3 #It writes the result to show it works correctly.
	syscall
```

#------------------------------Code For Subi Instruction--------------------------------