Group No. 76
Apoorv Kumar - 21CS10008
Swarnabh Mandal - 21CS10068

Instruction Format Encoding

1. R-type (register) instruction encoding

opcode	rs	rt	rd	shamt	func
5 bits	5 bits	5 bits	5 bits	8 bits (no bits are in use)	4 bits

   - First 5 bits will be reserved for the opcode
   - Next 5 bits will be reserved for the source register 1 (rs)
   - Next 5 bits will be reserved for the source register 2 (rt)
   - Next 5 bits will be reserved for the destination register (rd)
   - Next 8 bits are redundant
   - Next 4 bits will be reserved for the functional opcode

2. I-type (immediate) instruction encoding

opcode	rs	rt	immediate
5 bits	5 bits	5 bits	17 bits (only lower order 16 bits are used)

   - First 5 bits will be reserved for the opcode
   - Next 5 bits will be reserved for the source register 1 (rs)
   - Next 5 bits will be reserved for the destination register (rt)
   - Next 1 bit will be redundant
   - Next 16 bits will be reserved for the immediate value (which will be eventually sign-extended to 32 bits)

3. J-type (jump) instruction encoding

opcode	jump address
5 bits	27 bits (only lower 26 bits are used)

   - First 5 bits are reserved for the opcode
   - Next 1 bit is redundant

- Next 26 bits are reserved for the jump address (which will be left shifted by 2 bits)

Operations and their corresponding Opcodes and Functional Opcodes

Operation	OPCODE (in decimal)	FUNC (in decimal)
ADD	0	0
SUB	0	1
AND	0	2
OR	0	3
XOR	0	4
NOT	0	5
SLA	0	6
SRA	0	7
SRL	0	8
ADDI	1	NA
SUBI	2	NA
ANDI	3	NA
ORI	4	NA
XORI	5	NA
NOTI	6	NA
SLAI	7	NA
SRAI	8	NA
SRLI	9	NA
LD	10	NA

ST	11	NA
LDSP	12	NA
STSP	13	NA
BR	14	NA
BMI	15	NA
BPL	16	NA
BZ	17	NA
PUSH	18	NA
POP	19	NA
CALL	20	NA
RET	21	NA
MOVE	22	NA
HALT	23	NA
NOP	24	NA

ALU Control

The Control Unit will send the required ALUOp signals to the ALU Control, based on the instruction's opcode.

The ALU Control sends the corresponding operation signal to the ALU, based on the received ALUOp signal and the functional opcode (if applicable).

Opcode and corresponding ALUOp

Opcode (in decimal)	ALUOp (in decimal)
0	0
1	1
2	2
3	3
4	4

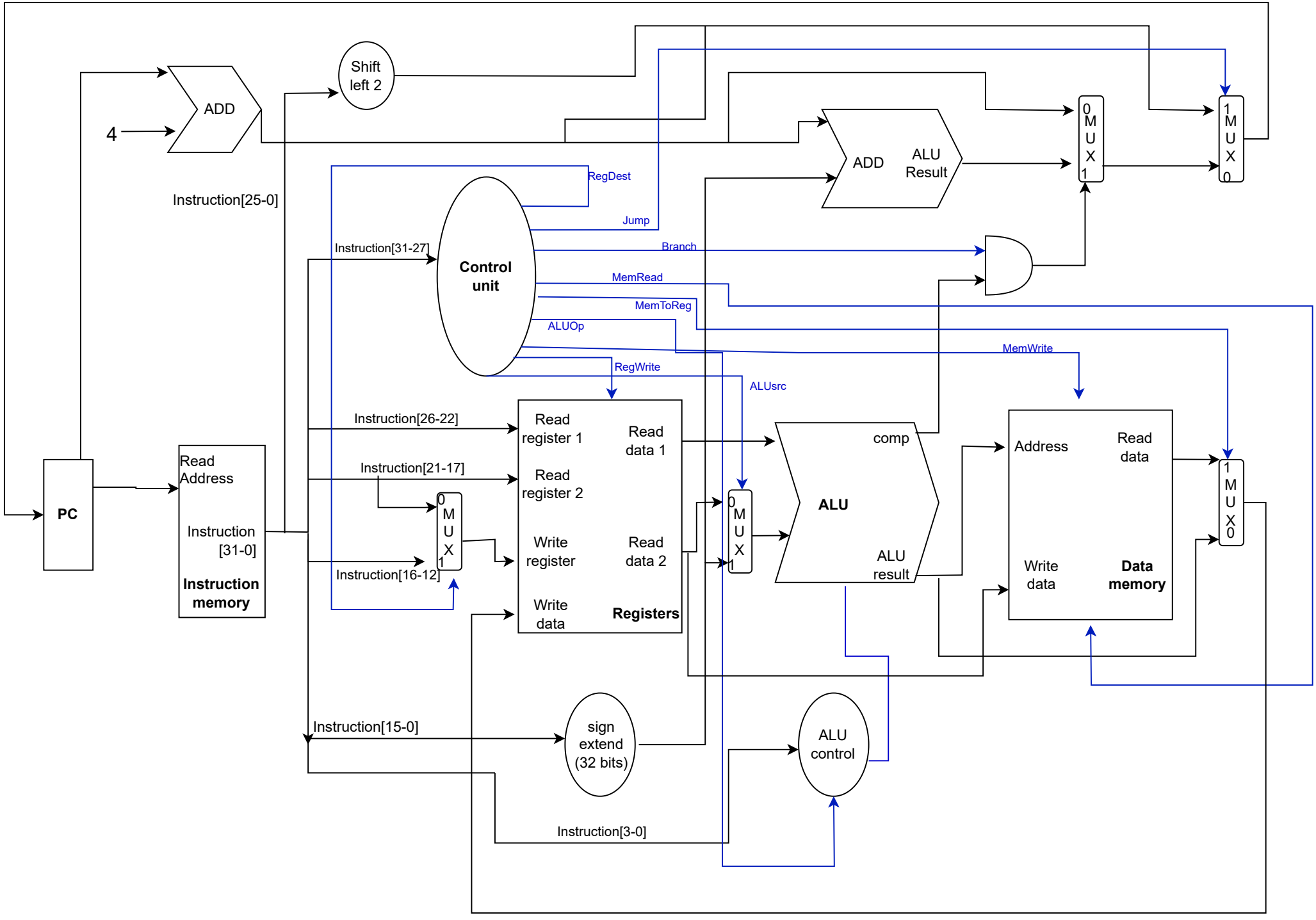
5	5
6	6
7	7
8	8
9	9
10	1
11	1
12	1
13	1
14	10
15	11
16	12
17	13
18	1
19	1
20	1
21	1
22	1
23	1
24	1

ALUOp, Functional Opcode and corresponding ALUfunc signal

ALUOp (in decimal)	func (in decimal)	ALUfunc
0	0	add
0	1	sub
0	2	and
0	3	or

0	4	xor
0	5	not
0	6	sla
0	7	sra
0	8	srl
1	X	add
2	X	sub
3	X	and
4	X	or
5	X	xor
6	X	not
7	X	sla
8	X	sra
9	X	srl
10	X	set ‘comp’ to 1
11	X	set ‘comp’ to 1 if operand1<0, else 0
12	X	set ‘comp’ to 1 if operand1>0, else 0
13	X	set ‘comp’ to 1 if operand1=0, else 0

*ALU also has a comparator, that performs the required comparison (w.r.t. zero) and sets the ‘comp’ value accordingly


ADD
4
Shift
left 2
Instruction[25-0]
ADD
ALU
Result
0
M
U
X
1
1
M
U
X
0
RegDest
Jump
Branch
MemRead
MemToReg
ALUOp
MemWrite
RegWrite
ALUsrc
Instruction[31-27]
Control
unit
PC
Read
Address
Instruction
[31-0]
Instruction
memory
Instruction[26-22]
Instruction[21-17]
Instruction[16-12]
Read
register 1
Read
register 2
Write
register
Write
data
Read
data 1
Read
data 2
Registers
comp
ALU
ALU
result
Address
Read
data
Write
data
Data
memory
Instruction[15-0]
sign
extend
(32 bits)
ALU
control
Instruction[3-0]

Control Signals for some Instructions

- ADD R1, R2, R3
    - ReadIM (Read the instruction)
    - RegDest = 1 (Destination register is given as ‘rd’=R3 in the instruction)
    - ReadData1 (Read the value of ‘rs’ = R1)
    - ReadData2 (Read the value of ‘rt’ = R2)
    - ALUOp = 0000 (This signal, along with ‘func’, in ALUControl will send the ‘add’ signal to the ALU)
    - RegWrite (Write the result value in the destination register)
    - LoadPC (Load the next instruction)

- ADDI R3, #25
    - ReadIM (Read the instruction)
    - RegDest = 0 (Destination register is given as ‘rt’=R3 in the instruction)
    - ReadData1 (Read the value of ‘rs’ = R3)
    - ALUsrc = 1 (The second operand for the ALU is the immediate value, obtained from the instruction)
    - ALUOp = 0001 (This signal in ALUControl will send the ‘add’ signal to the ALU)
    - RegWrite (Write the result value in the destination register)
    - LoadPC (Load the next instruction)

- LD R2, 10(R6)
    - ReadIM (Read the instruction)
    - RegDest = 0 (Destination register is given as ‘rt’ = R2 in the instruction)
    - ReadData1 (Read the value of ‘rs’ = R6)
    - ALUsrc = 1 (The second operand for the ALU is the immediate value, obtained from the instruction)

    - ALUOp = 0001 (This signal in ALUControl will send the ‘add’ signal to the ALU)
    - MemRead = 1 (Reads the value in the memory, at the address whose value is calculated in the ALU, i.e. R+10)
    - MemToReg = 1 (Sends the value read from the memory to the register-write port)
    - RegWrite (Write the required value in the destination register, i.e. Mem[R6+10])
    - LoadPC (Load the next instruction)

- ST R2, -2(R11)
    - ReadIm (Read the instruction)
    - ReadData1 (Read the value of ‘rs’ = R11)
    - ReadData2 (Read the value of ‘rt’ = R2)
    - ALUsrc = 1 (The second operand for the ALU is the immediate value, obtained from the instruction)
    - ALUOp = 0001 (This signal in ALUControl will send the ‘add’ signal to the ALU)
    - MemWrite (Writes the value read from ReadData2 signal in the memory, at the address value calculated in the ALU)
    - LoadPC (Load the next instruction)

- BZ R8, #-75
    - ReadIm (Read the instruction)
    - ReadData1 (Read the value of ‘rs’ = R8)
    - ALUOp = 1101 (This signal in ALUControl will send the ‘compare equal to zero’ signal to the ALU)
    - Branch = 1 (This will choose the new value of next PC, by adding the offset, i.e. the immediate value in the instruction, if the required condition is satisfied)
    - LoadPC (Load the next instruction)

- PUSH R6
    - ReadIm (Read the instruction)
    - RegDest = 0 (Destination register is given as ‘rt’ = SP, since this instruction’s encoding will have SP as the ‘rt’ register)
    - ReadData1 (Read the value of SP)
    - ReadData2 (Read the value of R6)
    - ALUsrc = 1 (The second operand for the ALU is the immediate value, obtained from the instruction, which will be -4 in this case)

    - ALUOp = 001 (This signal in ALUControl will send the ‘add’ signal to the ALU)
    - MemWrite (Writes the value read from ReadData2 signal in the memory, at the address value calculated in the ALU)
    - RegWrite (Write the result value, i.e. SP-4, in the destination register)
    - LoadPC (Load the next instruction)

- MOVE R10, R5
    - ReadIm (Read the instruction)
    - RegDest = 0 (Destination register is given as ‘rt’ = R10)
    - ReadData1 (Read the value of ‘rs’ = R5)
    - ALUsrc = 1 (The second operand for the ALU is the immediate value, obtained from the instruction, which will be 0 in this case)
    - ALUOp = 001 (This signal in ALUControl will send the ‘add’ signal to the ALU)
    - RegWrite (Write the result value, i.e. R5+0 = R5, in the destination register)
    - LoadPC (Load the next instruction)