Computer Organization and Architecture Lab

Group Number: 65
Pranav Mehrotra 20CS10085
Saransh Sharma 20CS30065

1. Instruction Set Architecture

Class	Instruction	Usage	Meaning
Arithmetic	Add	add rs,rt	$rs \leftarrow (rs) + (rt)$
	Comp	comp rs,rt	$rs \leftarrow$ 2's Complement (rs)
	Add immediate	addi rs,imm	$rs \leftarrow (rs) + imm$
	Complement Immediate	compi rs,imm	$rs \leftarrow$ 2's Complement (imm)
Logic	AND	and rs,rt	$rs \leftarrow (rs) \wedge (rt)$
	XOR	xor rs,rt	$rs \leftarrow (rs) \oplus (rt)$
Shift	Shift left logical	shll rs, sh	$rs \leftarrow (rs)$ left-shifted by sh
	Shift right logical	shrl rs, sh	$rs \leftarrow (rs)$ right-shifted by sh
	Shift left logical variable	shllv rs, rt	$rs \leftarrow (rs)$ left-shifted by (rt)
	Shift right logical	shrl rs, rt	$rs \leftarrow (rs)$ right-shifted by (rt)
	Shift right arithmetic	shra rs, sh	$rs \leftarrow (rs)$ arithmetic right-shifted by sh
	Shift right arithmetic variable	shrav rs, rt	$rs \leftarrow (rs)$ right-shifted by (rt)
Memory	Load Word	lw rt,imm(rs)	$rt \leftarrow mem[(rs) + imm]$
	Store Word	sw rt,imm,(rs)	$mem[(rs) + imm] \leftarrow (rt)$
Branch	Unconditional branch	b L	goto L
	Branch Register	br rs	goto (rs)
	Branch on less than 0	bltz rs,L	if$(rs) < 0$ then goto L
	Branch on flag zero	bz rs,L	if $(rs) = 0$ then goto L
	Branch on flag not zero	bnz rs,L	if$(rs) \neq 0$ then goto L
	Branch and link	bl L	goto L; $31 \leftarrow$ (PC)+4
	Branch on Carry	bcy L	goto L if Carry $= 1$
	Branch on No Carry	bncy L	goto L if Carry $= 0$
Complex	Diff	diff rs, rt	$rs \leftarrow$ the LSB bit at which rs and rt differ

2. Instruction Format

1. R-Format

Opcode 6 Bits	rs 5 bits	rt 5 bits	Don't Care 5 bits	shamt 5 bits	Func code 6 bits

2. I-Format

Opcode 6 bits	rs 5 bits	Don't Care 5 bits	Immediate 16 bits

lw rt,imm(rs) rt ← mem[(rs) + imm]

Opcode 6 bits	rs 5 bits	rt 5 bits	Immediate 16 bits

sw rt,imm(rs) mem[(rs) + imm] ← (rt)

Opcode 6 bits	rs 5 bits	rt 5 bits	Immediate 16 bits

3. Branch

Opcode	Label

3. Instruction format summary:

Operation	OPCODE(Decimal)	Function Code(Decimal)	Format
Add	000000	000000	R
Compliment	000000	001000	R
Add Immediate	000010	X	I
Complement Immediate	000011	X	I
AND	000000	010000	R
XOR	000000	011000	R
SLL	000001	100000	R
SLL Variable	000000	100000	R
SRL	000001	101000	R
SRL Variable	000000	101000	R
SRA	000001	110000	R

SRA Variable	000000	110000	R
Load Word	000110	X	I
Store Word	010110	X	I
Unconditional branch	100001	X	branch
Branch Register	100010	X	R
Branch Less Than 0	100011	X	I
Branch on flag zero	100100	X	I
Branch on flag not zero	100101	X	I
Branch and link	110001	X	branch
Branch on Carry	100110	X	branch
Branch on No Carry	100111	X	branch
Diff	000000	111000	R

4. Control Signals

Aluop [2:0] : To determine which operation would be performed by ALU

Operation	Alu Op CODE(Binary)
Add	000
Compliment	001
AND	010
XOR	011
SLL	100
SRL	101
SRA	110
Diff	111

Alusource [1:0]: To decide the second operand to pass to Alu out of rt, shamt and offset.

00 : pass rt to ALU

01: pass shamt to ALU

10: pass offset to ALU

Reg_dest [1:0]: To decide which register would store value after ALU operation

00: first argument i.e. rs

01: second argument i.e. rt

10: register number 31 for branch and link

Memtoreg [1:0]: To decide what to write in destination register

00: to store the value of ALU

01: to store the value extracted from memory

10: to store PC+4 in register number 31 during branch and link

Mem_write: To decide whether to perform write operation on memory

0: no write operation on data memory

1: perform write operation on data memory

Branch [2:0] : To decide the type of branching

Reg_write: To decide whether to write on register or not.

5. Instruction decode logic:

Operation	Opcode	Funccode	Aluop	Alusource	Reg_dest	Memtoreg	Mem_write	Branch	Reg_write
Add	000000	000000	000	00	00	00	0	000	0
Compliment	000000	001000	001	00	00	00	0	000	0
Add Immediate	000010	X	000	10	00	00	0	000	0
Complement Immediate	000011	X	001	10	00	00	0	000	0
AND	000000	010000	010	00	00	00	0	000	0
XOR	000000	011000	011	00	00	00	0	000	0

SLL	**000001**	**100000**	**100**	**01**	**00**	**00**	**0**	**000**	**0**
SLL Variable	**000000**	**100000**	**100**	**00**	**00**	**00**	**0**	**000**	**0**
SRL	**000001**	**101000**	**101**	**01**	**00**	**00**	**0**	**000**	**0**
SRL Variable	**000000**	**101000**	**101**	**00**	**00**	**00**	**0**	**000**	**0**
SRA	**000001**	**110000**	**110**	**01**	**00**	**00**	**0**	**000**	**0**
SRA Variable	**000000**	**110000**	**110**	**00**	**00**	**00**	**0**	**000**	**0**
Load Word	**000110**	**X**	**000**	**10**	**00**	**01**	**0**	**000**	**0**
Store Word	**010110**	**X**	**000**	**10**	**01**	**01**	**1**	**000**	**1**
Unconditional branch	**100001**	**X**	**000**	**00**	**00**	**00**	**0**	**001**	**1**
Branch Register	**100010**	**X**	**000**	**00**	**00**	**00**	**0**	**010**	**1**
Branch Less Than 0	**100011**	**X**	**000**	**00**	**00**	**00**	**0**	**011**	**1**
Branch on flag zero	**100100**	**X**	**000**	**00**	**00**	**00**	**0**	**100**	**1**
Branch on flag not zero	**100101**	**X**	**000**	**00**	**00**	**00**	**0**	**101**	**1**
Branch and link	**110001**	**X**	**000**	**00**	**10**	**10**	**0**	**001**	**0**
Branch on Carry	**100110**	**X**	**000**	**00**	**00**	**00**	**0**	**110**	**1**
Branch on No Carry	**100111**	**X**	**000**	**00**	**00**	**00**	**0**	**111**	**1**
Diff	**000000**	**111000**	**111**	**00**	**00**	**00**	**0**	**000**	**1**

Op code:

- First bit - 0
- Second bit - Mem_write
- (3-4th bit) - Reg_dest and Mem_to_Reg (Both are equal at all times)
- (5-6 th bit) - Alu Source
- If first bit - 1:
    1. 2nd & 3rd bit: Reg_dest and mem_to_reg
    2. Last three bits branch instructions.
    3. reg_write : 1 (Except Branch and link)
    4. mem_write : 0
    5. Alu op: 000
    6. Alu source: 00

Func Code:

- First 3 bits - ALU Op code
- Last 3 bits - Branch code

Add imm, Complement imm, Load Word and Store Word are special cases.