North South University

Department of Electrical & Computer Engineering

CSE332

Computer Organization and Architecture

Instruction Set Architecture Design

Submitted by:

Name: Ferdous Reza Niloy
ID: 2021281642

Submitted to: Tanjila Farah (TnF)

Objectives: My objective was to design a 13 Bit ISA which can solve a particular problems like simple arithmetic & logic operations, branching and loops.

Operands: My goal is to use accumulator base ISA. For this reason, I am going to take two operands. I will address these two operands as **d**, **t** and **s**.

Types of Operands: I need register operands to implement arithmetic instructions, and memory operands to implement data transfer instructions from memory to register. As a result, I'll need two sorts of operands.

- Register based.
- Memory based.

Operations: I will allocate 4 bits opcode, so the executable instructions number will be 2^4 or 16.

Types of operations: In my design there will be five different types of operation. The operations are:

- Arithmetic
- Logical
- Data Transfer
- Conditional Branch
- Unconditional Jump

Formats:

I use three types of formats for my ISA. They are:

- Register Type – R type
- Immediate Type – I type
- Jump Type - J Type

Category	Operation	Name	Type	OpCode	Syntax	Comments
	No operation	nop		0000	nop	
Data transfer	Load word	lw	I	0001	lw r0 r1 2	r1 = Mem[r0+2]
Arithmetic	Add number with an immediate	addi	I	0010	addi r1 r2 5	r2 = r1+5
Data transfer	Store word	sw	I	0011	sw r0 r1 2	Mem[r0+2] = r1
Conditional	Check equality	beq	I	0100	beq r1 r2 4	If(r1==r2) then go to line 4
Conditional	Compare less than	slt	R	0101	slt r1 r2 r3	If(r1<r2) then r3 = 1 else r3 = 0
Logical	Bit-by-bit and	and	R	0110	and r1 r2 r3	R3 = r1 & r2
Arithmetic	Subtraction	sub	R	0111	sub r1 r2 r3	r3 = r1-r2
Unconditional	Jump	jmp	J	1000	jmp 6	Go to line 6
Logical	Shift left	sll	R	1001	sll r1 r2 r3	r3 = r1<<r2
Arithmetic	Add two numbers	add	R	1010	add r1 r2 r3	r3 = r1 + r2

R Type ISA Format

OpCode	rs	rt	rd
4 bits	3 bits	3 bits	3 bits

I Type ISA Format

OpCode	rs	rt	Immediate
4 bits	3 bits	3 bits	3 bits

J Type ISA Format

OpCode	Target Address
4 bits	9 bits

List of Register:

As we have allocated three bits register so the number of registers will be $2^3 = 8$.

Register Number	Conventional Name	Usage	Binary Value
0	R0	General Purpose	000
1	R1	General Purpose	001
2	R2	General Purpose	010
3	R3	General Purpose	011
4	R4	General Purpose	100
5	R5	General Purpose	101
6	R6	General Purpose	110
7	R7	General Purpose	111

Translating some HLL codes using my designed 13-bit ISA

1. a = a + b

```
# r1 = a, r2 = q

add r1 r2 r1        # r1 gets r1 + r2
```

2. a = a – b

```
# r1 = a, r2 = b

sub r1 r2 r1        # r1 gets r1 – r2
```

3. a = a and b

```
# r1 = a, r2 = b

and r1 r2 r1        # r1 gets r1 and r2
```

4. if(a<b)

```
# r1 = a, r2 = b

        c = 1       # r3 = c

slt r1 r2 r3        # r3 gets 1 if(r1<r2) else r3 gets 0
```

Limitations: I have divided my 13 bits in 4 divisions. That's why we couldn't reserve space for shift amount. Thus, we used the immediate 3-bit space in I type format for shifting purpose.