North South University

Department of Electrical & Computer Engineering

CSE332

Computer Organization and Architecture

Instruction Set Architecture Design

Submitted by:

Name: Md. Mashruf Ehsan

ID: 1831253642

Submitted to:

Tanjila Farah (TnF)

Objectives: My objective was to design a 16 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

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

Formats:

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

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

R Type ISA Format

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

I Type ISA Format

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

J Type ISA Format

OpCode	Target Address
4 bits	12 bits

List of Register:

As we have allocated four bits register so the number of registers will be $2^4 = 16$.

Register Number	Conventional Name	Usage	Binary Value
0	R0	General Purpose	0000
1	R1	General Purpose	0001
2	R2	General Purpose	0010
3	R3	General Purpose	0011
4	R4	General Purpose	0100
5	R5	General Purpose	0101
6	R6	General Purpose	0110
7	R7	General Purpose	0111
8	R8	General Purpose	1000
9	R9	General Purpose	1001
10	R10	General Purpose	1010
11	R11	General Purpose	1011
12	R12	General Purpose	1100
13	R13	General Purpose	1101
14	R14	General Purpose	1110
15	R15	General Purpose	1111

Translating some HLL codes using my designed 16-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 16 bits in 4 divisions. That's why we couldn't reserve space for shift amount. Thus, we used the immediate 4-bit space in I type format for shifting purpose.