Title: Design a 10-bit Custom RISC-V Microprocessor

Course: CSE 332 - Computer Organization & Design

Section – 10

Summer 2022

Submitted to: Mainul Hossain
Assistant Professor,
Department of Electrical & Computer Engineering,
North South University

Submitted by:

	Name	ID
1	Barsha Talukdar	2013929642
2	Shamrana Mahmud	2012216642
3	Saaquib Rahman	2011623642
4	BM Monjur Morshed	2011630642

Introduction: Our task was to design a 10 bit RISC type ISA.

Objectives: Our objectives was to design a 10 Bit ISA which can solve a particular problems i. e. arithmetic addition, shifting etc.

How many Operands?
- There are three operands, which we represented as d, s and t.

Type of Operands?
- Register based
- Memory based

How many Operations? Why?
We allocated 3 bits for the opcode, so the number of instructions can be executed is 2^3 or 8.

Types of Operations?
There will be in total five different types of operations. The categories are:
- Arithmetic
- Logical
- Conditional Branch
- Unconditional Branch
- Data transfer

Category	**Operation**	**Name**	**opcode**	Type	Syntax	Comment
Arithmetic	Addition	add	000	R	add $rt, $rd, $rs	Three register operands
Arithmetic	Subtraction	sub	001	R	sub $rt, $rd, $rs	Three register operands
Arithmetic	Addition immediate	addi	010	I	addi $rt, $rd, const.	Used to add constants
Logical	Shift left logical	sll	011	R	sll $rd, offset	Shift left by constant
Data Transfer	Load word	lw	100	I	lw $rd, offset	Load data from memory to register
Data Transfer	Store word	sw	101	I	Sw offset($rd)	Load data from register to memory
Conditional	Branch on equal	beq	110	I	beq $rd, offset	Check equality if else condition
Unconditional	Jump	jmp	111	J	Jmp offset	Jump to Given location

How many Formats?

We would like to use 3 formats for our ISA

Register type – (R-Type)

Immediate Type – (I- Type)

Jump Type – (j- Type)

(R-Type) ISA Format

opcode	**rs**	**rt**	**rd**	**Shift Amount**
3	**2**	**2**	**2**	**1**

(I-Type) ISA Format

opcode	**rs**	**rt**	**Immediate**
3 bits	2 bits	2 bits	3 bits

(J-Type) ISA Format

Opcode	**Address**
3 bits	7 bits

List of registers

- As we have allocated 2 bits for the registers(rd, rs and rt) so we will have 2^2 = 4 registers and all of them will be store type.

Register Table

Register	**Type**	**Reg. Value**	**Binary Value**
$ac	Saves Values\ Accumulator	0	00
$s1	Saves values	1	01
$s2	Saves values	2	10
$s3	Saves values	3	11

Add: It adds two registers and stores the result in the third register.

ADD.
• Operation: d = s + t
• Syntax: add $rd, $rs, $rt ($rd = $rs +$rt)

Sub
• Operation: d = s – t
• Syntax: sub $rd, $rs, $rt ($rt = $rs -$rt)

Addi: It adds a value from register with an integer value and stores the result in destination register.
• Operation: **t = s + constant**
• Syntax: **addi $rt, $rs, Constant (***$rt = $rs +const.***)**

lw: It loads required value from the memory to the register for calculation
• Operation: t = M[s + offset]
• Syntax: lw $rt, $rs, offset

sw: It stores specific value from register to memory.
• Operation: M[s + offset] = t
• Syntax: sw $rt, $rs, offset

beq: It checks whether the values of two register are same or not. If it is same it performs the following operation
• Operation: if (s == t) jump to offset else goto next line
• Syntax: beq $rs, $rt, offset (if $rs == $rt, goto offset location)

Sll : Shift left by constant we can use sll for multiplication
Sll $rt, $rs, const. ($rt = $rs << const.)

Exit:

Analysis and limitation: We used 3 bit for opcode so we can use an extra 1 bit for shift amount.
10 bit constraints limited our options so we needed to allocate space for opcode, source and destination carefully. For this similar reason, we allocated 3 bit for immediate.

Limitations we have:
1. We have limited option in immediate format. For instance, we can not perform arithmetic operations like(Multiplication, division) and logical operations like AND, OR, NOT…. etc.

2. We cannot perform immediate operation whose size is larger than 2^3 =8 bits.