NORTH SOUTH UNIVERSITY

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Project

Computer Organization and Architecture

CSE332

Summer 2021

Project Part 1: ISA submission.

Course : CSE332.

Group #: 3.

Date: 23-July-21

Submitted to: Dr. Tanzilur Rahman.

Group Members:

Name	**ID**
Istiak Ahmed Sheam	1911139642
Md Ulfat Tahsin	1913057642
Md Saeem Hossain Shanto	1912218642

In this project, we are going to design a 14-bit processor as per the given instruction.

Our processor characteristics are:

- RISC Type processor (single-cycle CPU).
- Load-Store architecture.
- There are 3 operands.
- Operands are register-based.
- There are three formats of data our processor will receive:
  - ➔ R-type.
  - ➔ I-type.
  - ➔ J-type.
- We have 8 registers in the register file.
- We are going to handle 15 different types of operations. And those include:
  - ➔ Arithmetic.
  - ➔ Logical.
  - ➔ Data transfer.
  - ➔ Conditional branch.
  - ➔ Unconditional jump.
  - ➔ External communication.

Register File:

We have 8 registers in the register file as given below:

Each register is of 14 bit size.

Serial # of the register	Name of the register	Used for	Value assigned (3-bit each)
0	$zero	Hard wired to 0	000
1	$intr	IN	001
2	$outr	OUT	010
3	$shanto	Save	011
4	$sheam	Save	100
5	$t0	Temporary	101
6	$tanzil	Temporary	110
7	$tahsin	Temporary	111

Data formats & Operations:

R-type format:

Op (11-14) Source 1 (8-10) Source 2 (5-7) Destination (2-4) Shift (0-1)

Op-code	**rs1**	**rs2**	**rs3**	**shamt**
4-bit	3-bit	3-bit	3-bit	1-bit

Operations:

Serial #	**Category**	**Op-code**	**Operation**	**Example**	**Description**
1	Logical	0000	and	and $shanto, $t0, $tanzil	$tanzil−>$s0 and $t0. The destination register will consist of the value obtained after the logical and operation of value in in the source registers
2	Logical	0001	or	or $shanto, $t0, $tanzil	$tanzil−>$s0 or $t0. The destination register will consist of the value obtained after the logical or operation of value in in the source registers
3	Arithmetic	0010	add	add $shanto, $t0, $tanzil	$tanzil−>$t0 + $s0. The destination register will consist of the value obtained after summing the values in the two source registers
4	Arithmetic	0011	sub	sub $shanto, $t0, $tanzil	$tanzil−>$s0 - $t0. The destination register will consist of the value obtained after subtracting the values in the two source registers
5	Logical	0100	nor	nor $shanto, $t0, $tanzil	$tanzil−>$s0 nor $t0. The destination register will consist of the value obtained after the logical nor operation of value in in the source

					registers
6	Logical	0101	nand	nand $shanto, $t0, $tanzil	$tanzil−>$s0 nand $t0. The destination register will consist of the value obtained after the logical nand operation of value in in the source registers
7	Ext comm	0110	in	in ($inpr)	It is used to scan data into the device with the help of a keyboard to only the register designated for input.
8	Ext comm	0111	out	out ($outr)	It is used to print the data onto the seven segment display that is used as the output device.
9	Logical	1000	slt	slt $t0, $shanto, $sheam	If the values in the source registers are equal, the destination register will have a value of 1/0.

I-type format:

Op (11-14) Source 1 (8-10) Destination/Src (5-7) Immediate (0-4)

Op-code	**rs1**	**rs2**	**immediate**
4-bit	3-bit	3-bit	4-bit

Operations:

Serial #	**Category**	**Op-code**	**Operation**	**Example**	**Description**
10	Arithmetic	1001	addi	addi $shanto, $t0, immediate	$t0−>$s0 + (+/-immediate). The destination register will consist of the value obtained after summing the values in the source register and the (+/-)immediate value.
11	Logical	1010	sll	sll $shanto, $t0, 1(max shift possible for our ISA)	$s0−>$t0 << constant. This will left shift the value of the source register to 1 position maximum (0 to 1) and keep it in the destination register.
12	Data transfer	1011	lw	lw $t0, offset($sheam)	This instruction will go to the memory array with base_register+offset, get the value and load it to the destination register.

					Offset limit 0-15.
13	Data transfer	1100	sw	sw $t0, offset($s2)	This instruction will go to the source register, get the value and store it in the memory array with base_register+offset. Offset limit 0-15.
14	Conditional branch	1101	beq	beq $t0, $shanto, immediate.	If data values in the source registers are equal it jumps to the labeled target/offset location.

J-type format:

Op-code	Target
4-bit	10-bit

Operation:

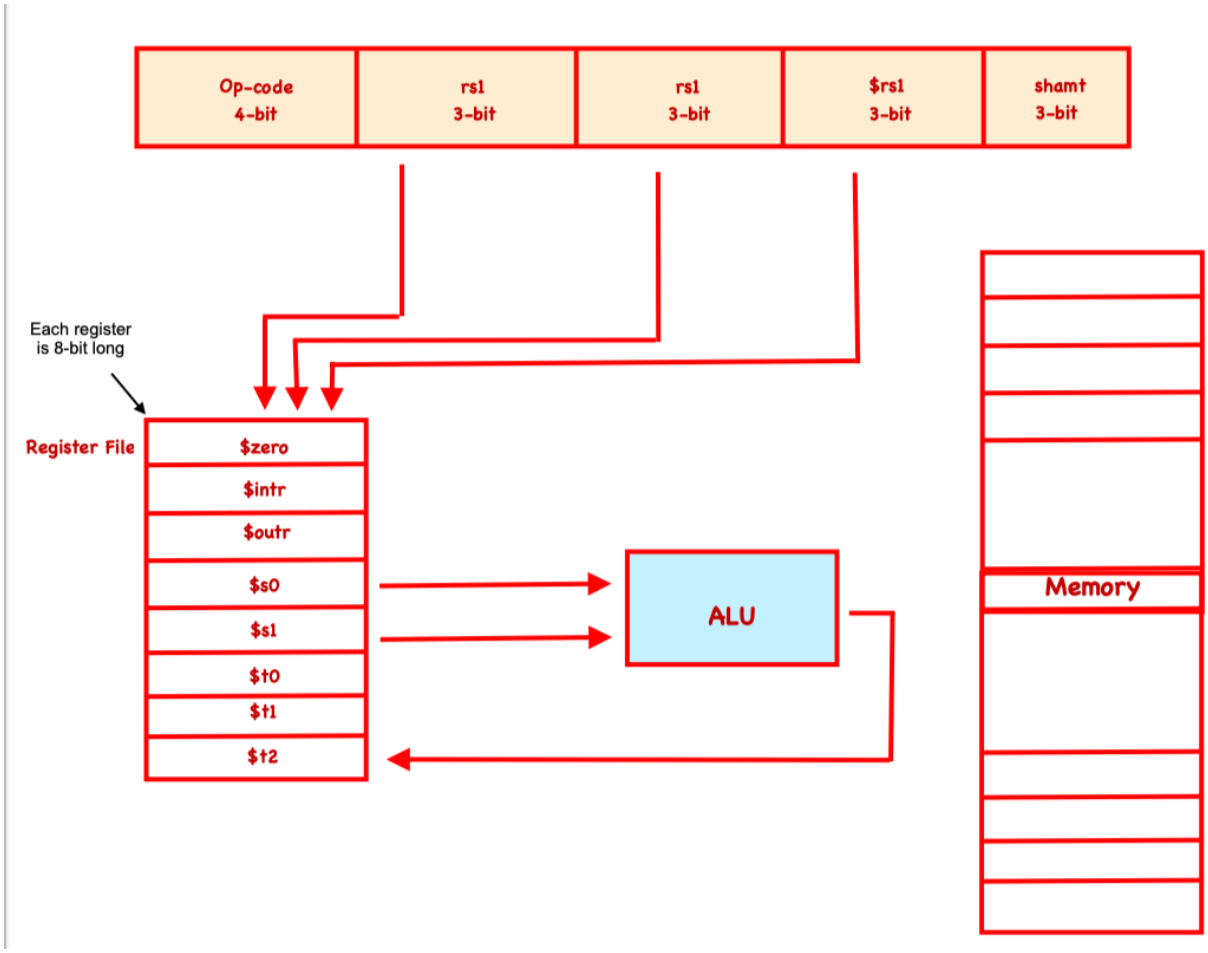
Category	Op-code	Operation	Example	Description
15. Unconditional jump	1110	j	j f	Jumps to target location

Notes on the different types of registers used in our project:

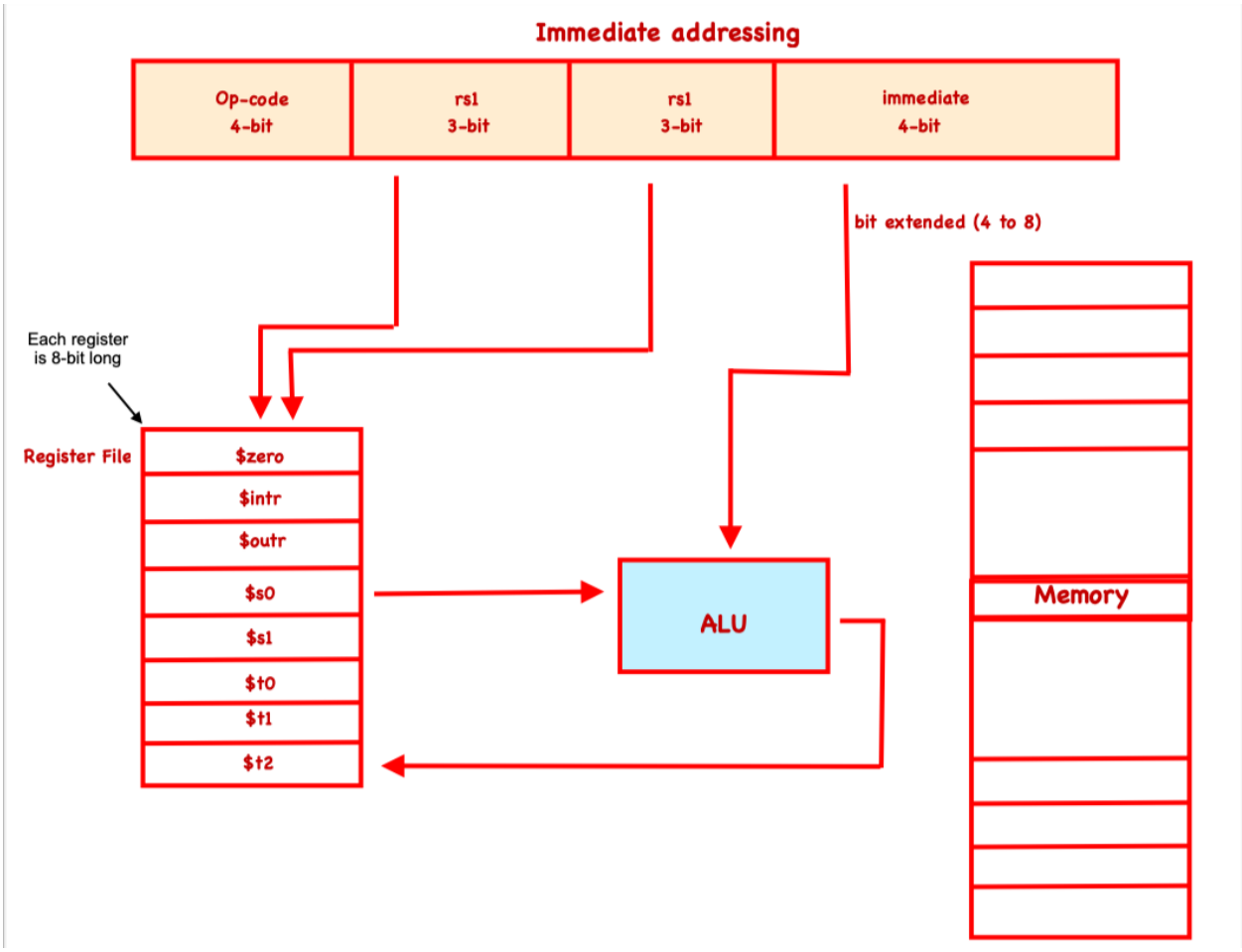
1) Temporary Register ($t): it is used to hold intermediate results. It is required to be initialized before it can be used. There can be one or more temporary registers as per need. These can store temporary results when the ALU performs any kind of temporary or logic operations. They can store operands or addresses that are part of an existing instruction.
2) Save Registers ($s): it is used to hold different values or to be specific location of the values needed to execute any instruction.
3) Zero Register ($zero): it is hard wired with the ground that always provides a zero value. Can be used for multiple tasks like copying, performing inverse operations etc.
4) Input Register (INPR): The Input Register INPR will store the information from the input device keyboard.
5) Output Register (OUTR): The output OUTR receives information, keeps it in the register and transfers it to the output device, 7 segment display.

Addressing Mode:

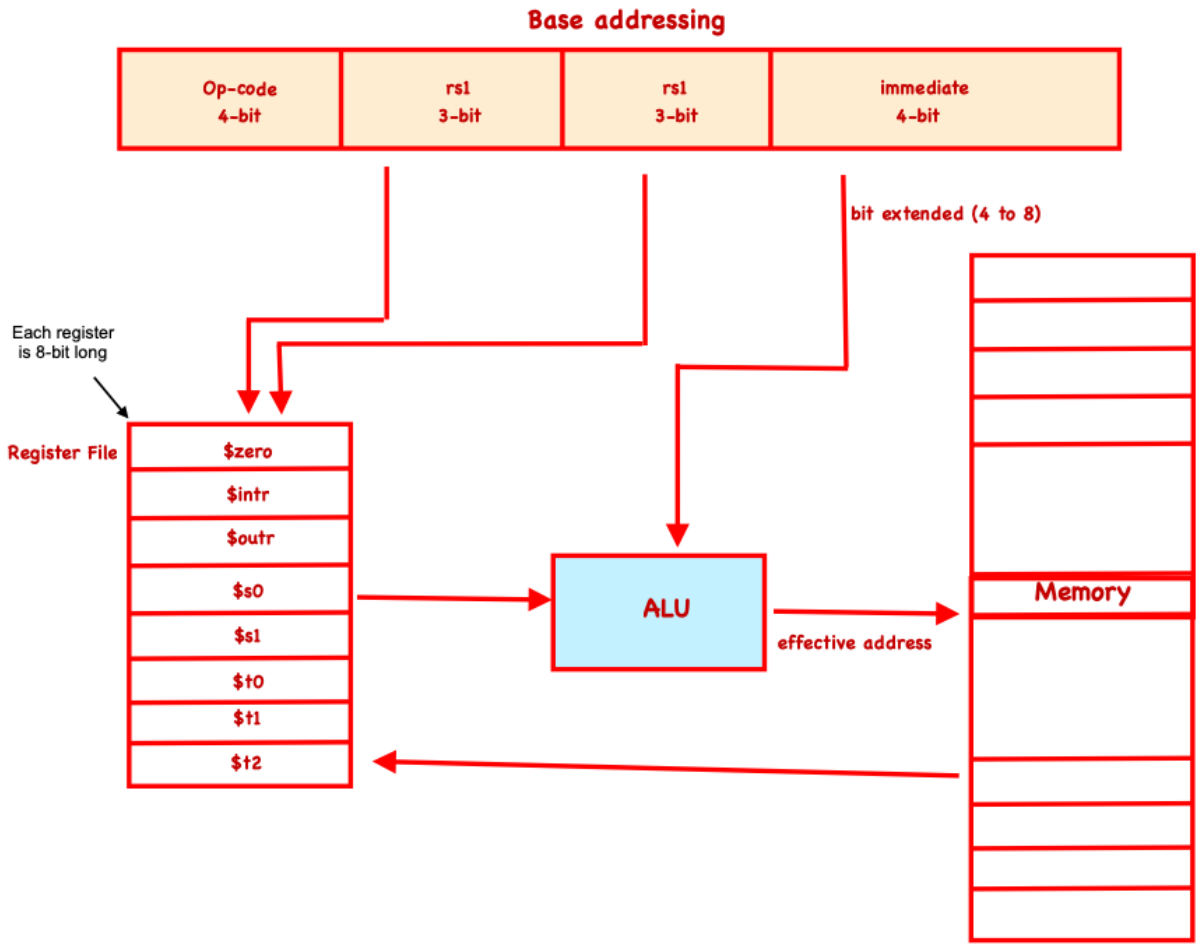
1. Dirict addressing:



2. Immediate addressing:



3. **Base addressing:**



.