**Date to Perform experiment: 17/10/23** **Report Submission Date:23/10/23, 11:59 PM**

**Experiment posted on Moodle:10/10/23**

# Experiment 10:

Design a 32-bit processor (Instruction decode (Instruction decoder + controller) + Instruction execute (ALU)+ register write (Register Bank)) which executes the following 10 instructions**.**

**ADD, SUB, SLL, SLT, SLTU, XOR, SRL, SRA, OR, AND**

## Objective.

Perform post-synthesis simulations for a basic 32-bit processor which executes the following 10 instructions. ADD/SUB/SLL/SLT/SLTU/XOR/SRL/SRA/OR/AND.

The processor will basically have the following modules.

a. A 32-bit register bank which will have 32 registers and each register can store 32-bit data.
b. An instruction decoder module that will slice the 32-bit instructions into the corresponding fields- opcode, func, and rs1,rs2, and rd values
c. A controller module that will generate a corresponding signal for the respective operations depending func and opcode.
d. 32-bit ALU for processing the data present in rs1 and rs2 registers.
e. Register write to store the result back into rd register in the register bank.

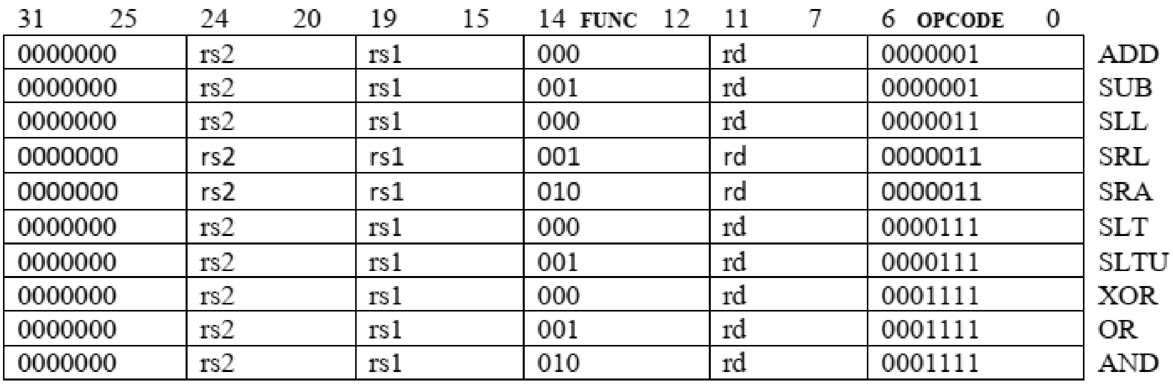
**Note:**

1. Initialize the 32 registers present in the register bank, before executing the instructions.
2. Perform ALU operations on the data's present in 'rs1' and 'rs2' registers. The result is stored on 'rd' register.

**Operations performed by ALU for each instruction.**

| **Sl no:** | **operation** | **functionality** | **Values to be taken for each operation to check the functionality** |
|---|---|---|---|
| 1 | ADD | reg[rd]=reg[rs1]+reg[rs2] | reg[rs1]=0000000F, reg[rs2]=0000000C |
| 2 | SUB | reg[rd]=reg[rs1]-reg[rs2] | reg[rs1]=0000000F, reg[rs2]=0000000C |
| 3 | SLL | reg[rd]=reg[rs1]<< lower 5bits of reg[rs2]<br>(shift left logical) | reg[rs1]=FF0000FF, reg[rs2]=00000004 |
| 4 | SLT | reg[rd]=1, if(reg[rs1]<reg[rs2])<br>Set less than signed | reg[rs1]=70000000, reg[rs2]=F0000000 |
| 5 | SLTU | reg[rd]=1, if(reg[rs1]<reg[rs2])<br>Set less than unsigned | reg[rs1]=70000000, reg[rs2]=F0000000 |
| 6 | XOR | reg[rd]=reg[rs1]^reg[rs2] | reg[rs1]=0000000F, reg[rs2]=0000000C |
| 7 | SRL | reg[rd]=reg[rs1]>>lower 5 bits of reg[rs2]<br>(shift right logical) | reg[rs1]=FF0000FF, reg[rs2]=00000004 |
| 8 | SRA | reg[rd]=reg[rs1]>>>lower 5 bits of reg[rs2]<br>(shift right arithmetic) | reg[rs1]=FF0000FF, reg[rs2]=00000004 |
| 9 | OR | reg[rd]=reg[rs1] \| reg[rs2] | reg[rs1]=0000000F, reg[rs2]=0000000C |
| 10 | AND | reg[rd]=reg[rs1]&reg[rs2] | reg[rs1]=0000000F, reg[rs2]=0000000C |

**Instruction set Specification of the 32-bit processor**



| 31 25 | 24 20 | 19 15 | 14 FUNC 12 | 11 7 | 6 OPCODE 0 | |
|---|---|---|---|---|---|---|
| 0000000 | rs2 | rs1 | 000 | rd | 0000001 | ADD |
| 0000000 | rs2 | rs1 | 001 | rd | 0000001 | SUB |
| 0000000 | rs2 | rs1 | 000 | rd | 0000011 | SLL |
| 0000000 | rs2 | rs1 | 001 | rd | 0000011 | SRL |
| 0000000 | rs2 | rs1 | 010 | rd | 0000011 | SRA |
| 0000000 | rs2 | rs1 | 000 | rd | 0000111 | SLT |
| 0000000 | rs2 | rs1 | 001 | rd | 0000111 | SLTU |
| 0000000 | rs2 | rs1 | 000 | rd | 0001111 | XOR |
| 0000000 | rs2 | rs1 | 001 | rd | 0001111 | OR |
| 0000000 | rs2 | rs1 | 010 | rd | 0001111 | AND |

**Results.**

Verify the functionality of the designed 32-bit processor and figure out the number of LUTs, flip flops, power and delay required in the corresponding designs

| | #LUTs | #Flip flops | Power | Delay |
|---|---|---|---|---|
| **Processor core** | | | | |

**NOTE:** Kindly provide the RTL schematic and simulation results of the designs in the report. Also upload the code in Moodle, during report submission.