1. 指令集概述

本实验所采用指令集为 RISC-V RV32I (base integer Instruction)中的部分指令，具体指令编码如下表所示。

31 25	24 20	19 15	14 12	11 7	6 0	指令名
R-type						
Funct7	**Rs2**	**Rs1**	**Funct3**	**Rd**	**Opcode**	
0000000	Rs2	Rs1	000	rd	0110011	add
0000000	Rs2	Rs1	111	rd	0110011	and
0000001	Rs2	Rs1	000	rd	0110011	mul
0000000	Rs2	Rs1	110	rd	0110011	or
0100000	Rs2	Rs1	000	rd	0110011	sub
0000000	Rs2	Rs1	010	rd	0110011	slt
0000000	Rs2	Rs1	011	rd	0110011	sltu
0000000	Rs2	Rs1	100	rd	0110011	xor
0100000	Rs2	Rs1	101	rd	0110011	sra
0000000	Rs2	Rs1	101	rd	0110011	srl
0000000	Rs2	Rs1	001	rd	0110011	sll
0000000	shamt	Rs1	001	rd	0010011	slli
0100000	shamt	Rs1	101	rd	0010011	srai
0000000	shamt	Rs1	101	rd	0010011	srli
I-type						
Imm[11:0]		**Rs1**	**Funct3**	**Rd**	**Opcode**	
Imm[11:0]		Rs1	000	rd	0010011	addi
Imm[11:0]		Rs1	111	rd	0010011	andi
Imm[11:0]		Rs1	110	rd	0011011	ori
Imm[11:0]		Rs1	010	rd	0010011	slti
Imm[11:0]		Rs1	011	rd	0010011	sltiu
Imm[11:0]		Rs1	100	rd	0010011	xori
Offset[11:0]		Rs1	010	rd	0000011	lb
Offset[11:0]		Rs1	100	rd	0000011	lbu
Offset[11:0]		Rs1	001	rd	0000011	lh
Offset[11:0]		Rs1	101	rd	0000011	lhu
Offset[11:0]		Rs1	010	rd	0000011	Lw
S-type						
Imm[11:5]	**Rs2**	**Rs1**	**Funct3**	**Imm[4:0]**	**Opcode**	
Imm[11:5]	Rs2	Rs1	000	Offset[4:0]	0100011	sb
Imm[11:5]	Rs2	Rs1	001	Offset[4:0]	0100011	sh
Imm[11:5]	Rs2	Rs1	010	Offset[4:0]	0100011	sw
B-type						
Imm[12,10:5]	**Rs2**	**Rs1**	**Funct3**	**Imm[4:1,11]**	**Opcode**	

Offset[12,10:5]	Rs2	Rs1	000	Offset[4:1,11]	1100011	beq
Offset[12,10:5]	Rs2	Rs1	001	Offset[4:1,11]	1100011	bne
Offset[12,10:5]	Rs2	Rs1	100	Offset[4:1,11]	1100011	blt
Offset[12,10:5]	Rs2	Rs1	110	Offset[4:1,11]	1100011	bltu
Offset[12,10:5]	Rs2	Rs1	101	Offset[4:1,11]	1100011	bge
Offset[12,10:5]	Rs2	Rs1	111	Offset[4:1,11]	1100011	bgeu
U-type						
Imm[31:12]				**rd**	**Opcode**	
Imm[31:12]				rd	0110111	lui

2. 指令功能

2.1 R-type 指令

R-type 指令可以分为两个小类别，一类为普通算术逻辑运算指令，一类为移位指令。

普通算术逻辑运算的指令编码中 rs2，rs1 编码域分别为两个操作数对应的寄存器编号，rd 域表示目标寄存器的编号，通过 funct 和 opcode 编码域值的不同来确定操作的类型。因此这一类指令完成的操作可以概括为

rd = rs1 OP. rs2

31 25	**24 20**	**19 15**	**14 12**	**11 7**	**6 0**	指令名
R-type						
Funct7	**Rs2**	**Rs1**	**Funct3**	**Rd**	**Opcode**	
0000000	Rs2	Rs1	000	rd	0110011	add
0000000	Rs2	Rs1	111	rd	0110011	and
0000001	Rs2	Rs1	000	rd	0110011	mul
0000000	Rs2	Rs1	110	rd	0110011	or
0100000	Rs2	Rs1	000	rd	0110011	sub
0000000	Rs2	Rs1	010	rd	0110011	slt
0000000	Rs2	Rs1	011	rd	0110011	sltu
0000000	Rs2	Rs1	100	rd	0110011	xor

指令名	功能
add	Rd= rs1 + rs2，忽略溢出
and	Rd= rs1 & rs2
mul	Rd= rs1 * rs2，忽略溢出
or	Rd= rs1 | rs2
sub	Rd= rs1 - rs2，忽略溢出
slt	Rd= rs1 < rs2 ? 1 : 0 (rs1 和 rs0 为有符号数)
sltu	Rd= rs1 < rs2 ? 1 : 0 (rs1 和 rs0 为无符号数)
xor	Rd= rs1 ^ rs2

移位指令可以分成普通移位和立即数移位两种。

普通移位操作，移位的位数存放在 rs2 寄存器中，rs1 根据 rs2 的值进行移位后结果保存到 rd 寄存器。

立即数移位操作，移位的位数在指令编码的 shamt 域，rs1 根据 shamt 的值进行移位后结果保存到 rd 寄存器中。

31 25	24 20	19 15	14 12	11 7	6 0	指令名
R-type						
Funct7	**Rs2**	**Rs1**	**Funct3**	**Rd**	**Opcode**	
0000000	Rs2	Rs1	001	rd	0110011	sll
0100000	Rs2	Rs1	101	rd	0110011	sra
0000000	Rs2	Rs1	101	rd	0110011	srl
0000000	shamt	Rs1	001	rd	0010011	slli
0100000	shamt	Rs1	101	rd	0010011	srai
0000000	shamt	Rs1	101	rd	0010011	srli

指令名	功能
sll	逻辑左移，rd = rs1 << rs2[4:0];
sra	算术右移，rd = rs1 >> rs2[4:0];最高位填充符号位
srl	逻辑右移，rd = rs1 >> rs2[4:0];最高位填充 0
slli	立即数逻辑左移，rd = rs1 << shamt;
srai	立即数算术右移，rd = rs1 >> shamt;最高位填充符号位
srli	立即数逻辑右移，rd = rs1 >> shamt;最高位填充 0

2.2 I-type 指令

I-type 类指令为有立即数参与操作的指令。

立即数参与的算术逻辑运算指令如下表，其中 rs1 为第一操作数，第二操作数为立即数 imm[11：0]经过符号位扩展后的一个 32bit 操作数 imm32，运算后的结果写入 rd

31 25	24 20	19 15	14 12	11 7	6 0	指令名
I-type						
Imm[11:0]		**Rs1**	**Funct3**	**Rd**	**Opcode**	
Imm[11:0]		Rs1	000	rd	0010011	addi
Imm[11:0]		Rs1	111	rd	0010011	andi
Imm[11:0]		Rs1	110	rd	0011011	ori
Imm[11:0]		Rs1	010	rd	0010011	slti
Imm[11:0]		Rs1	011	rd	0010011	sltiu
Imm[11:0]		Rs1	100	rd	0010011	xori

指令名	功能
addi	Rd= rs1 + imm32，忽略溢出
andi	Rd= rs1 & imm32

ori	Rd= rs1 | imm32
slti	Rd= rs1 < imm32 ? 1 : 0 (rs1 和 imm32 为有符号数)
sltiu	Rd= rs1 < imm32 ? 1 : 0 (rs1 和 imm32 为无符号数)
xori	Rd= rs1 ^ imm32

立即数参与的 memory 读取类指令中立即数 imm[11:0]经过符号位扩展后形成 32bit 的地址偏移量 offset32，rs1 寄存器作为地址的基址，rs1+offset32 形成最终 32bit byte 地址访问数据存储器，从存储器读出的数据存入 rd 寄存器

31 25	24 20	19 15	14 12	11 7	6 0	指令名
I-type						
Imm[11:0]		**Rs1**	**Funct3**	**Rd**	**Opcode**	
Offset[11:0]		Rs1	010	rd	0000011	lb
Offset[11:0]		Rs1	100	rd	0000011	lbu
Offset[11:0]		Rs1	001	rd	0000011	lh
Offset[11:0]		Rs1	101	rd	0000011	lhu
Offset[11:0]		Rs1	010	rd	0000011	Lw

指令名	功能
lb	rs1+offset32 形成的 byte 地址访问 mem 得到 8bit 数据，将该数据符号位扩展成 32bit 后写入 rd 寄存器
lbu	s1+offset32 形成的 byte 地址访问 mem 得到 8bit 数据，将该数据 0 扩展成 32bit 后写入 rd 寄存器
lh	rs1+offset32 形成的半字地址（最低位为 0）访问 mem 得到 16bit 数据，将该数据符号位扩展成 32bit 后写入 rd 寄存器
lhu	s1+offset32 形成的半字地址（最低位为 0）访问 mem 得到 16bit 数据，将该数据 0 扩展成 32bit 后写入 rd 寄存器
lw	rs1+offset32 形成的字地址（最低 2 位为 0）访问 mem 得到 32bit 数据写入 rd 寄存器

2.3 S-type 指令

S-type 为指令存储类指令，地址的 12bit offset 由指令域的[31:25]与[11:7]拼接而成，rs1+offset 经过符号位扩展后形成 32bit 的地址偏移量 offset32。Rs2 为需要存储入 mem 的数据

31 25	24 20	19 15	14 12	11 7	6 0	指令名
S-type						
Imm[11:5]	**Rs2**	**Rs1**	**Funct3**	**Imm[4:0]**	**Opcode**	
Offset [11:5]	Rs2	Rs1	000	Offset[4:0]	0100011	sb
Offset [11:5]	Rs2	Rs1	001	Offset[4:0]	0100011	sh
Offset [11:5]	Rs2	Rs1	010	Offset[4:0]	0100011	sw

指令名	功能
sb	将 rs2 的低 8 位存入 rs1+offset32 形成的 byte 地址

sh	将 rs2 的低 16 位存入 rs1+offset32 形成的半字地址
sw	将 rs2 存入 rs1+offset32 形成的字地址

2.4 B-type 指令

B-type 指令为跳转指令，其跳转地址的生成方式为 13bit 的 offset[12:0]经过符号位扩展后形成 32bit 跳转偏移量 offset32，和当前指令对应的 pc 相加后得到跳转地址。Rs1 和 rs2 作为条件判断的两个操作数。

U-type 的 lui 指令为寄存器立即数赋值指令

31 25	24 20	19 15	14 12	11 7	6 0	指令名
B-type						
Imm[12,10:5]	**Rs2**	**Rs1**	**Funct3**	**Imm[4:1,11]**	**Opcode**	
Offset[12,10:5]	Rs2	Rs1	000	Offset[4:1,11]	1100011	beq
Offset[12,10:5]	Rs2	Rs1	001	Offset[4:1,11]	1100011	bne
Offset[12,10:5]	Rs2	Rs1	100	Offset[4:1,11]	1100011	blt
Offset[12,10:5]	Rs2	Rs1	110	Offset[4:1,11]	1100011	bltu
Offset[12,10:5]	Rs2	Rs1	101	Offset[4:1,11]	1100011	bge
Offset[12,10:5]	Rs2	Rs1	111	Offset[4:1,11]	1100011	bgeu
U-type						
Imm[31:12]				**rd**	**Opcode**	
Imm[31:12]				rd	0110111	lui

指令名	功能
beq	如果 rs1=rs2, 跳转到 pc+offset32
bne	如果 rs1!=rs2, 跳转到 pc+offset32
blt	如果 rs1<rs2(rs1,rs2 为有符号数), 跳转到 pc+offset32
bltu	如果 rs1<rs2(rs1,rs2 为无符号数),跳转到 pc+offset32
bge	如果 rs1>=rs2(rs1,rs2 为有符号数), 跳转到 pc+offset32
bgeu	如果 rs1>=rs2(rs1,rs2 为无符号数),跳转到 pc+offset32
Lui	将 20bit 立即数 imm 低位拼接 12'd0 后写入 rd 寄存器， 即 rd={imm[31:12], 12'h000}