



**计算机组成原理**

**课 程 设 计**

团队实验报告

|  |  |
| --- | --- |
| 学 院 | 计算机学院 |
| 专 业 | 计算机科学与技术 |
| 指导老师 | 王殿欣 |
| 组 长 | 杨昊 |
| 组 员 | 王欣宇、李昊阳 |
| 组长联系方式 | 13818794782 |

二O二一年八月

目录

[第一章 项目简述 1](#_Toc143776920)

[第二章 组员分工 1](#_Toc143776921)

[第三章 设计目的 1](#_Toc143776922)

[第四章 设计环境 1](#_Toc143776923)

[第五章 设计原理及内容 2](#_Toc143776924)

[5.1 CPU整体架构 2](#_Toc143776925)

[5.1.1 指令集选取 2](#_Toc143776926)

[5.1.2 数据通路 5](#_Toc143776927)

[5.1.3 控制逻辑 5](#_Toc143776928)

[5.2 流水线冒险问题以及解决方案 5](#_Toc143776929)

[5.3 Cache的设计（可选） 6](#_Toc143776930)

[5.4 中断的设计（可选） 6](#_Toc143776931)

[5.5 其他特色（可选） 6](#_Toc143776932)

[第六章 设计与实现 6](#_Toc143776933)

[第七章 测试 6](#_Toc143776934)

[7.1 仿真测试用例、结果以及分析 6](#_Toc143776935)

[第八章 问题及解决方法 6](#_Toc143776936)

[第九章 心得体会及总结 6](#_Toc143776937)

[第十章 参考文献有价值的资源推荐 6](#_Toc143776938)

[附录A：RISC-V指令列表 6](#_Toc143776939)

# 项目简述

本项目实现了一种基于RISC-V架构的多周期流水线CPU，实现了完备的移位与逻辑运算、包括乘除在内的大部分算术运算，以及跳转指令、分支指令、加载与存储指令等45条指令。项目实现了经典的五级流水线CPU，解决了流水线中数据相关、结构相关和控制相关问题。对所有指令在vivado上进行行为仿真，通过测试。最终设计并编写汇编测试算法，结合汇编与接口设计实验，最终上板验证成功。

# 组员分工

小组分工见表2.1 小组分工。

|  |  |
| --- | --- |
| **组员** | **分工任务** |
| 组长：杨昊 |  |
| 组员1：王欣宇 |  |
| 组员2：李昊阳 |  |

表2.1 小组分工

# 设计目的

主要任务是以CPU作为设计核心，应用已经学过的专业知识，设计并实现一台简单的计算机系统，并测试通过。

# 设计环境

|  |  |
| --- | --- |
| 操作系统 | Windows x64 10 Pro 22H2 |
| 编程语言 | Verilog |
| EDA工具 | Vivado v2019.2 (64-bit) |
| 汇编语言 | 基于RISC-V指令集的汇编语言 |
| 汇编语言与Verilog代码编辑器 | Visual Studio Code 1.81.1 |
| 开发板 | xc7a35tcsg324-1 |

# 设计原理及内容

## CPU整体架构

本流水线CPU覆盖45条指令(见表5.1.1-1/2)，下面将从数据通路和控制逻辑阐述设计思路。

### 指令集选取

本项目决定实现RISC-V的32位子指令集，主要为RV32I与RV32M指令集，包括了六种基本指令格式，分别是：用于寄存器-寄存器操作的 R 类型指令，用于短立即数和访存 load 操作的 I 型指令，用于访存 store 操作的 S 型指令，用于条件跳转操作的 B 类型指令，用于长立即数的 U 型指令和用于无条件跳转的 J 型指令。所有指令都是32位长，以简化指令编码。

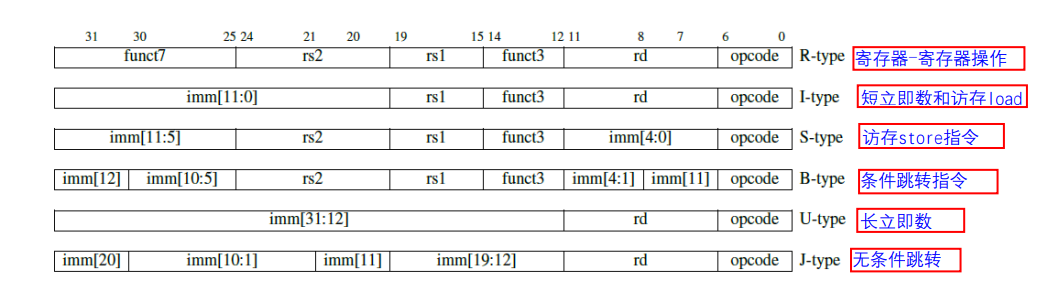


图5.1.1-1 RISC-V指令格式

RV32I是RISC-V 基础整数指令集，所有指令如图5.1.1-2所示，我们决定实现其中的37条指令。

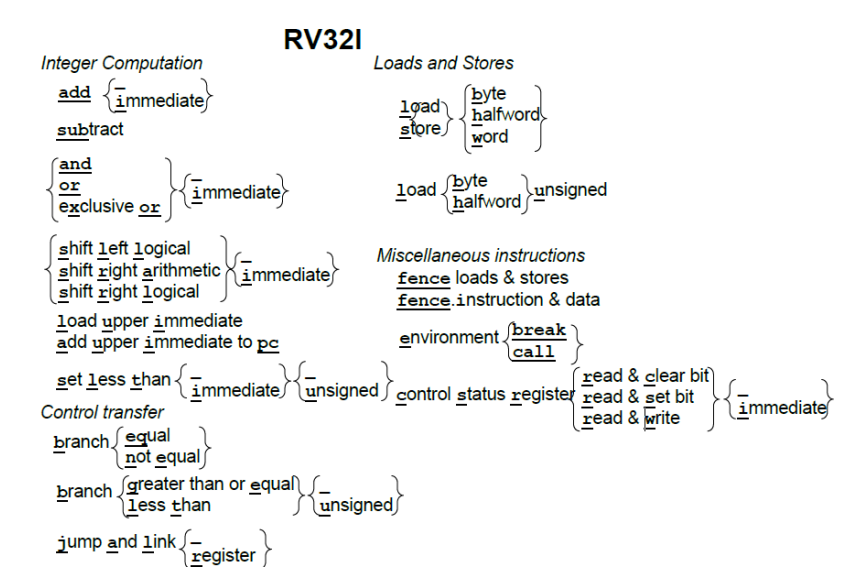


图5.1.1-2 RV32I指令图示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 分类 | 名称 | 格式 | 指令 | 用法 |
| 1 | Shifts | 逻辑左移 | R | SLL/sll | SLL rd,rs1,rs2 |
| 2 | 立即数逻辑左移 | I | SLLI/slli | SLLI rd,rs1,shamt |
| 3 | 逻辑右移 | R | SRL/srl | SRL rd,rs1,rs3 |
| 4 | 立即数逻辑右移 | I | SRLI/srli | SRLI rd,rs1,shamt |
| 5 | 算术右移 | R | SRA/sra | SRA rd,rs1,rs5 |
| 6 | 立即数算术右移 | I | SRAI/srai | SRAI rd,rs1,shamt |
| 7 | Arithmetic | 加 | R | ADD/add | ADD rd,rs1,rs2 |
| 8 | 加立即数 | I | ADDI/addi | ADDI rd,rs1,imm |
| 9 | 减 | R | SUB/sub | SUB rd,rs1,rs2 |
| 10 | 高位立即数加载 | U | LUI/lui | LUI rd,imm |
| 11 | PC 加立即数 | U | AUIPC/auipc | AUIPC rd,imm |
| 12 | Logical | 异或 | R | XOR/xor | XOR rd,rs1,rs2 |
| 13 | 立即数异或 | I | XORI/xori | XORI rd,rs1,imm |
| 14 | 取或 | R | OR/or | OR rd,rs1,rs2 |
| 15 | 立即数取或 | I | ORI/ori | ORI rd,rs1,imm |
| 16 | 与 | R | AND/and | AND rd,rs1,rs2 |
| 17 | 与立即数 | I | ANDI/andi | ANDI rd,rs1,imm |
| 18 | Compare | 小于则置位 | R | SLT/alt | SLT rd,rs1,rs2 |
| 19 | 小于立即数则置位 | I | SLTI/slti | SLTI rd,rs1,imm |
| 20 | 无符号小于则置位 | R | SLTU/sltu | SLTU rd,rs1,rs2 |
| 21 | 无符号小于立即数则置位 | I | SLTIU/sltiu | SLTIU rd,rs1,imm |
| 22 | Branches | 相等时分支 | B | BEQ/beq | BEQ rd,rs1,imm |
| 23 | 不相等时分支 | B | BNE/bne | BNE rd,rs1,imm |
| 24 | 小于时分支 | B | BLT/blt | BLT rd,rs1,imm |
| 25 | 大于等于时分支 | B | BGE/bge | BGE rd,rs1,imm |
| 26 | 无符号小于等于时分支 | B | BLTU/bltu | BLTU rd,rs1,imm |
| 27 | 无符号大于等于时分支 | B | BGEU/bgeu | BGEU rd,rs1,imm |
| 28 | Jump＆Link | 跳转并链接 | J | JAL/jal | JAL rd,imm |
| 29 | 跳转并寄存器链接 | I | JALR/jalr | JALR rd,rs1,imm |
| 30 | Loads | 字节加载 | I | LB/lb | LB rd,rs1,imm |
| 31 | 半字加载 | I | LH/lh | LH rd,rs2,imm |
| 32 | 无符号字节加载 | I | LBU/lbu | LBU rd,rs3,imm |
| 33 | 无符号半字加载 | I | LHU/lhu | LHU rd,rs4,imm |
| 34 | 字加载 | I | LW/lw | LW rd,rs5,imm |
| 35 | Stores | 存字节 | S | SB/sb | SB rs1,rs2,imm |
| 36 | 存半字 | S | SH/sh | SH rs1,rs2,imm |
| 37 | 存字 | S | SW/sw | SW rs1,rs2,imm |

表5.1.1-1 37条RV32I指令

RV32M 向 RV32I 中添加了整数乘法和除法指令，我们决定实现其中的8条指令。

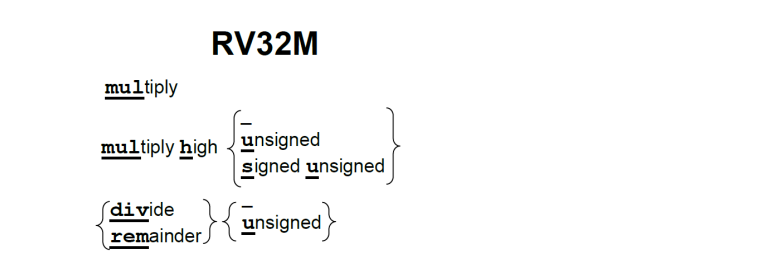


图5.1.1-3 RV32M指令图示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 分类 | 名称 | 格式 | 指令 | 用法 |
| 38 | Multiply | 乘 | R | MUL/mul | MUL rd,rs1,rs2 |
| 39 | 高位乘 | R | MULH/mulh | MULH rd,rs1,rs2 |
| 40 | 高位有符号-无符号乘 | R | MULHSU/mulhsu | MULHSU rd,rs1,rs2 |
| 41 | 高位无符号乘 | R | MULHU/mulhu | MULHU rd,rs1,rs2 |
| 42 | Divide | 除法 | R | DIV/div | DIV rd,rs1,rs2 |
| 43 | 无符号除法 | R | DIVU/divu | DIVU rd,rs1,rs2 |
| 44 | Remainder | 求余数 | R | REM/rem | REM rd,rs1,rs2 |
| 45 | 求无符号数的余数 | R | REMU/remu | REMU rd,rs1,rs2 |

表5.1.1-2 8条RV32M指令

每条指令的详细信息：指令名称、操作数、寄存器传输级定义、指令格式类型、中文描述等详见附录A。

### 数据通路

数据通路文字描述和图示。

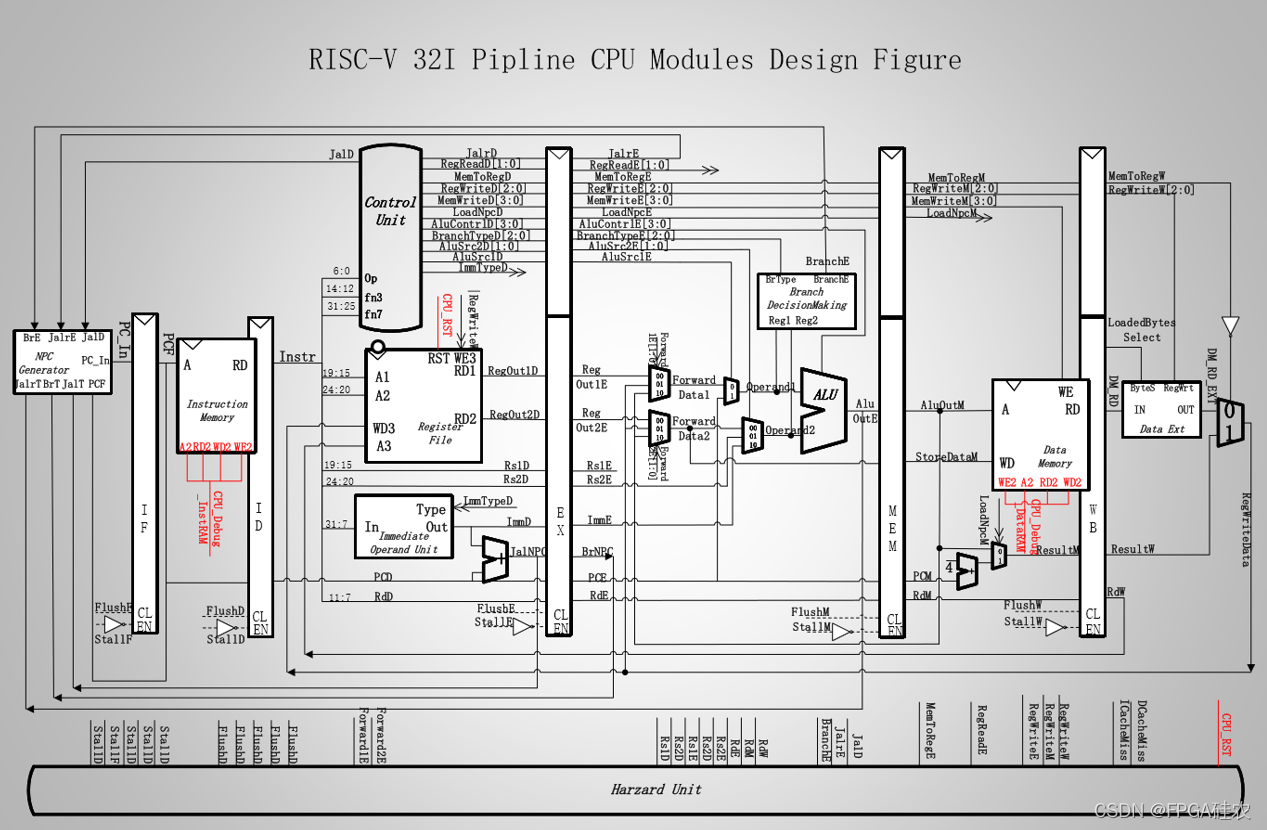


图5.1.2-1 5级流水线CPU数据通路

### 控制逻辑

内容不限定以下内容：

示例：分析实现控制逻辑采用的方式：组合逻辑？微程序等。

控制信号列表，控制信号取值的含义，控制信号编码、时序分配、控制存储器内容，微指令编码等。

## 流水线冒险问题以及解决方案

## Cache的设计（可选）

## 中断的设计（可选）

## 其他特色（可选）

# 设计与实现

# 测试

## 仿真测试用例、结果以及分析

功能性测试

性能测试

下板测试

# 问题及解决方法

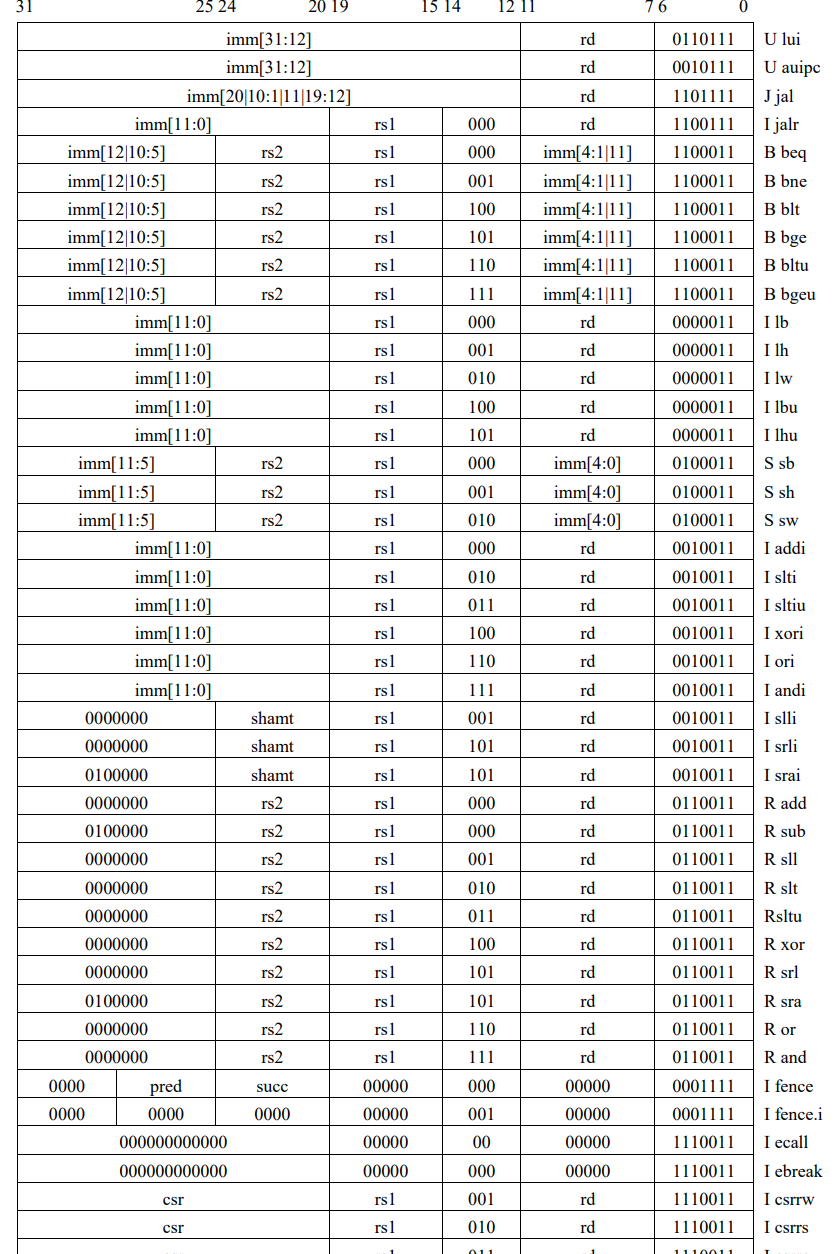
# 心得体会及总结

# 参考文献有价值的资源推荐

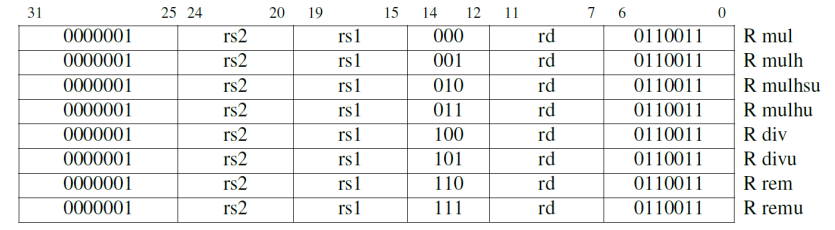
# 附录A：RISC-V指令列表

本附录列出了所选取的指令集的更详细信息，每个条目都包括指令名称、操作数、寄存器传输级定义、指令格式类型、中文描述，以及一张带有操作码的指令布局图。此外还给出了总的RV32I与RV32M操作码映射，包括指令布局，操作码，格式类型和名称。

所有资料均来自开放的RISC-V手册，作者是David Patterson与Andrew Waterman，译者是勾凌睿、黄成、刘志刚等人。我们选取了所需要的指令。

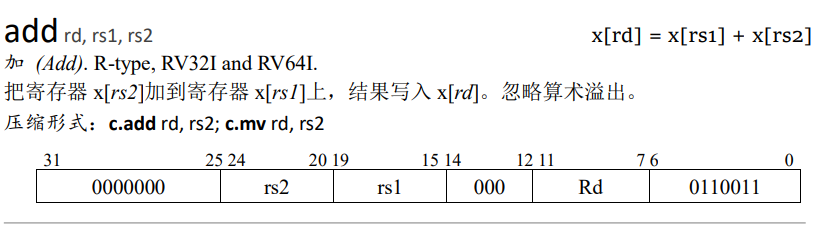


图A-1 RV32I操作码映射

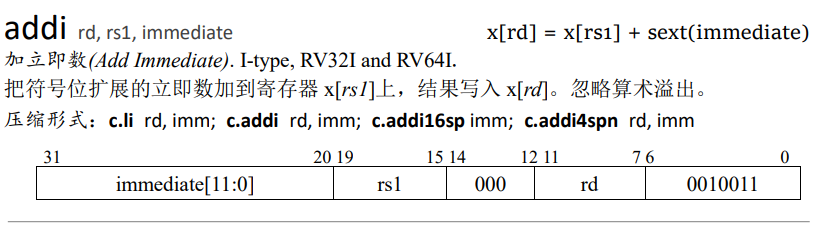


图A-2 RV32M操作码映射

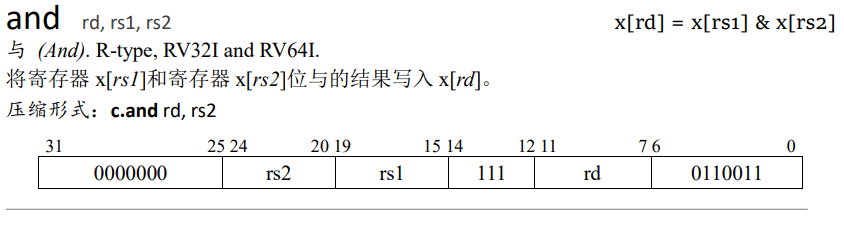
ADD:



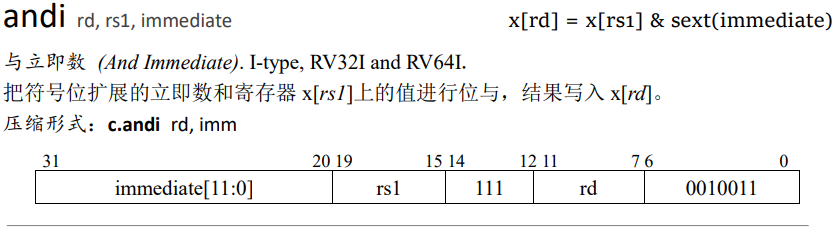
ADDI:



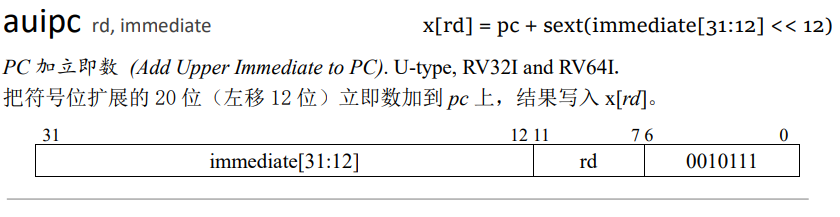
AND:



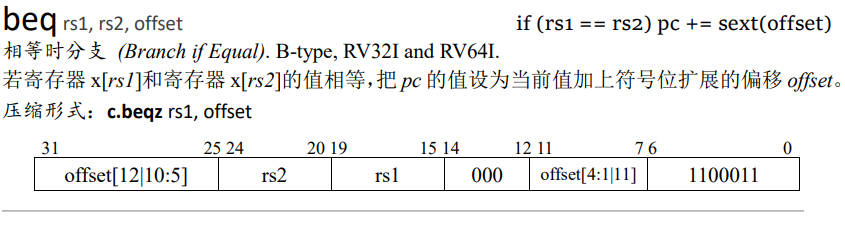
ANDI:



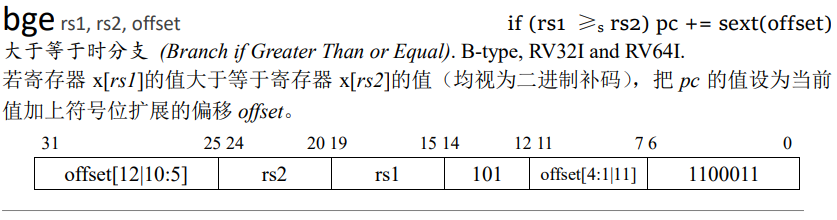
AUIPC:



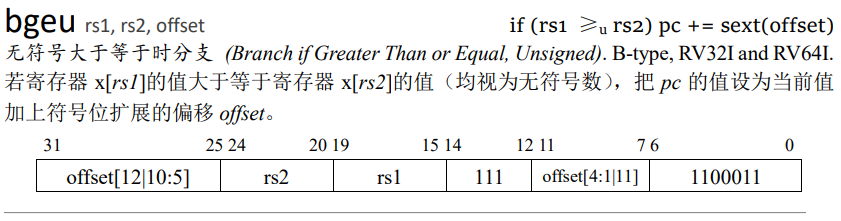
BEQ:



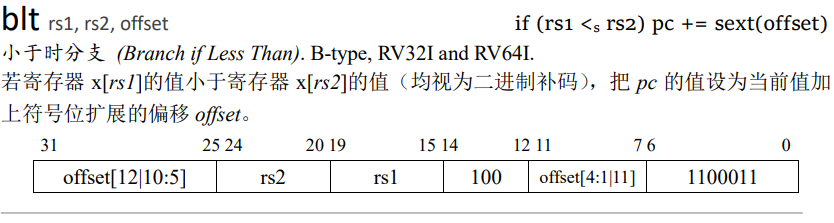
BGE:



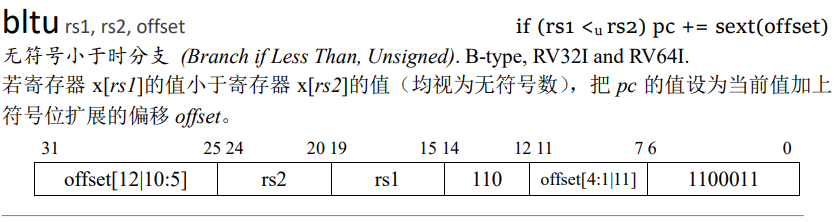
BGEU:



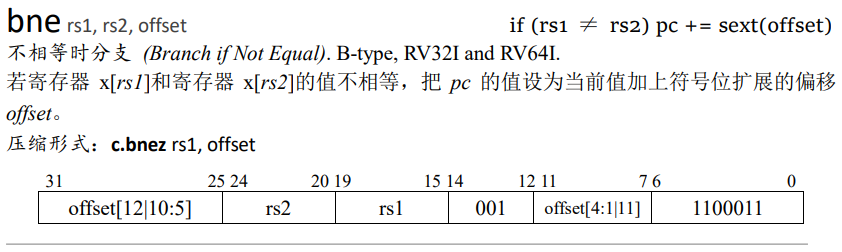
BLT:



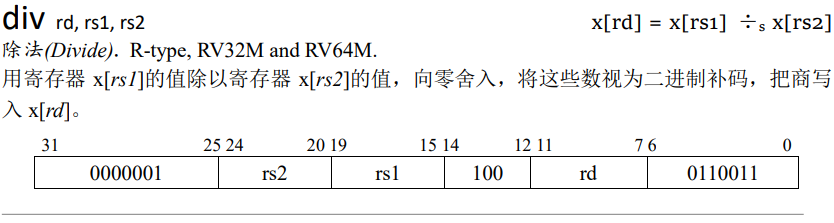
BLTU:



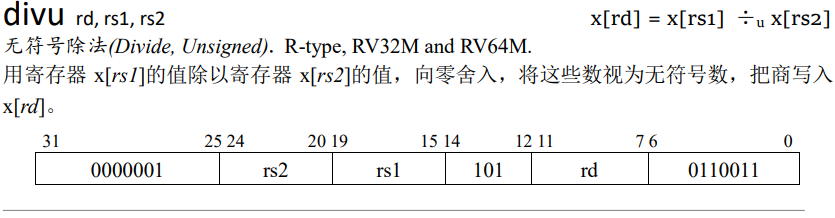
BNE:



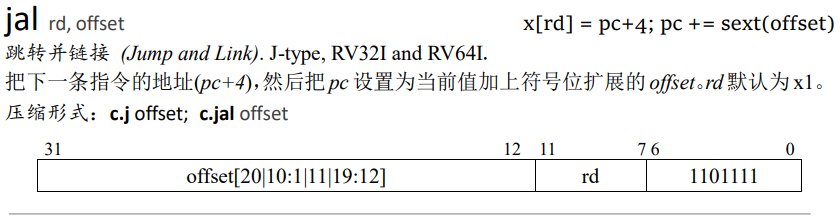
DIV:



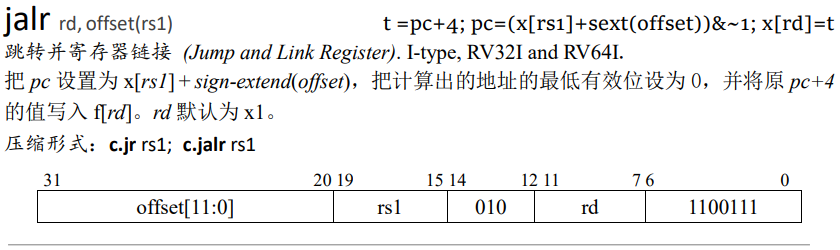
DIVU:



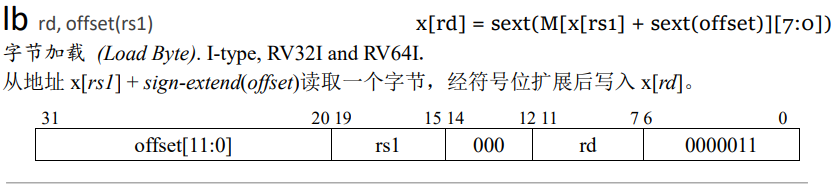
JAL:



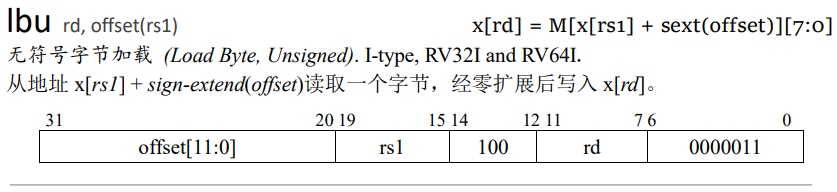
JALR:



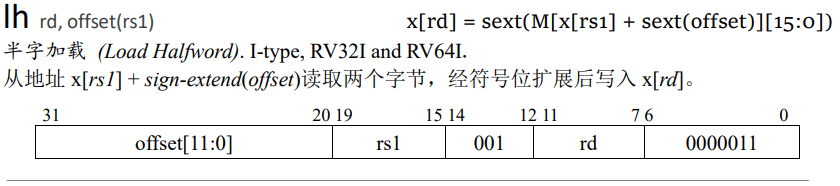
LB:



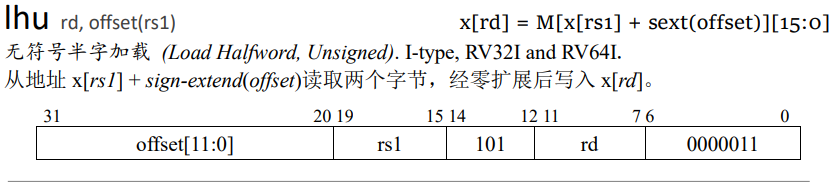
LBU:



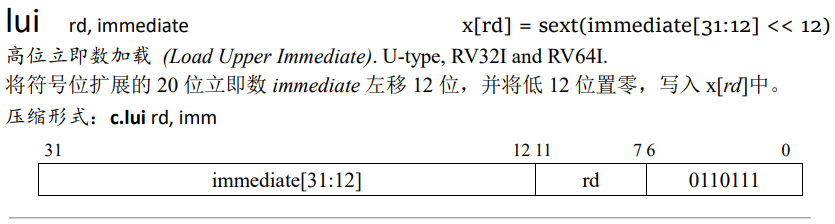
LH:



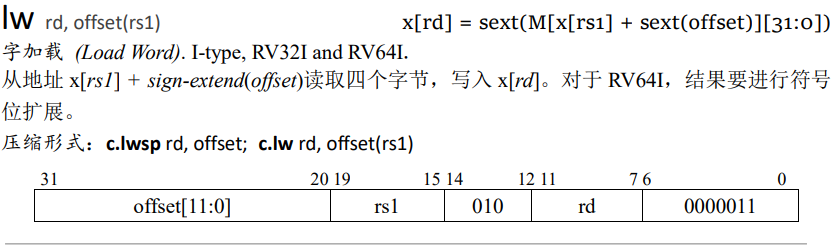
LHU:



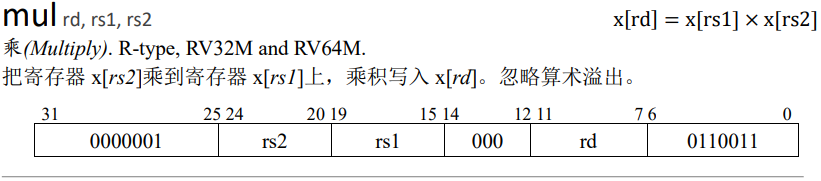
LUI:



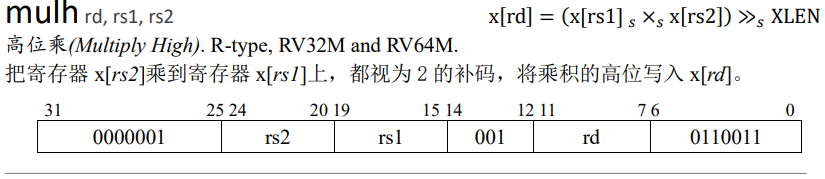
LW:



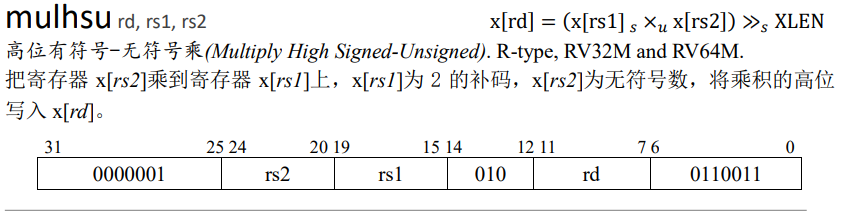
MUL:



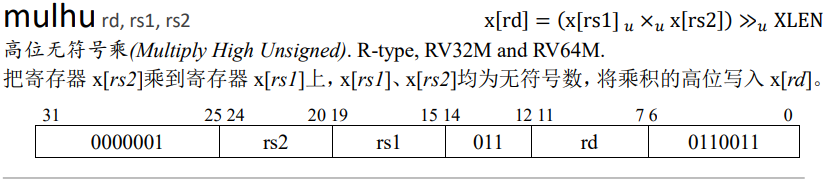
MULH:



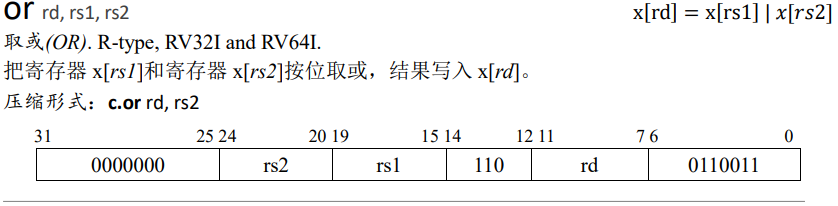
MULHSU:



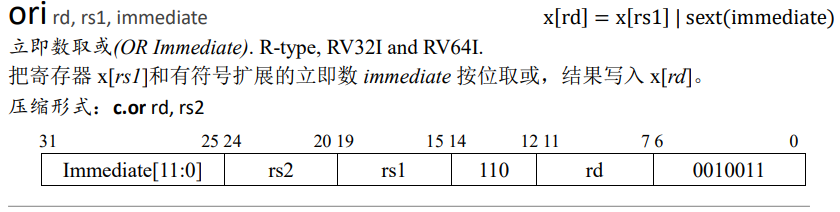
MULHU:



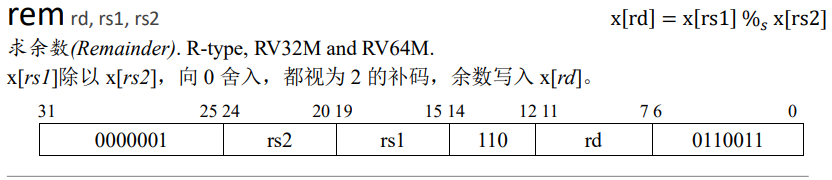
OR:



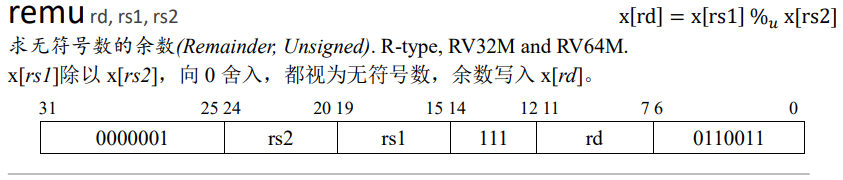
ORI:



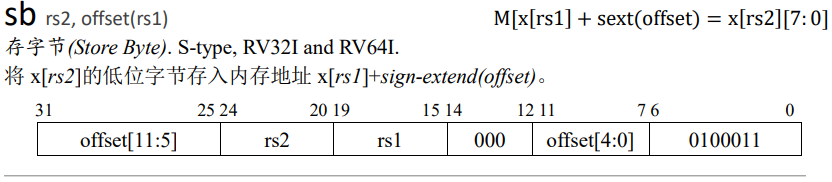
REM:



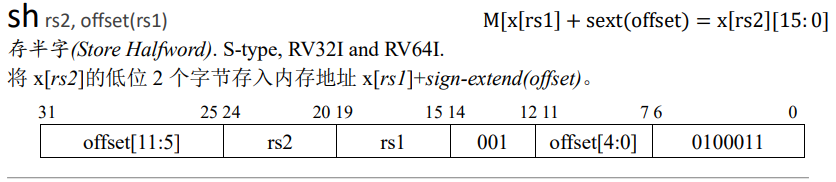
REMU:



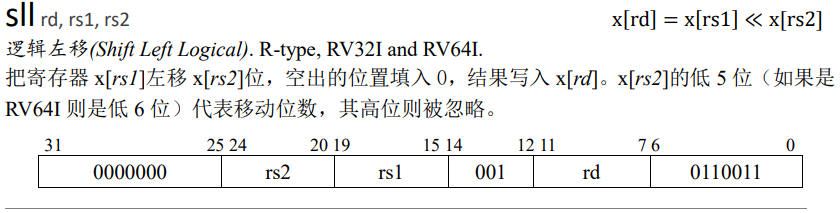
SB:



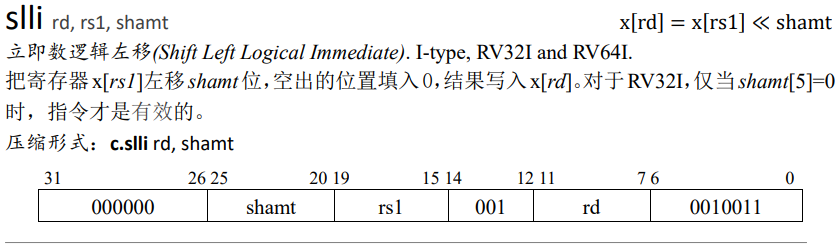
SH:



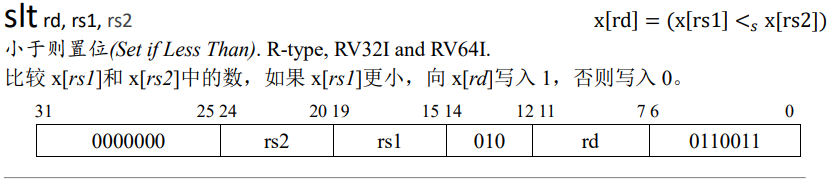
SLL:



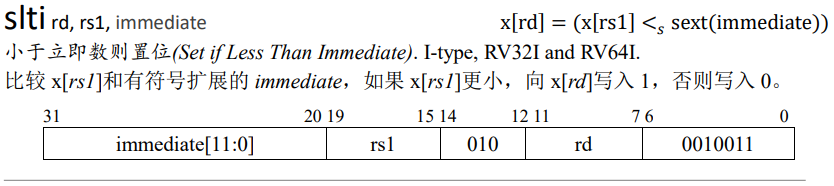
SLLI:



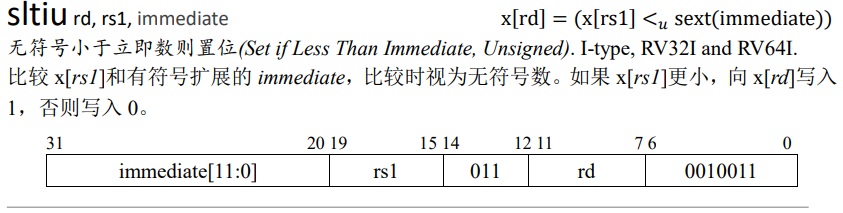
SLT:



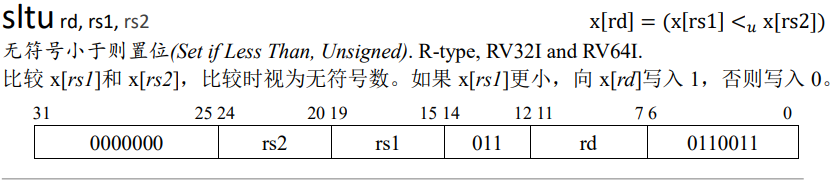
SLTI:



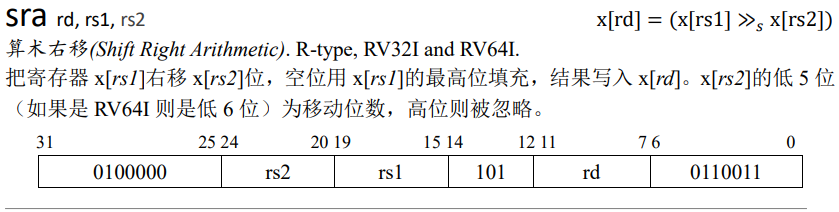
SLTIU:



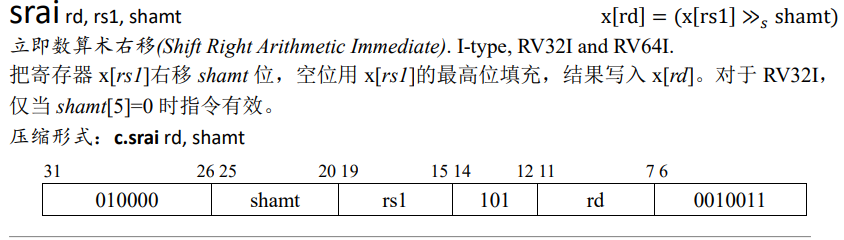
SLTU:



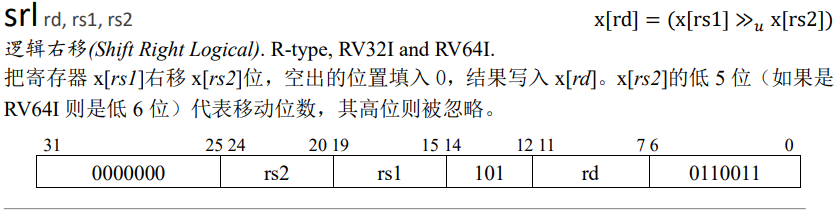
SRA:



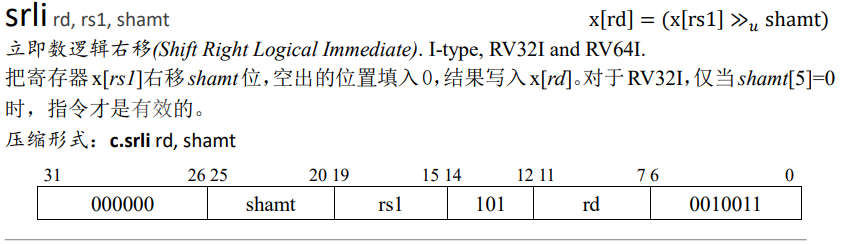
SRAI:



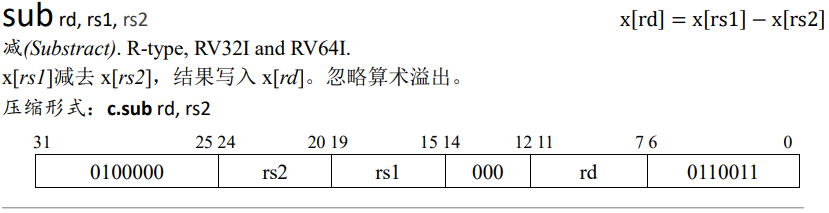
SRL:



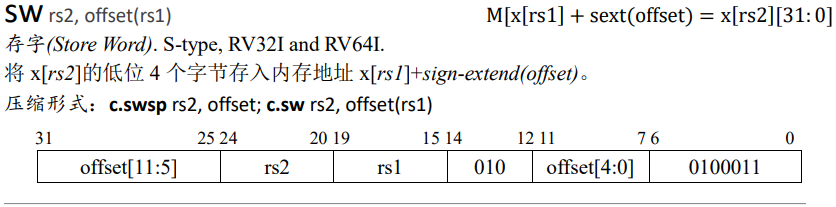
SRLI:



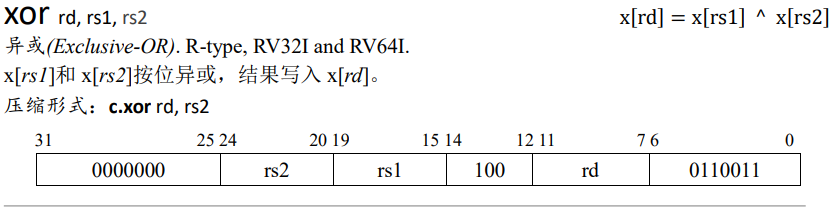
SUB:



SW:



XOR:



XORI:

