目录

图目录

表目录

“系统能力培养大赛（个人赛）&计算机系统能力等级认证”

MIPS 指令系统规范

“系统能力培养大赛（个人赛）&计算机系统能力等级认证”MIPS 指令系统是在 MIPS32 指令系统的基础之上进行一定程度地裁剪，在控制系统设计规模的前提下，保证简单系统的可实现性。概要来说，这套指令系统包含所 MIPS32 中的 34 条指令，不支持 CP0 寄存器和异常，不实现 TLB MMU 和特权等级。

本文档对竞赛所用指令系统进行具体规定。如果觉得本文档中有定义不精确之处，可以查阅参考文献[1-3]中的相关章节；如两者存在冲突，以参考文献[1-3]中的内容为准。

写在前面：

- MIPS-C1 指令集：ORI, ADDU, BNE, LW, SW;
- MIPS-C2 指令集：ORI, ADDU, BNE, LW, SW, ANDI, OR, XOR, ADDIU, BEQ, LB, SB,及以下 3 类指令中的各类随机 1 条指令：
  - 算数运算：ADD, ADDI, SUB, SLT;
  - 移位指令：SLLV, SRAV, SRA, SRLV;
  - 分支跳转：JALR, BGEZ, BLEZ, BLTZ;
- MIPS-C3 指令集：ADDU, ADDIU, MUL, AND, ANDI, LUI, OR, ORI, XOR, XORI, SLL, SRL, BEQ, BNE, BGTZ, J, JAL, JR, LB, LW, SB, SW。

本文档包含如下章节：

第 1 章，“编程模型”，对支持的数据类型、软件可见寄存器、大小尾端进行定义。

第 2 章，“操作模式”，对处理器需要支持的操作模式进行定义。

第 3 章，“指令定义”，对需实现指令逐条定义。

第 4 章，“存储管理”，定义一套线性虚实地址映射机制。

本文档供大家在本学期实验开展过程中按需索引。

每个章节对阅读者有不同的要求：

(1) 阅读第 1 章前，需要读者了解基本的 MIPS 汇编和一些计算机系统的知识。

(2) 阅读第 2 章前，需要读者理解用户态、核心态。

(3) 阅读第 3 章前，需要读者明白机器指令和汇编指令的区别，了解基本 MIPS 汇编。

(4) 阅读第 4 章前，需要读者理解虚实地址翻译的缘由和实现，了解软硬件协同。

通过本章节的学习，你将获得：

(1) 认识一个简单但比较全面的 MIPS 系统。

(2) 明确 MIPS 基础指令集的定义。

(3) 了解 MIPS 基本存储管理——固定地址映射。

1 编程模型

1.1 数据格式

处理器可处理的数据格式定义如下：

- 比特（bit, b）
- 字节（Byte, 8bits, B）
- 半字（Halfword, 16bits, H）
- 字（Word, 32bits, W）

1.2 寄存器

处理器包含的软件可见的寄存器种类如下：

- 32 个 32 位通用寄存器，r0~r31。其中有两个被赋予了特殊含义：r0，0 号通用寄存器，值永远为 0；r31，31 号通用寄存器，被 JAL 指令隐式的用作目标寄存器，存放返回地址。
- 程序计数器（PC）。这个寄存器软件无法直接访问。

1.3 大小尾端

处理器中的字节顺序和比特顺序均采用小尾端模式，即第 0 字节永远是最低有效字节，第 0 比特永远是最低有效比特。

1.4 存储访问类型

处理器至少支持不可缓存（uncached）这种存储访问类型。属于此类型的访问将直接读、写物理内存（或物理内存地址映射的寄存器），但不会访问或修改各级缓存中的内容。

2 操作模式

处理器永远处于核心态模式（Kernel Mode）。该操作模式下处理器可以执行所有指令，访问 32 位全地址空间。

3 指令定义

3.1 指令格式

所有指令长度均为 32 比特。

除个别指令外，所有指令的格式均为立即数型（I-Type）、跳转型（J-Type）和寄存器型（R-Type）三种类型中的一种。三类指令格式如图 3-1 所示。

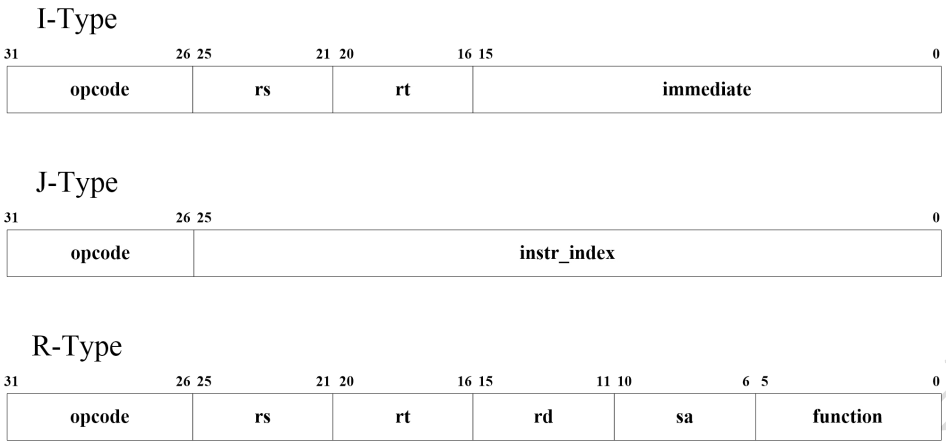


图 3-1 指令格式

3.2 指令功能分类

处理器需要实现的指令包括 7 条算术运算指令、7 条逻辑运算指令，6 条移位指令、10 条分支跳转指令、4 条访存指令，共计 34 条。下面分类给出各部分指令的简要功能介绍。

表 3-1 算术运算指令

指令名称格式	指令功能简述
ADD rd, rs, rt	加（可产生溢出例外）
ADDI rt, rs, immediate	加立即数（可产生溢出例外）
ADDU rd, rs, rt	加（不产生溢出例外）
ADDIU rt, rs, immeidate	加立即数（不产生溢出例外）
SUB rd. rs, rt	减（可产生溢出例外）
SLT rd, rs, rt	有符号小于置 1
MUL rd, rs, rt	有符号字乘

表 3-2 逻辑运算指令

指令名称格式	指令功能简述
AND rd, rs, rt	位与
ANDI rt, rs, immediate	立即数位与
LUI rt,immediate	寄存器高半部分置立即数
OR rd, rs, rt	位或
ORI rt, rs, immediate	立即数位或
XOR rd, rs, rt	位异或

指令名称格式	指令功能简述
XORI rt, rs, immediate	立即数位异或

表 3-3 移位指令

指令名称格式	指令功能简述
SLL rd, rt, sa	立即数逻辑左移
SLLV rd, rs, rt	变量逻辑左移
SRA rd, rt, sa	立即数算术右移
SRAV rd, rs, rt	变量算术右移
SRL rd, rt, sa	立即数逻辑右移
SRLV rd, rs, rt	变量逻辑右移

表 3-4 分支跳转指令

指令名称格式	指令功能简述
BEQ rs, rt, offset	相等转移
BNE rs, rt, offset	不等转移
BGEZ rs, offset	大于等于 0 转移
BGTZ rs, offset	大于 0 转移
BLEZ rs, offset	小于等于 0 转移
BLTZ rs, offset	小于 0 转移
J target	无条件直接跳转
JAL target	无条件直接跳转至子程序并保存返回地址
JR rs	无条件寄存器跳转
JALR rd, rs	无条件寄存器跳转至子程序并保存返回地址下

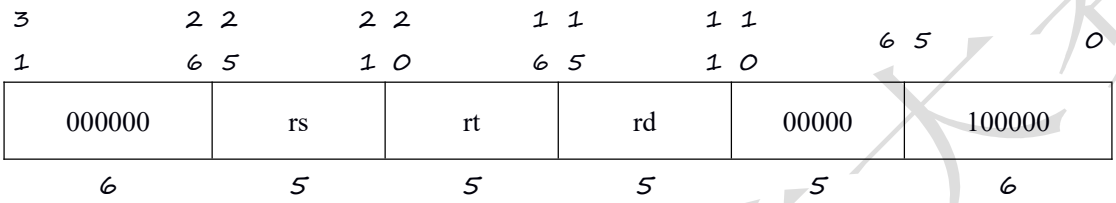
表 3-5 访存指令

指令名称格式	指令功能简述
LB rt, offset(base)	取字节有符号扩展
LW rt, offset(base)	取字

指令名称格式	指令功能简述
SB rt, offset(base)	存字节
SW rt, offset(base)	存字

3.3 算术运算指令

3.3.1 ADD



汇编格式：ADD rd, rs, rt

功能描述：将寄存器 rs 的值与寄存器 rt 的值相加，结果写入寄存器 rd 中。如果产生溢出，则触发整型溢出例外（IntegerOverflow）。

操作定义：tmp ← (GPR[rs]$_{31}$||GPR[rs]$_{31..0}$) + (GPR[rt]$_{31}$||GPR[rt]$_{31..0}$)

if tmp$_{32}$≠tmp$_{31}$ then

SignalException(IntegerOverflow)

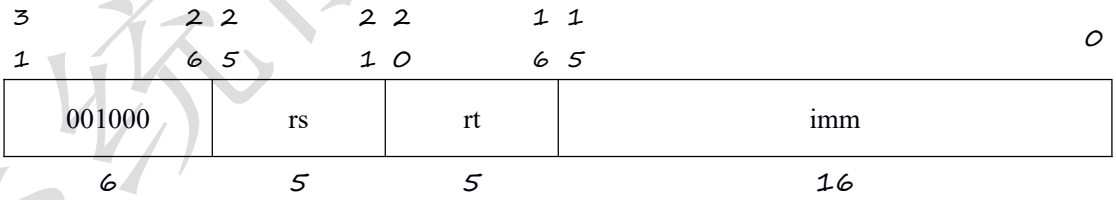
else

GPR[rd]←tmp$_{31..0}$

endif

例　外：如果有溢出，则触发整型溢出例外。

3.3.2 ADDI



汇编格式：ADDI rt, rs, imm

功能描述：将寄存器 rs 的值与有符号扩展至 32 位的立即数 imm 相加，结果写入 rt 寄存器中。如果产生溢出，则触发整型溢出例外（IntegerOverflow）。

操作定义：tmp ← (GPR[rs]$_{31}$||GPR[rs]$_{31..0}$) + sign_extend(imm)

if tmp$_{32}$≠tmp$_{31}$ then

SignalException(IntegerOverflow)

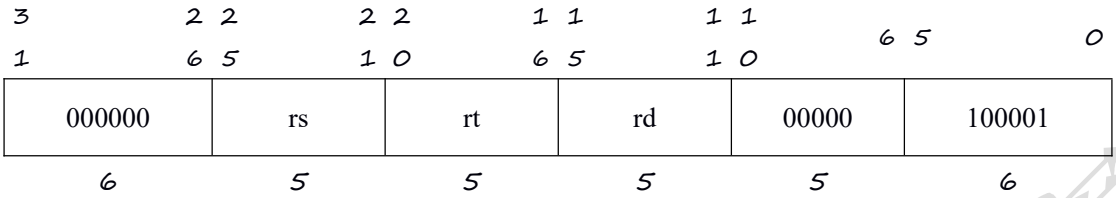
else

GPR[rt]←tmp$_{31..0}$

endif

例　外：如果有溢出，则触发整型溢出例外。

3.3.3 ADDU



汇编格式：ADDU rd, rs, rt

功能描述：将寄存器 rs 的值与寄存器 rt 的值相加，结果写入 rd 寄存器中。

操作定义：GPR[rd] ← GPR[rs] + GPR[rt]

例　外：无

3.3.4 ADDIU

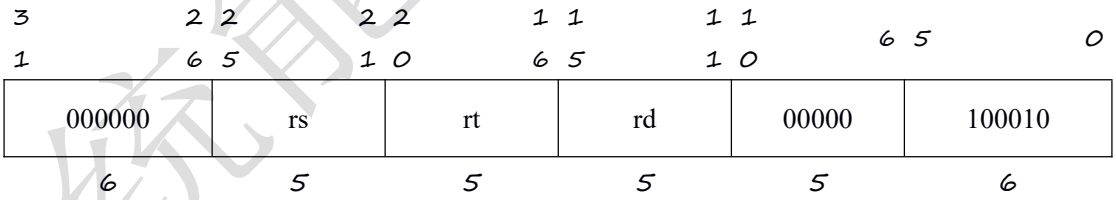
汇编格式：ADDIU rt, rs, imm

功能描述：将寄存器 rs 的值与**有符号扩展**至 32 位的立即数 imm 相加，结果写入 rt 寄存器中。

操作定义：GPR[rt] ← GPR[rs] + sign_extend(imm)

例　外：无

3.3.5 SUB



汇编格式：SUB rd, rs, rt

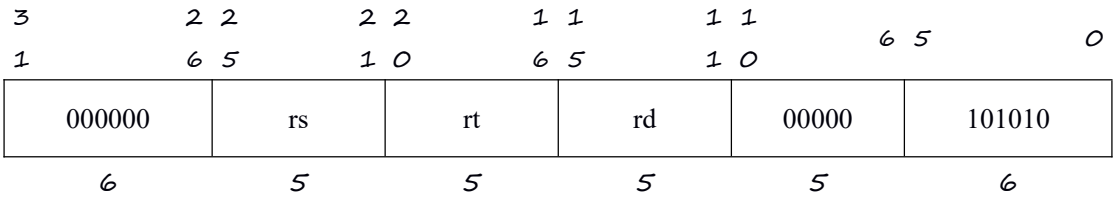
功能描述：将寄存器 rs 的值与寄存器 rt 的值相减，结果写入 rd 寄存器中。如果产生溢出，则触发整型溢出例外（IntegerOverflow）。

操作定义：tmp ← (GPR[rs]$_{31}$||GPR[rs]$_{31..0}$) - (GPR[rt]$_{31}$||GPR[rt]$_{31..0}$)

```
if tmp32≠tmp31 then
    SignalException(IntegerOverflow)
else
    GPR[rd]←tmp31..0
endif
```

例　外：如果有溢出，则触发整型溢出例外。

3.3.6 SLT



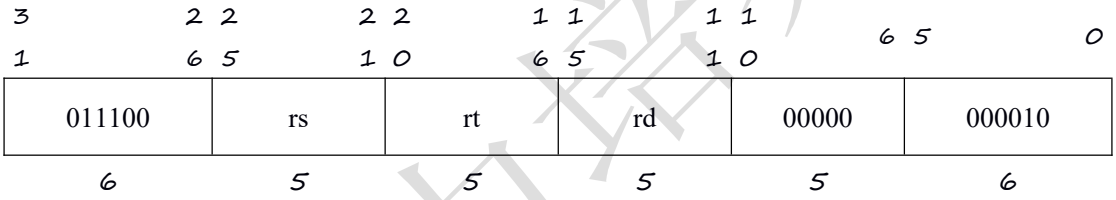
汇编格式：SLT rd, rt, rs

功能描述：将寄存器 rs 的值与寄存器 rt 中的值进行有符号数比较，如果寄存器 rs 中的值小，则寄存器 rd 置 1；否则寄存器 rd 置 0。

操作定义：
```
if GPR[rs] < GPR[rt] then
    GPR[rd] ← 1
else
    GPR[rd] ← 0
endif
```

例　外：无

3.3.7 MUL



汇编格式：MUL rd, rs, rt

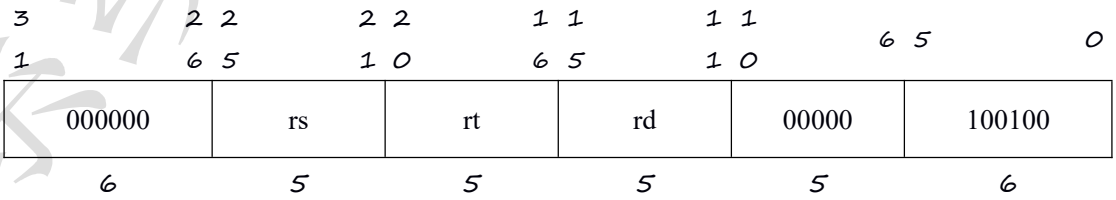
功能描述：有符号乘法，寄存器 rs 的值乘以寄存器 rt 的值，乘积的低 32 位写入 rd 寄存器。

操作定义：$GPR[rd] \leftarrow GPR[rs]_{31..0} \times GPR[rt]_{31..0}$

例　外：无

3.4 逻辑运算指令

3.4.1 AND



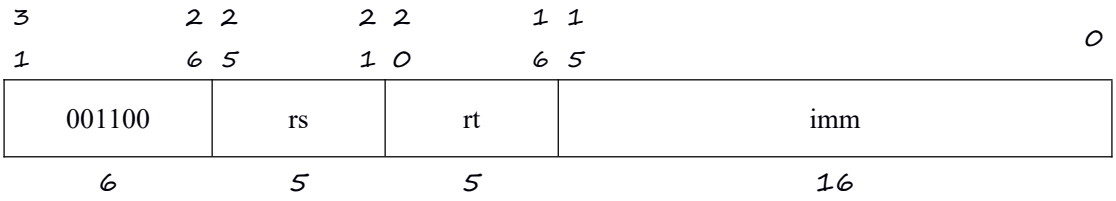
汇编格式：AND rd, rs, rt

功能描述：寄存器 rs 中的值与寄存器 rt 中的值按位逻辑与，结果写入寄存器 rd 中。

操作定义：GPR[rd] ← GPR[rs] & GPR[rt]

例　外：无

3.4.2 ANDI



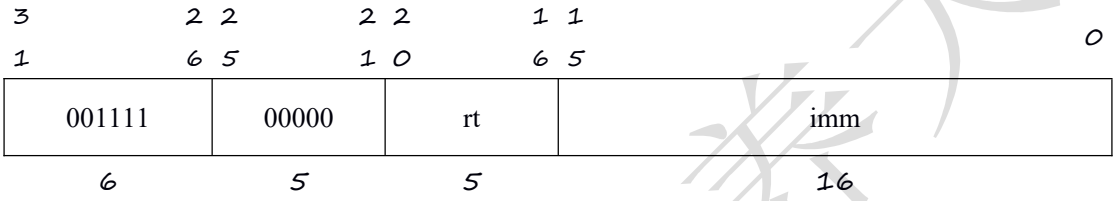
汇编格式：ANDI rt, rs, imm

功能描述：寄存器 rs 中的值与 0 扩展至 32 位的立即数 imm 按位逻辑与，结果写入寄存器 rt 中。

操作定义：GPR[rt] ← GPR[rs] and Zero_extend(imm)

例　外：无

3.4.3 LUI



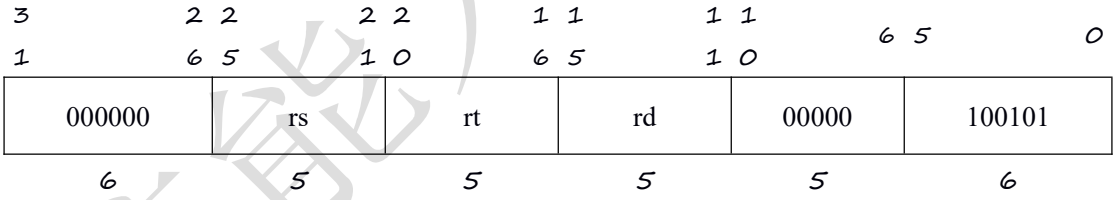
汇编格式：LUI rt, imm

功能描述：将 16 位立即数 imm 写入寄存器 rt 的高 16 位，寄存器 rt 的低 16 位置 0。

操作定义：GPR[rt] ← (imm || 0^{16})

例　外：无

3.4.4 OR



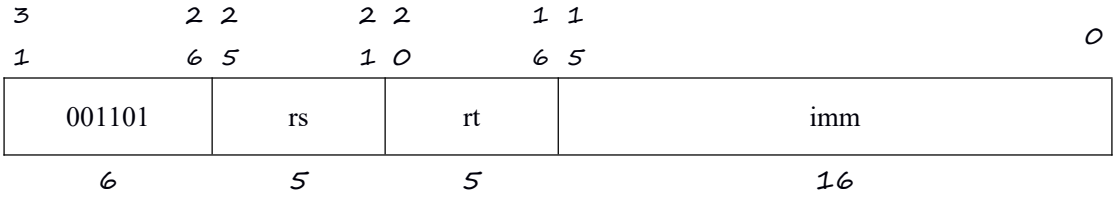
汇编格式：OR rd, rs, rt

功能描述：寄存器 rs 中的值与寄存器 rt 中的值按位逻辑或，结果写入寄存器 rd 中。

操作定义：GPR[rd] ← GPR[rs] or GPR[rt]

例　外：无

3.4.5 ORI



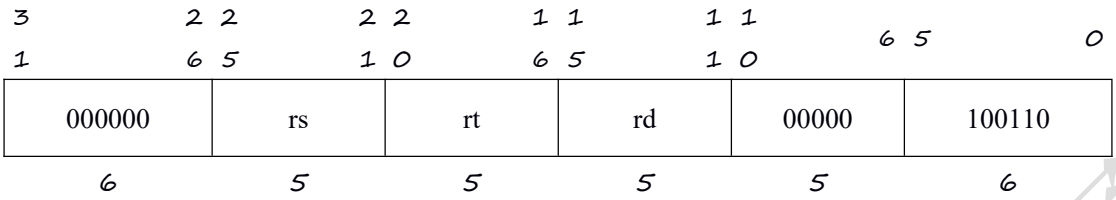
汇编格式：ORI rt, rs, imm

功能描述：寄存器 rs 中的值与 0 扩展至 32 位的立即数 imm 按位逻辑或，结果写入寄存器 rt 中。

操作定义：GPR[rt] ← GPR[rs] or Zero_extend(imm)

例　外：无

3.4.6 XOR



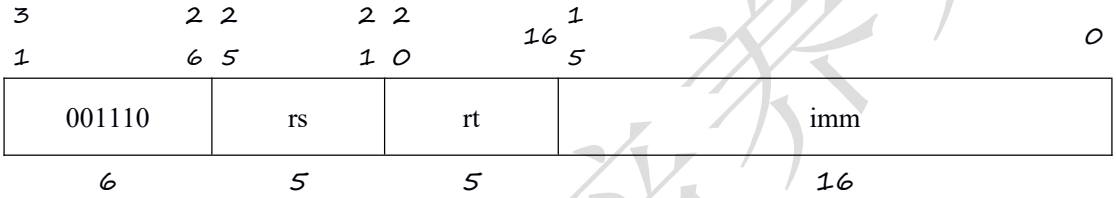
汇编格式：XOR rd, rs, rt

功能描述：寄存器 rs 中的值与寄存器 rt 中的值按位逻辑异或，结果写入寄存器 rd 中。

操作定义：GPR[rd] ← GPR[rs] xor GPR[rt]

例　外：无

3.4.7 XORI



汇编格式：XORI rt, rs, imm

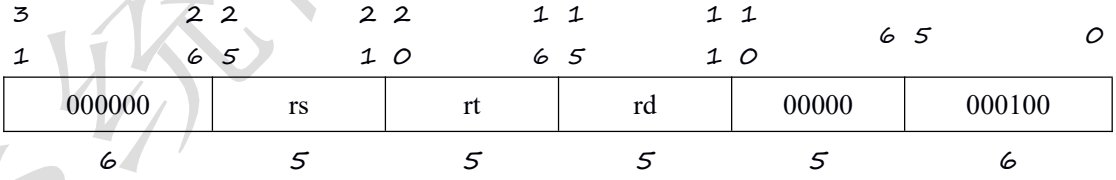
功能描述：寄存器 rs 中的值与 0 扩展至 32 位的立即数 imm 按位逻辑异或，结果写入寄存器 rt 中。

操作定义：GPR[rt] ← GPR[rs] xor Zero_extend(imm)

例　外：无

3.5 移位指令

3.5.1 SLLV



汇编格式：SLLV rd, rt, rs

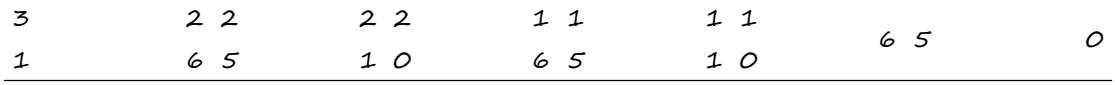
功能描述：由寄存器 rs 中的值指定移位量，对寄存器 rt 的值进行逻辑左移，结果写入寄存器 rd 中。

操作定义：s ← $GPR[rs]_{4..0}$

$GPR[rd] \leftarrow GPR[rt]_{(31-s)..0} || 0^s$

例　外：无

3.5.2 SLL



000000	00000	rt	rd	sa	000000
6	5	5	5	5	6

汇编格式：SLL rd, rt, sa

功能描述：由立即数 sa 指定移位量，对寄存器 rt 的值进行逻辑左移，结果写入寄存器 rd 中。

操作定义：s ← sa

GPR[rd] ← GPR[rt]$_{(31-s)..0}$||0^s

例　外：无

3.5.3 SRAV

31	26	25	21	20	16	15	11	10	6	5	0
000000		rs		rt		rd		00000		000111	
6		5		5		5		5		6	

汇编格式：SRAV rd, rt, rs

功能描述：由寄存器 rs 中的值指定移位量，对寄存器 rt 的值进行算术右移，结果写入寄存器 rd 中。

操作定义：s ← GPR[rs]$_{4..0}$

GPR[rd] ← (GPR[rt]$_{31}$)^s || GPR[rt]$_{31..s}$

例　外：无

3.5.4 SRA

31	26	25	21	20	16	15	11	10	6	5	0
000000		00000		rt		rd		sa		000011	
6		5		5		5		5		6	

汇编格式：SRA rd, rt, sa

功能描述：由立即数 sa 指定移位量，对寄存器 rt 的值进行算术右移，结果写入寄存器 rd 中。

操作定义：s ← sa

GPR[rd] ← (GPR[rt]$_{31}$)^s || GPR[rt]$_{31..s}$

例　外：无

3.5.5 SRLV

31	26	25	21	20	16	15	11	10	6	5	0
000000		rs		rt		rd		00000		000110	
6		5		5		5		5		6	

汇编格式：SRLV rd, rt, rs

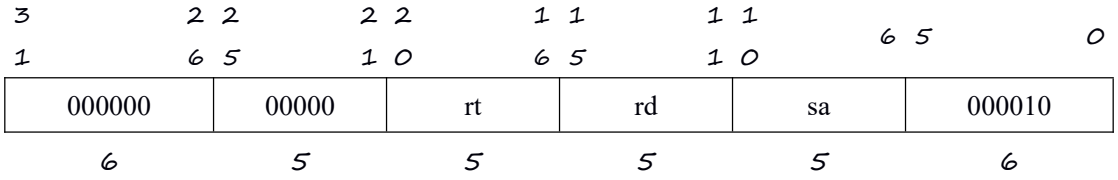
功能描述：由寄存器 rs 中的值指定移位量，对寄存器 rt 的值进行逻辑右移，结果写入寄存器 rd 中。

操作定义：s ← GPR[rs]$_{4..0}$

GPR[rd] ← 0^s || GPR[rt]$_{31..s}$

例　外：无

3.5.6 SRL



汇编格式：SRL rd, rt, sa

功能描述：由立即数 sa 指定移位量，对寄存器 rt 的值进行逻辑右移，结果写入寄存器 rd 中。

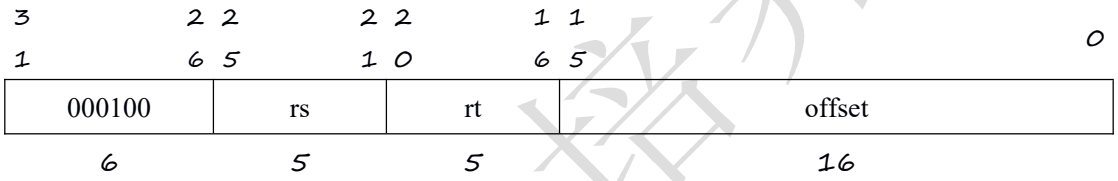
操作定义：s ← sa

GPR[rd] ← 0^s || GPR[rt]$_{31..s}$

例　外：无

3.6 分支跳转指令

3.6.1 BEQ



汇编格式：BEQ rs, rt, offset

功能描述：如果寄存器 rs 的值等于寄存器 rt 的值则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I:　　condition ← GPR[rs] = GPR[rt]

target_offset ← Sign_extend(offset||0^2)

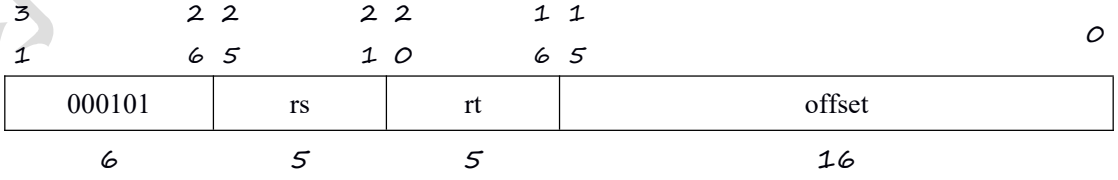
I+1:　if condition then

PC ← PC+ target_offset

endif

例　外：无

3.6.2 BNE



汇编格式：BNE rs, offset

功能描述：如果寄存器 rs 的值不等于寄存器 rt 的值则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I:　　condition ← GPR[rs] ≠ GPR[rt]

target_offset ← Sign_extend(offset||0^2)

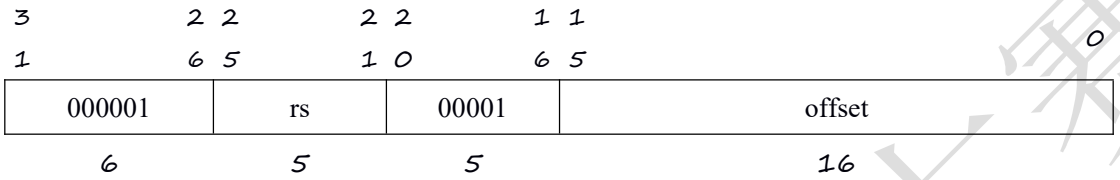
I+1: if condition then

PC ← PC+ target_offset

endif

例　外：无

3.6.3 BGEZ



汇编格式：BGEZ rs, offset

功能描述：如果寄存器 rs 的值大于等于 0 则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I: condition ← GPR[rs] ≥ 0

target_offset ← Sign_extend(offset||0^2)

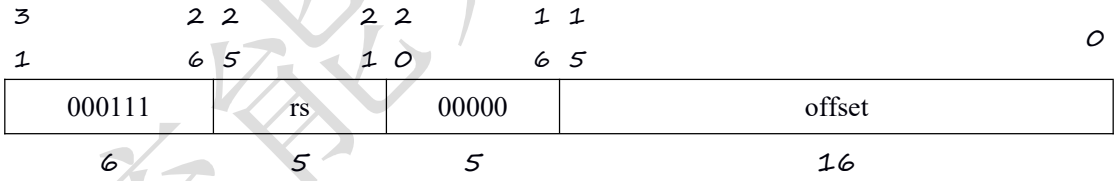
I+1: if condition then

PC ← PC+ target_offset

endif

例　外：无

3.6.4 BGTZ



汇编格式：BGTZ rs, offset

功能描述：如果寄存器 rs 的值大于 0 则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I: condition ← GPR[rs] > 0

target_offset ← Sign_extend(offset||0^2)

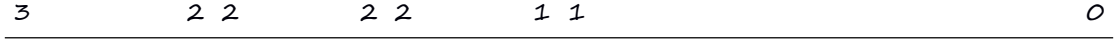
I+1: if condition then

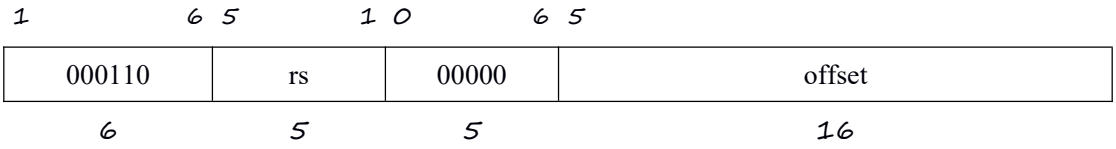
PC ← PC+ target_offset

endif

例　外：无

3.6.5 BLEZ





汇编格式：BLEZ rs, offset

功能描述：如果寄存器 rs 的值小于等于 0 则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I: condition ← GPR[rs] ≤ 0

target_offset ← Sign_extend(offset||0^2)

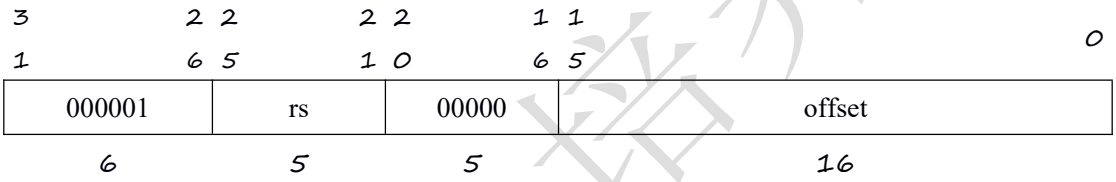
I+1: if condition then

PC ← PC+ target_offset

endif

例　外：无

3.6.6 BLTZ



汇编格式：BLTZ rs, offset

功能描述：如果寄存器 rs 的值小于 0 则转移，否则顺序执行。转移目标由立即数 offset 左移 2 位并进行有符号扩展的值加上该分支指令对应的延迟槽指令的 PC 计算得到。

操作定义：I: condition ← GPR[rs] < 0

target_offset ← Sign_extend(offset||0^2)

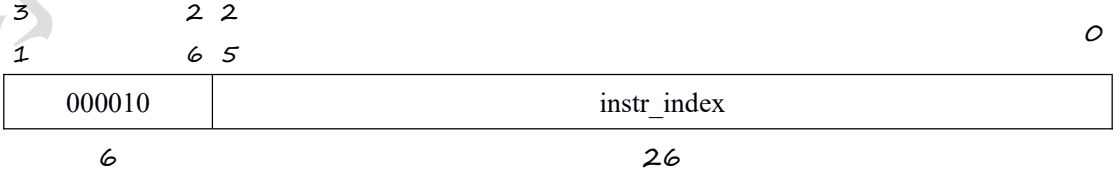
I+1: if condition then

PC ← PC+ target_offset

endif

例　外：无

3.6.7 J



汇编格式：J target

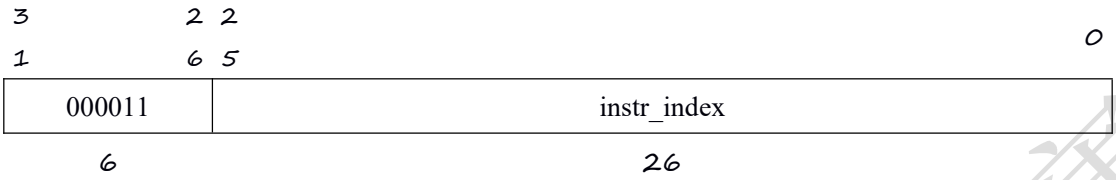
功能描述：无条件跳转。跳转目标由该分支指令对应的延迟槽指令的 PC 的最高 4 位与立即数 instr_index 左移 2 位后的值拼接得到。

操作定义：I:

I+1: PC ← $PC_{31..28}$ || instr_index || 0^2

例　外：无

3.6.8 JAL



汇编格式：JAL target

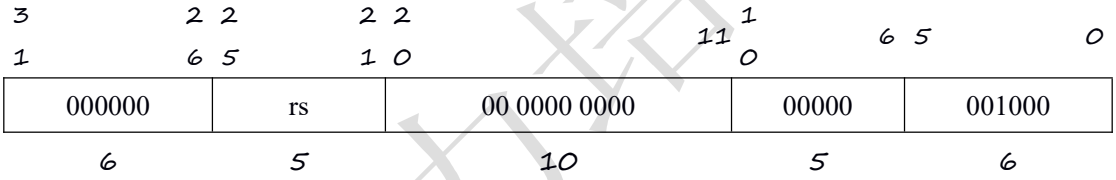
功能描述：无条件跳转。跳转目标由该分支指令对应的延迟槽指令的 PC 的最高 4 位与立即数 instr_index 左移 2 位后的值拼接得到。同时将该分支对应延迟槽指令之后的指令的 PC 值保存至第 31 号通用寄存器中。

操作定义：I: GPR[31] ← PC + 8

I+1: PC ← $PC_{31..28}$ || instr_index || 0^2

例　外：无

3.6.9 JR



汇编格式：JR rs

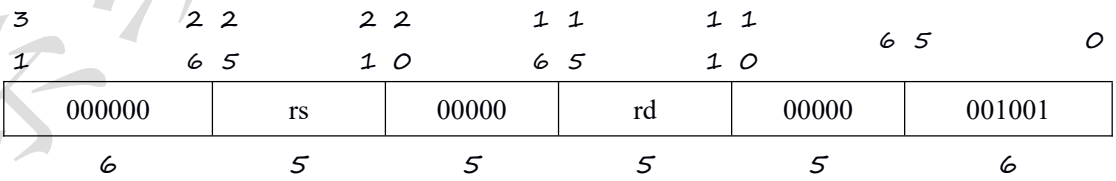
功能描述：无条件跳转。跳转目标为寄存器 rs 中的值。

操作定义：I: temp ← GPR[rs]

I+1: PC ← temp

例　外：无

3.6.10 JALR



汇编格式：JALR rd, rs

JALR rs (rd=31 implied)

功能描述：无条件跳转。跳转目标为寄存器 rs 中的值。同时将该分支对应延迟槽指令之后的指令的 PC 值保存至寄存器 rd 中。

操作定义：I： temp← GPR[rs]

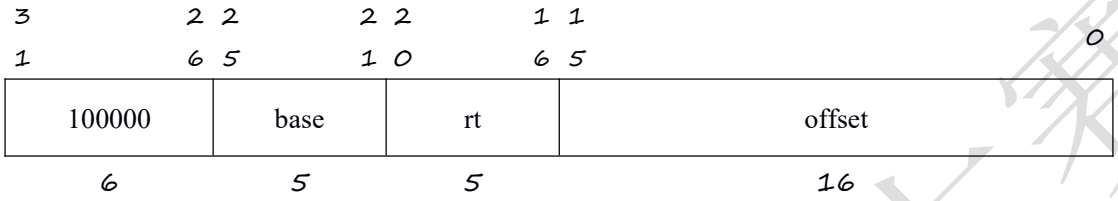
GPR[rd] ← PC + 8

I+1: PC ← temp

例　外：无

3.7 访存指令

3.7.1 LB



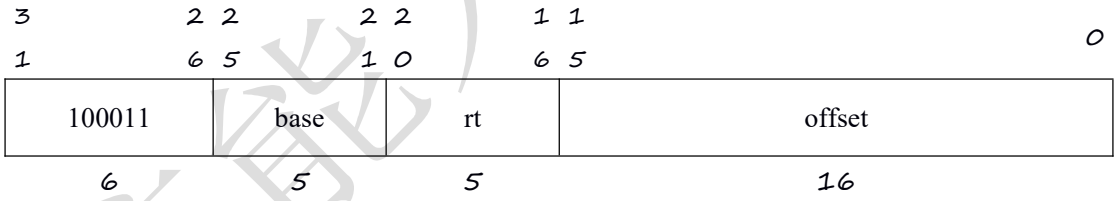
汇编格式：LB rt, offset(base)

功能描述：将 base 寄存器的值加上符号扩展后的立即数 offset 得到访存的虚地址，据此虚地址从存储器中读取 1 个字节的值并进行符号扩展，写入到 rt 寄存器中。

操作定义：vAddr ← GPR[base] + sign_extend(offset)
(pAddr, CCA) ← AddressTranslation(vAddr, DATA, LOAD)
membyte ← LoadMemory(CCA, BYTE, pAddr, vAddr, DATA)
GPR[rt] ← sign_extend($membyte_{7..0}$)

例　外：无

3.7.2 LW



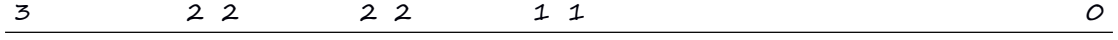
汇编格式：LW rt, offset(base)

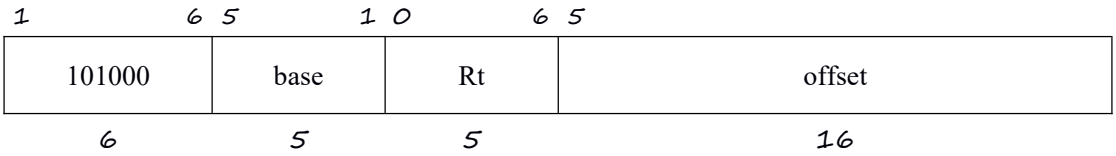
功能描述：将 base 寄存器的值加上符号扩展后的立即数 offset 得到访存的虚地址，如果地址不是 4 的整数倍则触发地址错例外，否则据此虚地址从存储器中读取连续 4 个字节的值，写入到 rt 寄存器中。

操作定义：vAddr ← GPR[base] + sign_extend(offset)
if $vAddr_{1..0} \neq 0$ then
　　SignalException(AddressError)
endif
(pAddr, CCA) ← AddressTranslation(vAddr, DATA, LOAD)
memword ← LoadMemory(CCA, WORD, pAddr, vAddr, DATA)
GPR[rt] ← sign_extend($memword_{31..0}$)

例　外：地址最低 2 位不为 0，触发地址错例外

3.7.3 SB





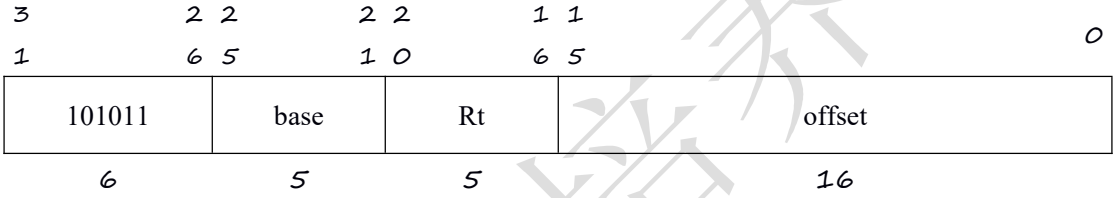
汇编格式：SB rt, offset(base)

功能描述：将 base 寄存器的值加上符号扩展后的立即数 offset 得到访存的虚地址，据此虚地址将 rt 寄存器的最低字节存入存储器中。

操作定义：vAddr ← GPR[base] + sign_extend(offset)
(pAddr, CCA) ← AddressTranslation(vAddr, DATA, STORE)
databyte ← $GPR[rt]_{7..0}$
StoreMemory(CCA, BYTE, databyte, pAddr, vAddr, DATA)

例　外：无

3.7.4 SW



汇编格式：SW rt, offset(base)

功能描述：将 base 寄存器的值加上符号扩展后的立即数 offset 得到访存的虚地址，如果地址不是 4 的整数倍则触发地址错例外，否则据此虚地址将 rt 寄存器存入存储器中。

操作定义：vAddr ← GPR[base] + sign_extend(offset)
if $vAddr_{1..0} \neq 0$ then
　　SignalException(AddressError)
endif
(pAddr, CCA) ← AddressTranslation(vAddr, DATA, STORE)
dataword ← $GPR[rt]_{31..0}$
StoreMemory(CCA, WORD, dataword, pAddr, vAddr, DATA)

例　外：地址最低 2 位不为 0，触发地址错例外

4 存储管理

本指令系统对操作模式和存储访问类型进行了严格的限定，所以存储管理部分仅对虚实地址映射方式进行定义。

处理器可访问 32 位虚地址空间。整个 4GB 大小的虚地址空间被划分成 5 个段。处理器采用固定地址映射（FMT）机制，在进行虚实地址映射时，不同段的虚地址以线性方式固定地映射到一段连续的物理地址上。具体映射方式如图 4-1 所示。

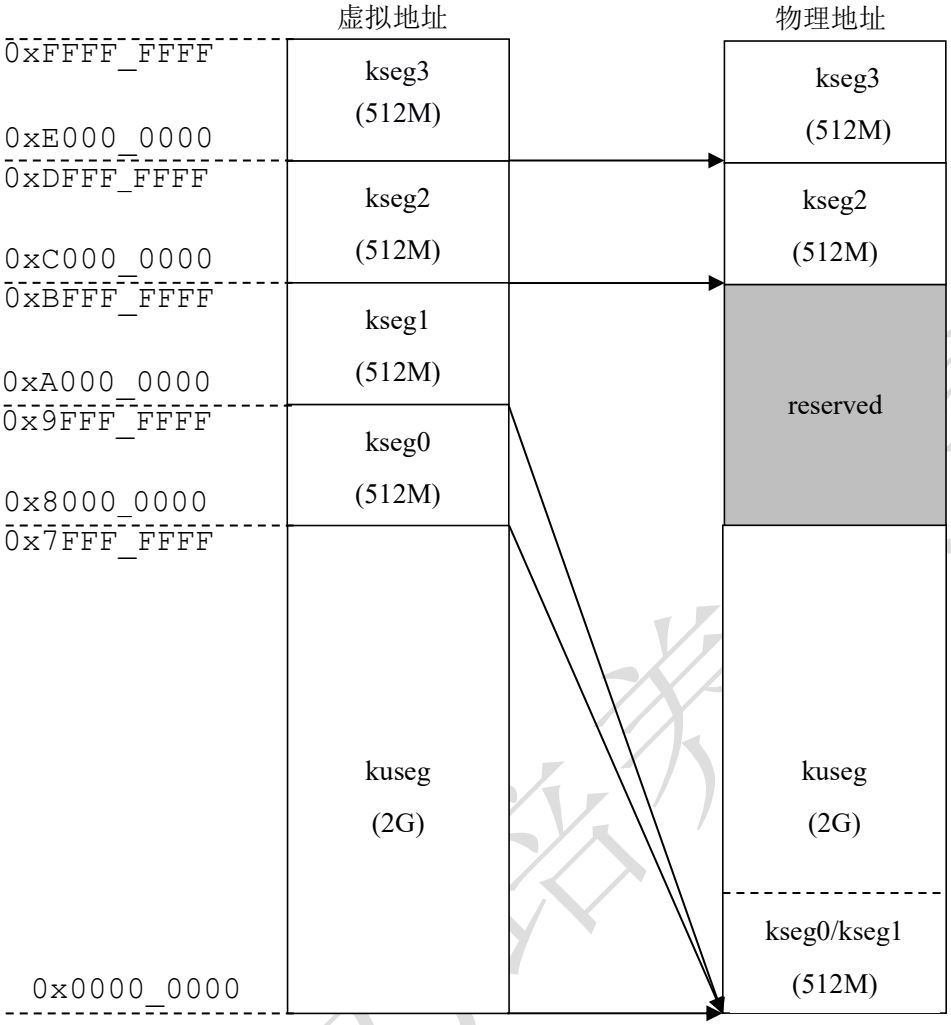


图 4-1 虚实地址映射关系图

参考文献

[1] MIPS Architecture For Programmers Volume I-A: Introduction to the MIPS32 Architecture. Revision 3.02.

[2] MIPS Architecture For Programmers Volume II-A: The MIPS32 Instruction Set. Revision 3.02.

[3] MIPS Architecture For Programmers Vol. III: MIPS32/microMIPS32 Privileged Resource Architecture. Revision 3.02.

[4] See MIPS Run Linux, 2nd Edition, 2006, Morgan Kaufmann, Dominic Sweetman