3.5 MIPS32指令架构的CPU设计实例

MIPS（Micro-processor without interlocked piped stages），一种无内部互锁流水级微处理器。

--1980s，斯坦福Hennessy；

--RISC型R系列工业处理器；

--MIPS-I~V，MIPS-16/32/64；

3.5.1 MIPS32的指令

总体情况：

(1) 存储器按字节编址

(2) 可用寄存器32个，宽度32位

(3) RISC架构

结合高级语言编程，考虑处理器应该有哪些类型的指令？

运算？　访存？　转移？　…

※可提供的寄存器列表

寄存器名	地址编号		用途说明
$s0	0	√	保存固定的常数0
$at	1	√	汇编器专用
$v0～$v1	2～3	√	表达式计算或函数调用的返回结果
$a0～$a3	4～7	√	函数调用参数1～3
$t0～$t7	8～15	√	临时变量，函数调用时不需要保存和恢复
$s0～$s7	16～23	√	函数调用时需要保存和恢复的寄存器变量
$t8～$t9	24～25	√	临时变量，函数调用时不需要保存和恢复
$k0～$k1	26～27	√	操作系统专用
$gp	28	√	全局指针变量(Global Pointer)
$sp	29	√	堆栈指针变量(Stack Pointer)
$fp	30	√	帧指针变量(Frame Pointer)
$ra	31	√	返回地址(Return Address)

1. 指令格式与指令集

指令字长固定为32位，寄存器型寻址，指令中给出寄存器号。

指令类型	指令长度（32位定长）					
	31 ~ 26	25~21	20~16	15~11	10 ~ 6	5 ~ 0
R型	op(6)	rs(5)	rt(5)	rd(5)	shamt	func
I型	op(6)	rs(5)	rt(5)	imm (16)		
J型	op(6)	address(26)				

■ R型指令(Register)

指令长度（32位定长）					
31 ~ 26	25~21	20~16	15~11	10 ~ 6	5 ~ 0
op(6)	rs(5)	rt(5)	rd(5)	sa	func

操作数和保存结果均通过寄存器进行；

- op：操作码，所有R型指令中都全为0；
- rs：寄存器编号，对应第1个源操作数；
- rt：寄存器编号，对应第2个源操作数；
- rd：寄存器编号，据此保存结果；
- sa：常数，在移位指令中使用；
- func：功能码，指定指令的具体功能；

典型指令集见表3-22

■ I型指令(Immediate)

指令长度（32位定长）					
31 ~ 26	25~21	20~16	15~11	10 ~ 6	5 ~ 0
op(6)	rs(5)	rt(5)	imm (16)		

操作数中涉及立即数，结果保存到寄存器；

- op：标识指令的操作功能；
- rs：第1个源操作数，是寄存器操作数；
- rt：目的寄存器编号，用来保存运算结果；
- imm：第2个源操作数，立即数；

典型指令集见表3-23

■ J型指令(Jump)

指令长度（32位定长）					
31 ~ 26	25~21	20~16	15~11	10 ~ 6	5 ~ 0
op(6)	address(26)				

实现无条件转移；

- op：确定指令的功能；
- address：转移目标地址的偏移量字段；

典型指令集见表3-24

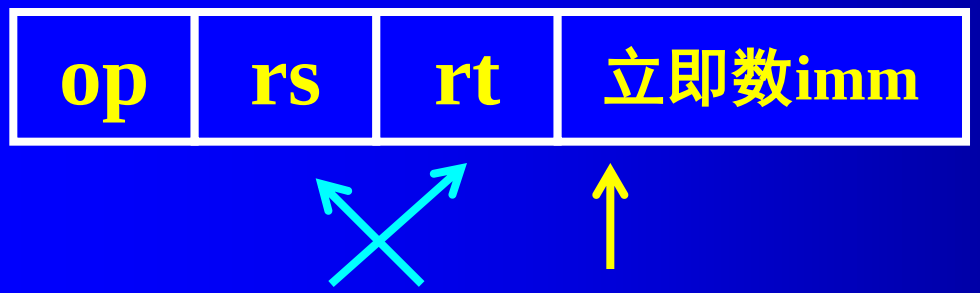
2. 寻址方式

在MIPS32指令集中，不会单设寻址方式说明字段，通过op字段和func字段(针对R型指令)隐含说明。

R型指令：	由op和func字段共同隐含说明当前的寻址方式。
I型和J型指令：	由op字段隐含说明当前指令使用的寻址方式。

- 立即数寻址(Immediate addressing)

操作数在指令中的立即数字段。



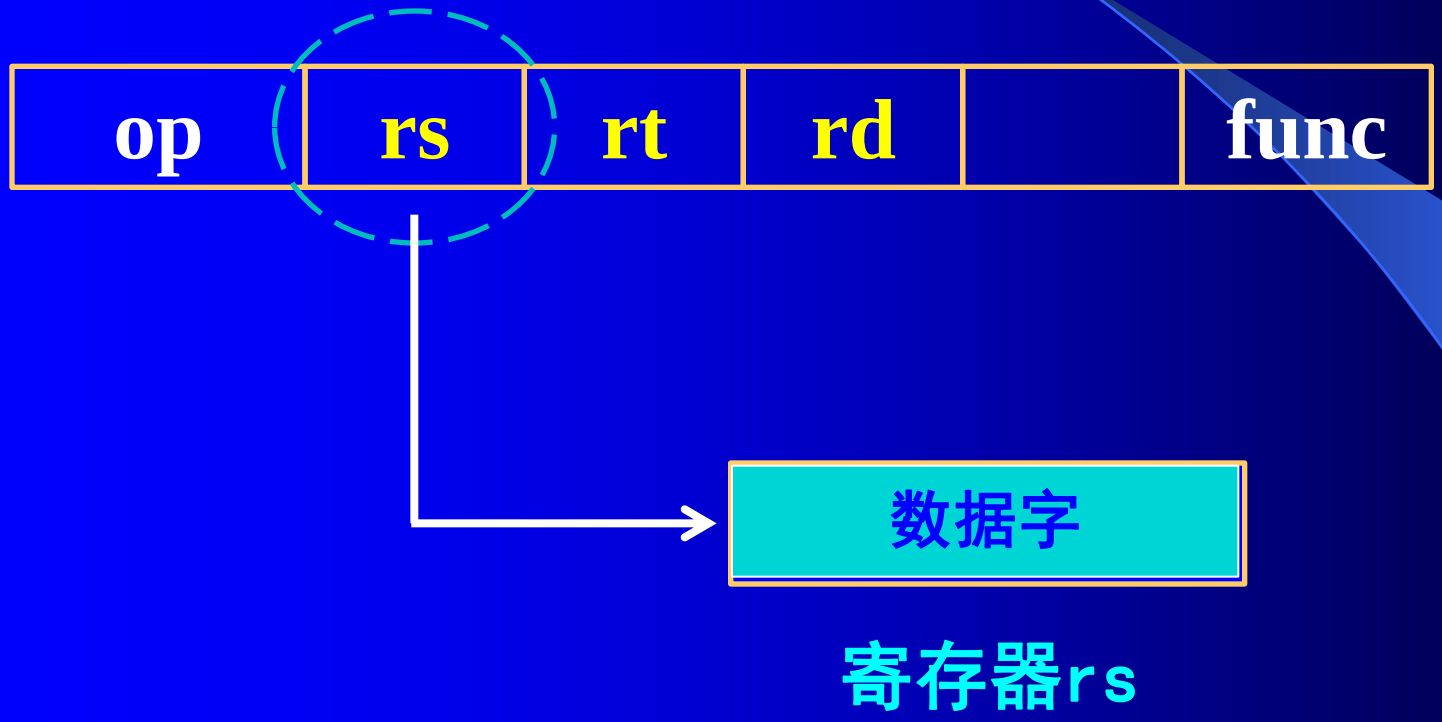
addi s1, s2, 10

$s2+10→$s1

注意：汇编格式和编码格式段的对应关系。

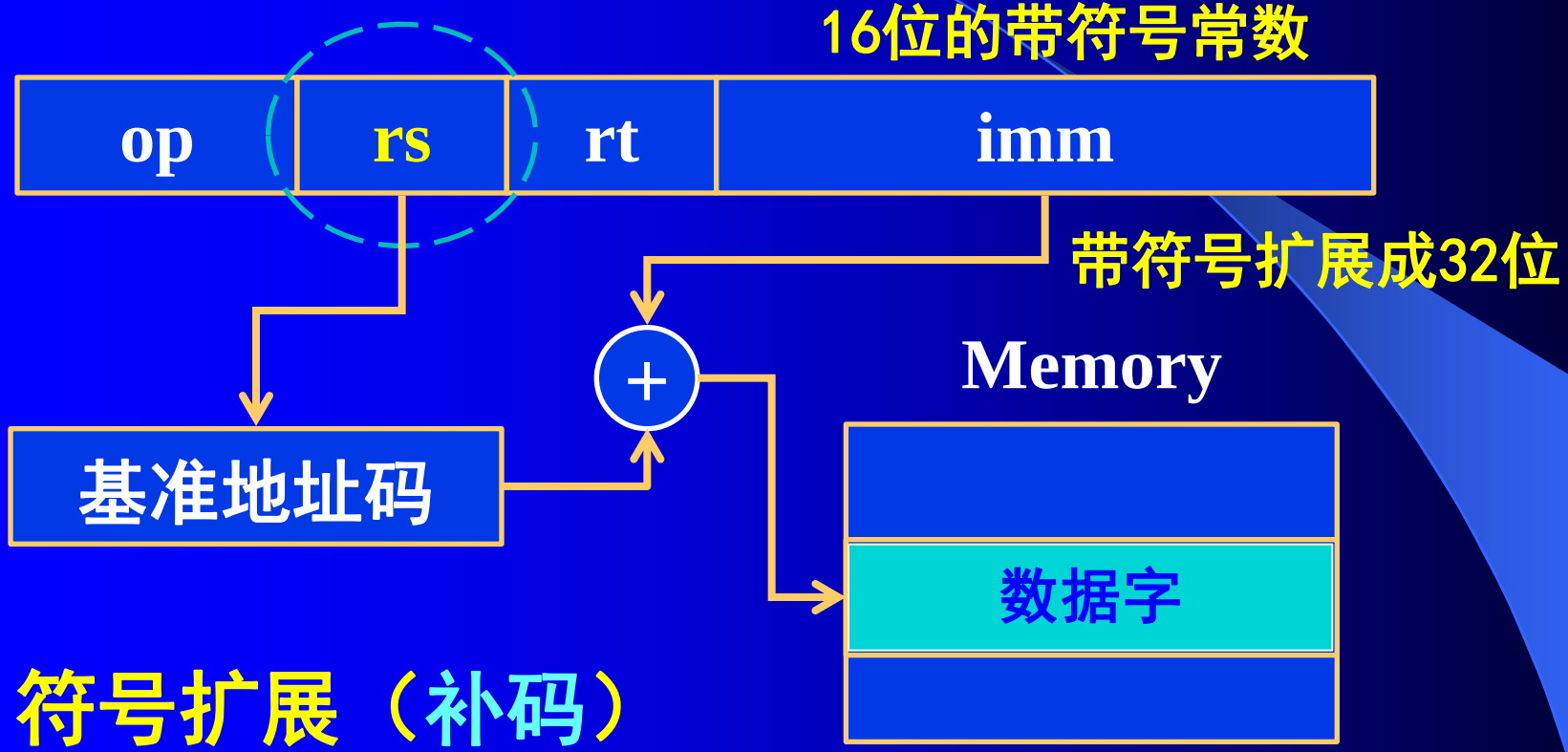
● 寄存器直接寻址(Register Addressing)

操作数直接在寄存器中。



● 基址寻址(Basic Addressing)

操作数由寄存器和立即数字段联合产生



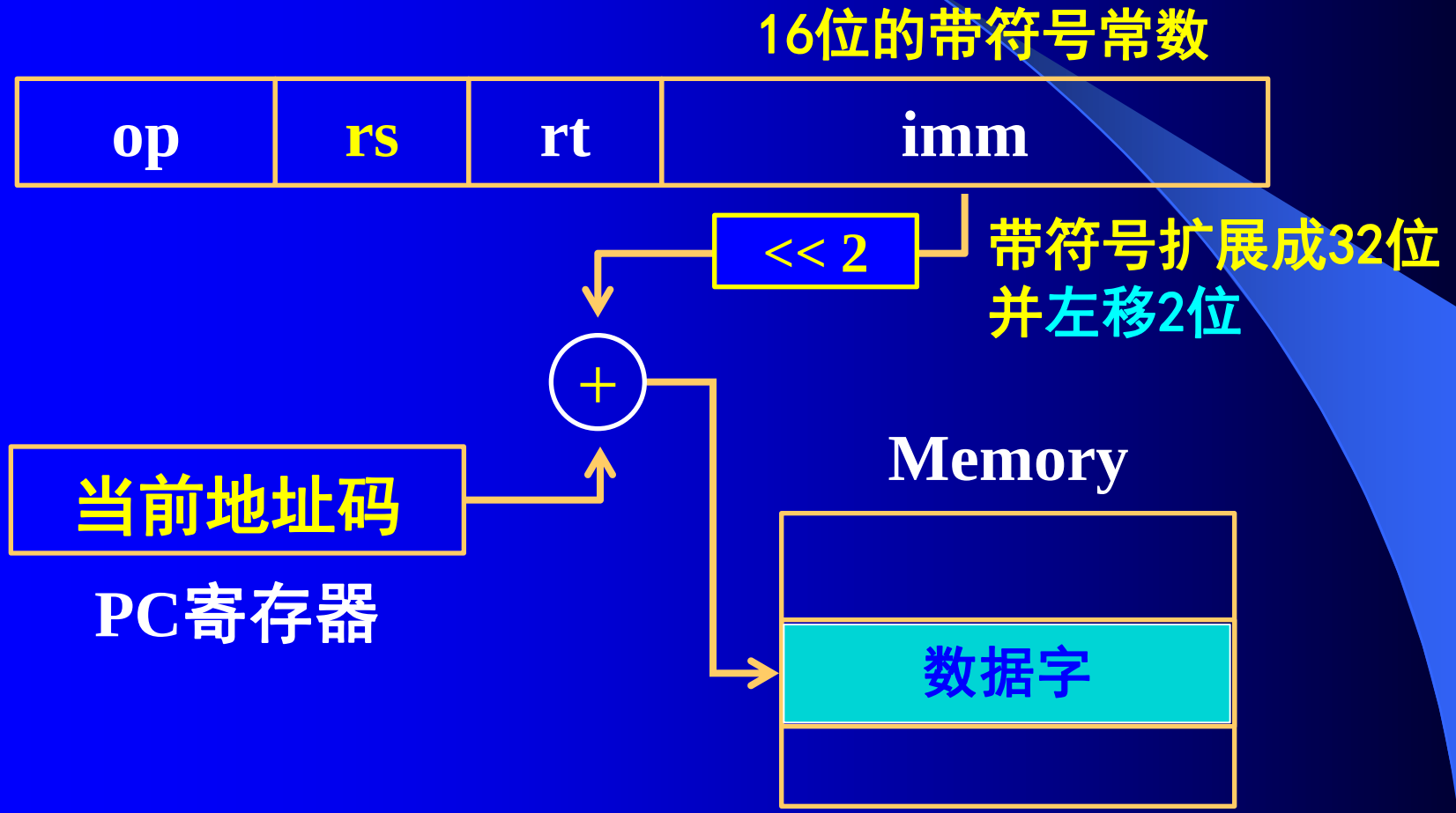
符号扩展（补码）

正数，在高位补上16个0；$012A_H \rightarrow \underline{0000}012A_H$

负数，在高位补上16个1；$812A_H \rightarrow \underline{1111}812A_H$

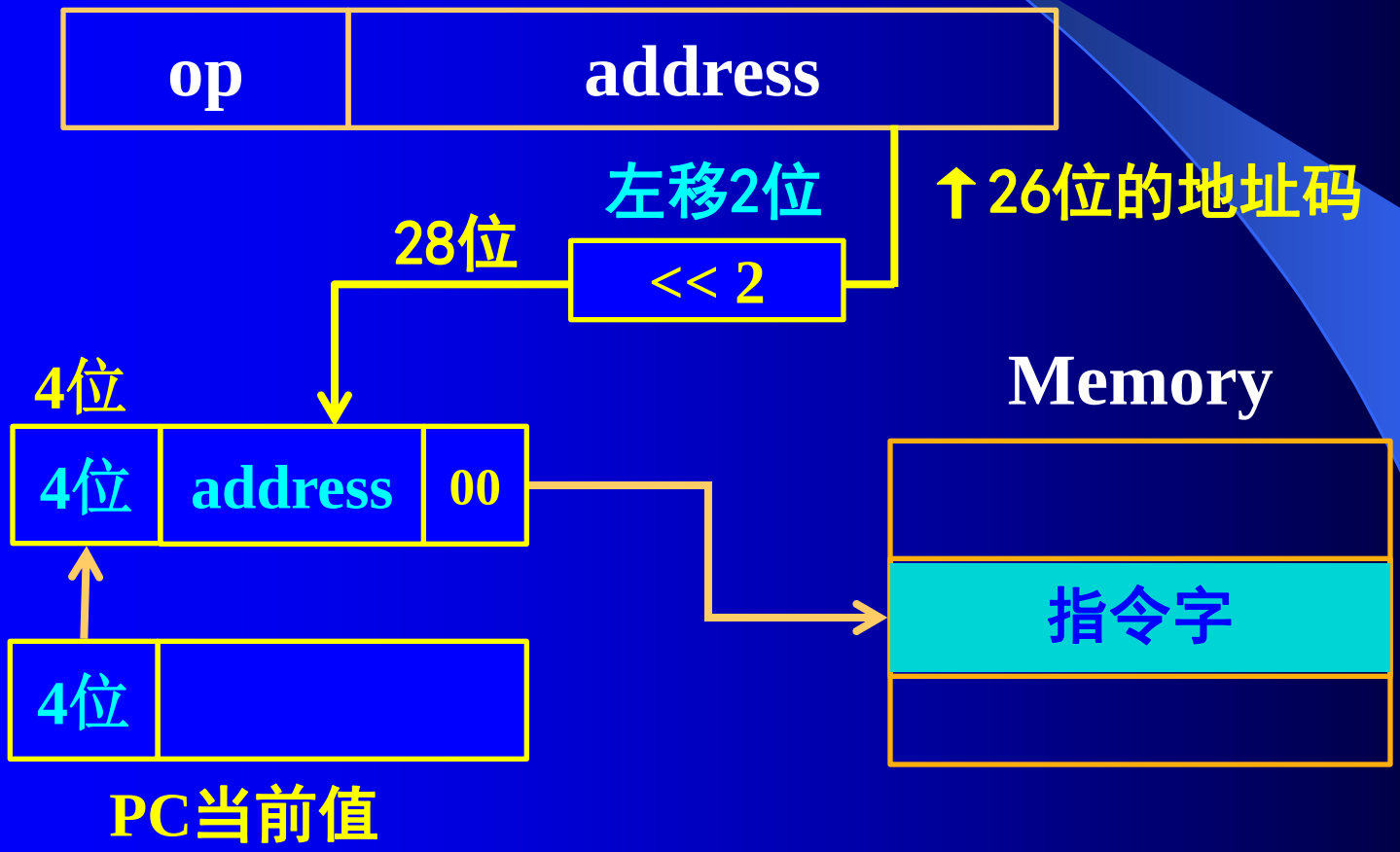
● PC相对寻址(PC-relative Addressing)

操作数由寄存器和立即数字段联合产生



●伪直接寻址（Pseudo-direct Addressing）

也叫页面寻址，由PC高4位与指令中的地址段组合产生有效地址。



3. 指令代码与功能

※ R型指令（只列出了9条）

指令	[31:26]	[25:21]	[20:16]	[15:11]	[10:6]	[5:0]	指令功能
add	000000	rs	rt	rd	00000	100000	寄存器加
sub	000000	rs	rt	rd	00000	100010	寄存器减
and	000000	rs	rt	rd	00000	100100	寄存器与
or	000000	rs	rt	rd	00000	100101	寄存器或
xor	000000	rs	rt	rd	00000	100110	寄存器异或
sll	000000	00000	rt	rd	sa	000000	逻辑左移
srl	000000	00000	rt	rd	sa	000010	逻辑右移
sra	000000	00000	rt	rd	sa	000011	算术右移
jr	000000	rs	00000	00000	00000	001000	寄存器跳转

由操作码op配合func字段，确定具体的操作

R型指令，存在3种不同类型：

① 3寄存器R型指令

指令	[31:26]	[25:21]	[20:16]	[15:11]	[10:6]	[5:0]	指令功能
add	000000	rs	rt	rd	00000	100000	寄存器加
sub	000000	rs	rt	rd	00000	100010	寄存器减
and	000000	rs	rt	rd	00000	100100	寄存器与
or	000000	rs	rt	rd	00000	100101	寄存器或
xor	000000	rs	rt	rd	00000	100110	寄存器异或

add/sub/and/or/xor rd, rs, rt;

指令功能： $rd ← $rs op $rt;

②2寄存器R型指令

指令	[31:26]	[25:21]	[20:16]	[15:11]	[10:6]	[5:0]	指令功能
sll	000000	00000	rt	rd	sa	000000	逻辑左移
srl	000000	00000	rt	rd	sa	000010	逻辑右移
sra	000000	00000	rt	rd	sa	000011	算术右移

sll/srl/sra rd, rt, sa；

指令功能： $rd ← $rt shift sa；

③1寄存器R型指令

指令	[31:26]	[25:21]	[20:16]	[15:11]	[10:6]	[5:0]	指令功能
jr	000000	rs	00000	00000	00000	001000	寄存器跳转

jr rs； 指令功能： PC ← $rs;

※I型指令(只列出9条)

指令	[31:26]	[25:21]	[20:16]	[15:0]	指令功能
addi	001000	rs	rt	imm	寄存器和立即数“加”
andi	001100	rs	rt	imm	寄存器和立即数“与”
ori	001101	rs	rt	imm	寄存器和立即数“或”
xori	001110	rs	rt	imm	寄存器和立即数“异或”
lw	100011	rs	rt	imm	从存储器中读取数据
sw	101011	rs	rt	imm	把数据保存到存储器
beq	000100	rs	rt	imm	寄存器相等则转移
bne	000101	rs	rt	imm	寄存器不等则转移
lui	001111	00000	rt	imm	设置寄存器的高16位

I型指令，存在4种不同类型：

①面向运算的I型指令

指令	[31:26]	[25:21]	[20:16]	[15:0]	指令功能
addi	001000	rs	rt	imm	寄存器和立即数“加”
andi	001100	rs	rt	imm	寄存器和立即数“与”
ori	001101	rs	rt	imm	寄存器和立即数“或”
xori	001110	rs	rt	imm	寄存器和立即数“异或”

addi rt, rs, imm; # $rt ← $rs + E(imm)

这里的imm为数值型数据，故“带符号扩展”。

andi/ori/xori rt, rs, imm; # $rt ← $rs op E(imm)

这里的imm为逻辑型数据，故“无符号扩展”。

②面向访存的I型指令

指令	[31:26]	[25:21]	[20:16]	[15:0]	指令功能
lw	100011	rs	rt	imm	从存储器中读取数据
sw	101011	rs	rt	imm	把数据保存到存储器

MIPS32中唯一两条访问存储器的指令（RISC）

lw rt, imm(rs) # $rt ← mem[$rs+E(imm)]

sw rt, imm(rs) # mem[$rs+E(imm)] ←$rt

符号扩展

③面向数位设置的I型指令

指令	[31:26]	[25:21]	[20:16]	[15:0]	指令功能
lui	001111	00000	rt	imm	设置寄存器的高16位

lui rt, imm # $rt ← imm<<16（空位补0）

④面向条件转移(分支)的I型指令

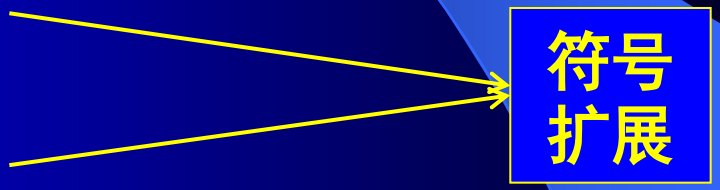
指令	[31:26]	[25:21]	[20:16]	[15:0]	指令功能
beq	000100	rs	rt	imm	寄存器相等则转移
bne	000101	rs	rt	imm	寄存器不等则转移

beq rs, rt, imm

#if($rs==$rt) PC←PC+E(imm)<<2

bne rs, rt, imm

#if($rs!=$rt) PC←PC+E(imm)<<2



是标准的PC相对寻址方式

其中imm要先“带符号扩展”成32位，再左移2位。

※J型指令(列出2条)

指令	[31:26]	[25:0]	指令功能
j	000010	address	无条件跳转
jal	001100	address	调用与联接

j address;

指令功能：$PC← ($PC+4)$_{H4}$ ∪ (address<<2)

jal address;

指令功能：

$ra←$PC+4（保存返回地址）

$PC ← ($PC+4)$_{H4}$ ∪ (address<<2)

这也是标准的页面寻址方式（伪直接寻址）。