MIPS 指令集(共31条)

转载 白手起家的亿万富翁 最后发布于2018-08-31 15:22:24 阅读数 24773 收藏

MIPS 指令集(共31条)

MIPS 指令集(共31条)

助记符	指令格式						示例	示例含义	操作及其解释
Bit #	31..26	25..21	20..16	15..11	10..6	5..0			
R-type	op	rs	rt	rd	shamt	func			
add	000000	rs	rt	rd	00000	100000	add $1,$2,$3	$1=$2+$3	rd <- rs + rt ； 其中rs＝$2，rt=$3, rd=$1
addu	000000	rs	rt	rd	00000	100001	addu $1,$2,$3	$1=$2+$3	rd <- rs + rt ； 其中rs＝$2，rt=$3, rd=$1,无符号数
sub	000000	rs	rt	rd	00000	100010	sub $1,$2,$3	$1=$2-$3	rd <- rs - rt ； 其中rs＝$2，rt=$3, rd=$1
subu	000000	rs	rt	rd	00000	100011	subu $1,$2,$3	$1=$2-$3	rd <- rs - rt ； 其中rs＝$2，rt=$3, rd=$1,无符号数
and	000000	rs	rt	rd	00000	100100	and $1,$2,$3	$1=$2 & $3	rd <- rs & rt ； 其中rs＝$2，rt=$3, rd=$1
or	000000	rs	rt	rd	00000	100101	or $1,$2,$3	$1=$2 | $3	rd <- rs | rt ； 其中rs＝$2，rt=$3, rd=$1
xor	000000	rs	rt	rd	00000	100110	xor $1,$2,$3	$1=$2 ^ $3	rd <- rs xor rt ； 其中rs＝$2，rt=$3, rd=$1(异或)
nor	000000	rs	rt	rd	00000	100111	nor $1,$2,$3	$1=~($2 | $3)	rd <- not(rs | rt) ； 其中rs＝$2，rt=$3, rd=$1(或非)
slt	000000	rs	rt	rd	00000	101010	slt $1,$2,$3	if($2<$3) $1=1 else $1=0	if (rs < rt) rd=1 else rd=0 ； 其中rs＝$2，rt=$3, rd=$1
sltu	000000	rs	rt	rd	00000	101011	sltu $1,$2,$3	if($2<$3) $1=1 else $1=0	if (rs < rt) rd=1 else rd=0 ； 其中rs＝$2，rt=$3, rd=$1 (无符号数)
sll	000000	00000	rt	rd	shamt	000000	sll $1,$2,10	$1=$2<<10	rd <- rt << shamt ； shamt存放移位的位数, 也就是指令中的立即数，其中rt=$2, rd=$1
srl	000000	00000	rt	rd	shamt	000010	srl $1,$2,10	$1=$2>>10	rd <- rt >> shamt ； (logical) ，其中rt=$2, rd=$1
sra	000000	00000	rt	rd	shamt	000011	sra $1,$2,10	$1=$2>>10	rd <- rt >> shamt ； (arithmetic) 注意符号位保留 其中rt=$2, rd=$1
sllv	000000	rs	rt	rd	00000	000100	sllv $1,$2,$3	$1=$2<<$3	rd <- rt << rs ； 其中rs＝$3，rt=$2, rd=$1
srlv	000000	rs	rt	rd	00000	000110	srlv $1,$2,$3	$1=$2>>$3	rd <- rt >> rs ； (logical)其中rs＝$3，rt=$2, rd=$1
srav	000000	rs	rt	rd	00000	000111	srav $1,$2,$3	$1=$2>>$3	rd <- rt >> rs ； (arithmetic) 注意符号位保留 其中rs＝$3，rt=$2, rd=$1
jr	000000	rs	00000	00000	00000	001000	jr $31	goto $31	PC <- rs
I-type	op	rs	rt	immediate					
addi	001000	rs	rt	immediate			addi $1,$2,100	$1=$2+100	rt <- rs + (sign-extend)immediate ； 其中rt=$1,rs=$2
addiu	001001	rs	rt	immediate			addiu $1,$2,100	$1=$2+100	rt <- rs + (zero-extend)immediate ； 其中rt=$1,rs=$2
andi	001100	rs	rt	immediate			andi $1,$2,10	$1=$2 & 10	rt <- rs & (zero-extend)immediate ； 其中rt=$1,rs=$2
ori	001101	rs	rt	immediate			andi $1,$2,10	$1=$2 | 10	rt <- rs | (zero-extend)immediate ； 其中rt=$1,rs=$2
xori	001110	rs	rt	immediate			andi $1,$2,10	$1=$2 ^ 10	rt <- rs xor (zero-extend)immediate ； 其中rt=$1,rs=$2
lui	001111	00000	rt	immediate			lui $1,100	$1=100*65536	rt <- immediate*65536 ； 将16位立即数放到目标寄存器高16位，目标寄存器的低16位填0
lw	100011	rs	rt	immediate			lw $1,10($2)	$1=memory[$2+10]	rt <- memory[rs + (sign-extend)immediate] ； rt=$1,rs=$2
sw	101011	rs	rt	immediate			sw $1,10($2)	memory[$2+10]=$1	memory[rs + (sign-extend)immediate] <- rt ； rt=$1,rs=$2

beq	000100	rs	rt	immediate	beq $1,$2,10	if($1==$2) goto PC+4+40	if (rs == rt) PC <- PC+4 + (sign-extend)immediate<<2
bne	000101	rs	rt	immediate	bne $1,$2,10	if($1!=$2) goto PC+4+40	if (rs != rt) PC <- PC+4 + (sign-extend)immediate<<2
slti	001010	rs	rt	immediate	slti $1,$2,10	if($2<10) $1=1 else $1=0	if (rs <(sign-extend)immediate) rt=1 else rt=0； 其中rs＝$2，rt=$1
sltiu	001011	rs	rt	immediate	sltiu $1,$2,10	if($2<10) $1=1 else $1=0	if (rs <(zero-extend)immediate) rt=1 else rt=0； 其中rs＝$2，rt=$1
J-type	op	address					
j	000010	address			j 10000	goto 10000	PC <- (PC+4)[31..28],address,0,0 ；address=10000/4
jal	000011	address			jal 10000	$31<-PC+4; goto 10000	$31<-PC+4；PC <- (PC+4)[31..28],address,0,0 ；address=10000/4

注意：因为MIPS16只有16个16位的寄存器，所以JAL指令中$31改成$15, 所有立即数均无需扩展，LUI指令直接就是将立即数付给RT寄存器。