MIPS指令系统介绍

- MIPS寄存器组
- MIPS指令目录
- MIPS指令格式
- 部分MIPS指令格式详解

注：本章着重介绍32位MIPS的指令系统中我们要涉及到的指令以及有关的内部组织与结构，其中有部分指令和寄存器定义会在后续章节中根据MIPS16 的特点进行调整。

MIPS寄存器组

共有32个32位寄存器

Register name	Number	Usage
$zero	0	constant 0
$at	1	reserved for assembler
$v0~$v1	2~3	expression evaluation and results of a function
$a0~$a3	4~7	arguments 0~3
$t0~$t7	8~15	temporary (not preserved across call)
$s0~$s7	16~23	saved temporary (preserved across call)
$t8~$t9	24~25	temporary (not preserved across call)
$k0~$k1	26~27	reserved for OS kernel
$gp	28	point to global area
$sp	29	stack pointer
$fp	30	frame point
$ra	31	return address (used by function call)

MIPS指令目录

- 算术指令－add, addu, addi, addiu, sub, subu, mul, mulu, div, divu, mfhi, mflo
- 逻辑指令－and, andi, or, ori, xor, xori, nor, sll, srl, sra, sllv, srlv, srav
- 数据传送指令－lw, lh, lhu, lb, lbu, sw, sh, sb, lui
- 比较、条件转移指令－beq, bne, bnez, slt, slti, sltu, sltiu
- 无条件转移指令－j, jr, jal

MIPS指令格式(1)

- （1） R-format

add $1, $2, $3 # $1=$2+$3

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	2	3	1	0	32
000000	00010	00011	00001	00000	100000

MIPS指令格式(2)

- （2） I-format

lw $1, 10($2) # $1=Memory[$2 +10]

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Address/Immediate
35	2	1	10
100011	00010	00011	0000 0000 0000 1010

MIPS指令格式(3)

- （3） J-format

j 10000 # go to 10000

6-bit	26-bit
op	Target Address
2	2500

000010 00010 00011 0000 0000 0000 1010

部分MIPS指令格式详解

- 这里只介绍几个典型的指令格式，比较完整的格式请参看网站中相关页的说明。

部分MIPS指令格式详解

- `add $s1, $s2, $s3 # $s1=$s2+$s3`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	18	19	17	0	32
000000	10010	10011	10001	00000	100000

- `sub $s1, $s2, $s3 # $s1=$s2-$s3`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	18	19	17	0	34
000000	10010	10011	10001	00000	100010

部分MIPS指令格式详解

- `and $s1, $s2, $s3 # $s1=$s2 AND $s3`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	18	19	17	0	36
000000	10010	10011	10001	00000	100100

- `or $s1, $s2, $s3 # $s1=$s2 OR $s3`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	18	19	17	0	37
000000	10010	10011	10001	00000	100101

部分MIPS指令格式详解

- `addi $s1, $s2, 100 # $s1=$s2+100`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Immediate
8	18	17	100

001000 10010 10001 0000 0000 0110 0100

- `andi $s1, $s2, 100 # $s1=$s2 AND 100`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Immediate
12	18	17	100

001100 10010 10001 0000 0000 0110 0100

部分MIPS指令格式详解

- `ori $s1, $s2, 100 # $s1=$s2 OR 100`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Immediate
13	18	17	100
001101	10010	10001	0000 0000 0110 0100

- `sll $s1, $s2, 10 # $s1=shift($s2) left logical 10 bits`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	0	18	17	10	0
000000	00000	10010	10001	01010	000000

部分MIPS指令格式详解

- srl $s1, $s2, 10 #$s1=shift($s2) right logical 10 bits

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	0	18	17	10	2
000000	00000	10010	10001	01010	000011

- sra $s1, $s2, 10 #$s1=shift($s2) right arithmetic 10 bits

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	0	18	17	10	2
000000	00000	10010	10001	01010	000011

部分MIPS指令格式详解

- `lw $s1, 100($s2) # $s1=Memory[$s2+100]`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Address
35	18	17	100

100011 10010 10001 0000 0000 0110 0100

- `sw $s1, 100($s2) # $Memory[$s2+100]=$s1`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Address
43	18	17	100

101011 10010 10001 0000 0000 0110 0100

部分MIPS指令格式详解

- `beq $s1, $s2, 25 # if $s1=$s2, goto PC+4+25*4`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Address
4	17	18	25

000100 10001 10010 0000 0000 0001 1001

- `bne $s1, $s2, 25 # if $s1≠$s2, goto PC+4+25*4`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Address
5	17	18	25

000101 10001 10010 0000 0000 0001 1001

部分MIPS指令格式详解

- `slt $s1, $s2, $s3 # if $s2<$s3, $s1=1; else $s1=0`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	18	19	17	0	42
000000	10010	10011	10001	00000	101010

- `j 10000 # jump to 10000`

6-bit	26-bit
op	Target address
2	2500
000010	00 0000 0000 0000 1001 1100 0100

部分MIPS指令格式详解

- `jal 10000 # $31=PC+4; jump to 10000`

6-bit	26-bit
op	Target address
3	2500

000013 00 0000 0000 0000 1001 1100 0100

- `jr $ra # jump register $ra`

6-bit	5-bit	5-bit	5-bit	5-bit	6-bit
op	rs	rt	rd	shamt	funct
0	31	0	0	0	8

000000 11111 00000 00000 00000 001000

部分MIPS指令格式详解

- `lui $s1, 100 # $s1=100<<16`

6-bit	5-bit	5-bit	16-bit
op	rs	rt	Immediate
15	0	17	100
001111	00000	10001	0000 0000 0110 0100