REGISTROS DEL MIPS R2000/R3000

Nombre	Código del registro	Uso
$zero	0	Valor constante 0
$v0 - $v1	2 – 3	Valores de retorno para las llamadas al sistema
$a0 - $a3	4 – 7	Argumentos para las llamadas al sistema
$t0 - $t7	8 – 15	Resultados temporales
$s0 - $s7	16 – 23	Valores que se quieren mantener a lo largo del programa
$t8 - $t9	24 – 25	Resultados temporales
$ra	31	Dirección de retorno

MODOS DE DIRECCIONAMIENTO

Las instrucciones del MIPS R2000/R3000 principalmente utilizan registros y números a la hora de operar con los datos. A veces es necesario acceder a la memoria para encontrar los operandos.

El procesador MIPS R2000/R3000 solo ofrece un modo de direccionamiento para el acceso a memoria: **`num(rx)`** con el que se obtiene una dirección sumando el contenido del registro `rx` y el valor inmediato `num`.

El simulador permite otros modos de direccionamiento para el acceso a memoria. Estos modos solo se pueden utilizar con instrucciones que hagan referencia a una dirección.

Formato	Cálculo de la dirección
(registro)	La dirección está en el registro
num	El valor inmediato representa una dirección
num (registro)	La dirección es la suma del contenido del registro más el valor inmediato
etiqueta	La etiqueta indica la dirección
etiqueta ± num	La dirección se calcula sumándole num a la dirección que representa la etiqueta
etiqueta ± num (registro)	La dirección se calcula sumándole a la dirección que representa la etiqueta el contenido del registro y el valor inmediato num

FORMATOS DE LAS INSTRUCCIONES DEL MIPS R2000/R3000

Tres son los diferentes formatos de instrucciones utilizados por el MIPS R2000/R3000.

Formato tipo R:

co	rs	rt	rd	desp	func
6	5	5	5	5	6

En este formato las instrucciones sólo utilizan registros. Los campos que componen este formato de instrucciones son:

- **co**: código de operación
- **rs**: primera referencia a un registro fuente
- **rt**: segunda referencia a un registro fuente
- **rd**: referencia a un registro destino
- **desp**: valor numérico utilizado en las instrucciones de desplazamiento
- **func**: campo de función utilizado para seleccionar una variante del código de operación.

Ejemplo:

add $t0,$t1,$t2 codificación: 0x012A4020

0	9	10	8	0	32
000000	01001	01010	01000	00000	100000

Formato tipo I:

co	**rs**	**rd**	**num**
6	5	5	16

En este formato las instrucciones utilizan dos registros y un número de 16 bits. Los campos que componen este formato de instrucciones son:

- **co**: código de operación
- **rs**: referencia a un registro fuente
- **rd**: referencia a un registro destino
- **num**: valor numérico

Ejemplo:

ori $t0,$t1,1 codificación: 0x35280001

13	9	8	1
001101	01001	01010	0000000000000001

Formato tipo J:

co	**etiqueta**
6	26

Este formato solo se utiliza con instrucciones de salto incondicional. En este formato las instrucciones no utilizan ningún registro. Los campos que componen este formato de instrucciones son:

- **co**: código de operación
- **etiqueta**: nombre asignado a la dirección de la siguiente instrucción a ejecutar

LAS LLAMADAS AL SISTEMA

Servicio	Cód. servicio	Argumentos	Valor devuelto
Escribir entero	1	$a0 = entero	
Escribir coma flotante de simple precisión	2	$f12 = coma flotante	
Escribir coma flotante de doble precisión	3	$f12 = coma flotante	
Escribir cadena	4	$a0 = dir a partir de la cual tenemos la cadena	
Pedir entero	5		$v0 = entero
Pedir coma flotante de simple precisión	6		$f0 = coma flotante
Pedir coma flotante de doble precisión	7		$f0 = coma flotante
Pedir cadena	8	$a0 = dir a partir de la cual se guardará la cadena, $a1 = longitud máxima permitida	$v0 = longitud de la cadena introducida
Terminar programa	10		

DIRECTIVAS

Directiva	Uso
.align n	Alinea los datos que siguen a la siguiente dirección múltiplo de 2^n
.ascii "cad1", …	Guarda cadenas sin terminación nula
.asciiz "cad1", …	Guarda cadenas con terminación nula en cada una de ellas
.byte b1,…	Guarda valores enteros de 1 byte
.data [dir]	Todo lo que siga esta directiva se guarda como dato a partir de dir. Si no se especifica dir, por defecto dir = 0x10010000
.double d1, …	Guarda valores de coma flotante de doble precisión
.float f1, …	Guarda valores de coma flotante de simple precisión
.globl etiqueta	Declara etiqueta como global para todos los archivos
.half h1, …	Guarda valores enteros de 2 bytes
.space n	Reserva n bytes contiguos en la memoria
.text [dir]	Todo lo que siga esta directiva se guarda como instr. a partir de dir. Si no se especifica dir, por defecto dir = 0x00400000
.word w1, …	Guarda valores enteros de 4 bytes

TABLA ASCII

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	¿
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	⌂
80	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
90	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø	£	Ø	×	ƒ
100	á	í	ó	ú	ñ	Ñ	ª	º	¿	®	¬	½	¼	¡	«	»

REPERTORIO DE INSTRUCCIONES UTILIZADO CON EL SIMULADOR PCSPIM

Leyenda:

R	Formato de instrucción tipo R
I	Formato de instrucción tipo I
J	Formato de instrucción tipo J
Ps	Pseudoinstrucción

Para las pseudoinstrucciones:

rdest	Como destino hay que utilizar un registro entero
rori	Como origen hay que utilizar un registro entero
ori	Como origen se puede utilizar un registro entero o un valor inmediato entero
fdest	Como destino hay que utilizar un registro de coma flotante

Instrucción		Tipo	Campos						Comentarios
Instrucciones con registros de propósito general enteros									
Sumas									
add	rd, rs, rt	R	0	rs	rt	rd	0	0x20	rd = rs + rt (con signo)
addu	rd, rs, rt	R	0	rs	rt	rd	0	0x21	rd = rs + rt (sin signo)
addi	rd, rs, num	I	0x8	rs	rd	num			rd = rs + num (con signo)
addiu	rd, rs, num	I	0x9	rs	rd	num			rd = rs + num (sin signo)
add	rdest, rori1, ori2	Ps							rdest = rori1 + ori2 (con signo)
addu	rdest, rori1, ori2	Ps							rdest = rori1 + ori2 (sin signo)
Valor absoluto									
abs	rdest, rori	Ps							rdest = | rori |
Restas									
sub	rd, rs, rt	R	0	rs	rt	rd	0	0x22	rd = rs – rt (con signo)
subu	rd, rs, rt	R	0	rs	rt	rd	0	0x23	rd = rs – rt (sin signo)
sub	rdest, rori1, ori2	Ps							rdest = rori1 - ori2 (con signo)
subu	rdest, rori1, ori2	Ps							rdest = rori1 - ori2 (sin signo)

Instrucción		Tipo	Campos						Comentarios
Negación									
neg	rdest, rori	Ps							rdest = - rori
Multiplicación									
mult	rs, rt	R	0	rs	rt	0	0	0x18	hi = rs x rt $_{63-32}$, lo = rs x rt $_{31-0}$ (con signo)
multu	rs, rt	R	0	rs	rt	0	0	0x19	hi = rs x rt $_{63-32}$, lo = rs x rt $_{31-0}$ (sin signo)
mul	rdest, rori1, ori2	Ps							rdest = rori1 x ori2
División									
div	rs, rt	R	0	rs	rt	0	0	0x1a	lo = rs / rt , hi = rs % rt (con signo)
divu	rs, rt	R	0	rs	rt	0	0	0x1b	lo = rs / rt , hi = rs % rt (sin signo)
div	rdest, rori1, ori2	Ps							rdest = rori1 / ori2
rem	rdest, rori1, ori2	Ps							rdest = rori1 % ori2
Lógicas									
and	rd, rs, rt	R	0	rs	rt	rd	0	0x24	rd = rs and rt
andi	rd, rs, num	I	0xc	rs	rd	num			rd = rs and num
and	rdest, rori1, ori2	Ps							rdest = rori1 and ori2
or	rd, rs, rt	R	0	rs	rt	rd	0	0x25	rd = rs or rt
ori	rd, rs, num	I	0xd	rs	rd	num			rd = rs or num
or	rdest, rori1, ori2	Ps							rdest = rori1 or ori2
xor	rd, rs, rt	R	0	rs	rt	rd	0	0x26	rd = rs xor rt
xori	rd, rs, num	I	0xe	rs	rd	num			rd = rs xor num
xor	rdest, rori1, ori2	Ps							rdest = rori1 xor ori2
nor	rd, rs, rt	R	0	rs	rt	rd	0	0x27	rd = rs nor rt
not	rdest, rori	Ps							rdest = not rori
Desplazamientos lógicos y rotación a la izquierda									
sll	rd, rt, desp	R	0	0	rt	rd	desp	0	rd = bits rt desplazados desp posiciones
sllv	rd, rt, rs	R	0	rs	rt	rd	0	0x4	rd = bits rt desplazados rs posiciones

Instrucción		Tipo	Campos						Comentarios
sll	rdest, rori1, ori2	Ps							rdest = bits rori1 desplazados ori2 posic.
rol	rdest, rori1, ori2	Ps							rdest = bits rori1 rotados ori2 posiciones
Desplazamientos lógicos y rotación a la derecha									
srl	rd, rt, desp	R	0	0	rt	rd	desp	0x2	rd = bits rt desplazados desp posiciones
srlv	rd, rt, rs	R	0	rs	rt	rd	0	0x6	rd = bits rt desplazados rs posiciones
srl	rdest, rori1, ori2	Ps							rdest = bits rori1 desplazados ori2 posic.
ror	rdest, rori1, ori2	Ps							rdest = bits rori1 rotados ori2 posiciones
Desplazamientos aritméticos a la derecha									
sra	rd, rt, desp	R	0	0	rt	rd	desp	0x3	rd = bits rt desplazados desp posiciones
srav	rd, rt, rs	R	0	rs	rt	rd	0	0x7	rd = bits rt desplazados rs posiciones
sra	rdest, rori1, ori2	Ps							rdest = bits rori1 desplazados ori2 posic.
Comparaciones									
slt	rd, rs, rt	R	0	rs	rt	rd	0	0x2a	rd = 1 si rs < rt (con signo)
sltu	rd, rs, rt	R	0	rs	rt	rd	0	0x2b	rd = 1 si rs < rt (sin signo)
slti	rd, rs, num	I	0xa	rs	rd	num			rd = 1 si rs < num (con signo)
sltiu	rd, rs, num	I	0xb	rs	rd	num			rd = 1 si rs < num (sin signo)
seq	rdest, rori1, ori2	Ps							rdest = 1 si rori1 = ori2
sge	rdest, rori1, ori2	Ps							rdest = 1 si rori1 ≥ ori2 (con signo)
sgeu	rdest, rori1, ori2	Ps							rdest = 1 si rori1 ≥ ori2 (sin signo)
sgt	rdest, rori1, ori2	Ps							rdest = 1 si rori1 > ori2 (con signo)
sgtu	rdest, rori1, ori2	Ps							rdest = 1 si rori1 > ori2 (sin signo)
sle	rdest, rori1, ori2	Ps							rdest = 1 si rori1 ≤ ori2 (con signo)
sleu	rdest, rori1, ori2	Ps							rdest = 1 si rori1 ≤ ori2 (sin signo)
slt	rdest, rori1, ori2	Ps							rdest = 1 si rori1 < ori2 (con signo)
sltu	rdest, rori1, ori2	Ps							rdest = 1 si rori1 < ori2 (sin signo)
sne	rdest, rori1, ori2	Ps							rdest = 1 si rori1 ≠ ori2

Instrucción		Tipo	Campos						Comentarios
Saltos incondicionales									
j	etiqueta	J	0x2	etiqueta					Salta a etiqueta
jal	etiqueta	J	0x3	etiqueta					Salta a etiqueta , $ra = dir. sig. instr.
jalr	rs, rd	R	0	rs	0	rd	0	0x9	Salta a dir. en rs , rd = dir. instr. sig.
jr	rs	R	0	rs	0	0	0	0x8	Salta a dir. en rs
b	etiqueta	Ps							Salta a etiqueta
Saltos condicionales									
beq	rs, rt, etiqueta	I	0x4	rs	rt	offset			Salta a etiqueta si rs = rt
bgez	rs, etiqueta	I	0x1	rs	0x1	offset			Salta a etiqueta si rs ≥ 0
bgtz	rs, etiqueta	I	0x7	rs	0	offset			Salta a etiqueta si rs > 0
blez	rs, etiqueta	I	0x6	rs	0	offset			Salta a etiqueta si rs ≤ 0
bltz	rs, etiqueta	I	0x1	rs	0	offset			Salta a etiqueta si rs < 0
bne	rs, rt, etiqueta	I	0x5	rs	rt	offset			Salta a etiqueta si rs ≠ rt
beqz	rori, etiqueta	Ps	(offset = etiqueta – posición actual)						Salta a etiqueta si rori = 0
bge	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 ≥ ori2 (con signo)
bgeu	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 ≥ ori2 (sin signo)
bgt	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 > ori2 (con signo)
bgtu	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 > ori2 (sin signo)
ble	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 ≤ ori2 (con signo)
bleu	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 ≤ ori2 (sin signo)
blt	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 < ori2 (con signo)
bltu	rori1,ori2, etiqueta	Ps							Salta a etiqueta si rori1 < ori2 (sin signo)
bnez	rori, etiqueta	Ps							Salta a etiqueta si rori ≠ 0
beq	rori1,ori2,etiqueta	Ps							Salta a etiqueta si rori1 = ori2
bne	rori1,ori2,etiqueta	Ps							Salta a etiqueta si rori1 ≠ ori2

Instrucción		Tipo	Campos						Comentarios
Instrucciones de acceso a memoria									
lb	rd, num(rs)	I	0x20	rs	rd	num			Carga 1 byte de dir = num + rs a rd (con)
lbu	rd, num(rs)	I	0x24	rs	rd	num			Carga 1 byte de dir = num + rs a rd (sin)
sb	rt, num(rs)	I	0x28	rs	rt	num			Almacena 1 byte de rt en dir = num + rs
lh	rd, num(rs)	I	0x21	rs	rd	num			Carga 2 bytes de dir = num + rs a rd (con)
lhu	rd, num(rs)	I	0x25	rs	rd	num			Carga 2 bytes de dir = num + rs a rd (sin)
sh	rt, num(rs)	I	0x29	rs	rt	num			Almacena 2 bytes de rt en dir = num + rs
lw	rd, num(rs)	I	0x23	rs	rd	num			Carga 4 bytes de dir = num + rs a rd
sw	rt, num(rs)	I	0x2b	rs	rt	num			Almacena 4 bytes de rt en dir = num + rs
(Nota: Se recuerda que el segundo operando se puede sustituir por cualquier modo de direccionamiento – En ese caso se están utilizando pseudoinstrucciones – ver tabla correspondiente)									
Transferencia de datos									
lui	rd, num	I	0xf	0	rd	num			Carga num en los bits más sign. de rd
li	rdest, num	Ps							Pone num en rdest
la	rdest, etiqueta	Ps							Pone la dirección etiqueta en rdest
move	rdest, rori	Ps							rdest = rori
Otros									
syscall		R	0	0	0	0	0	0xc	Llamada al sistema
nop		R	0	0	0	0	0	0	Ninguna operación
Instrucciones con registros de propósito general de coma flotante									
Aritméticas									
add.s	fd, fs, ft	R	0x11	0x10	ft	fs	fd	0	fd = fs + ft (simple precisión)
add.d	fd, fs, ft	R	0x11	0x11	ft	fs	fd	0	fd = fs + ft (doble precisión)
sub.s	fd, fs, ft	R	0x11	0x10	ft	fs	fd	1	fd = fs - ft (simple precisión)
sub.d	fd, fs, ft	R	0x11	0x11	ft	fs	fd	1	fd = fs - ft (doble precisión)

Instrucción	Tipo	Campos						Comentarios
mul.s fd, fs, ft	R	0x11	0x10	ft	fs	fd	2	fd = fs x ft (simple precisión)
mul.d fd, fs, ft	R	0x11	0x11	ft	fs	fd	2	fd = fs x ft (doble precisión)
div.s fd, fs, ft	R	0x11	0x10	ft	fs	fd	3	fd = fs / ft (simple precisión)
div.d fd, fs, ft	R	0x11	0x11	ft	fs	fd	3	fd = fs / ft (doble precisión)
neg.s fd, fs	R	0x11	0x10	0	fs	fd	7	fd = - fs (simple precisión)
neg.d fd, fs	R	0x11	0x11	0	fs	fd	7	fd = - fs (doble precisión)
abs.s fd, fs	R	0x11	0x10	0	fs	fd	5	fd = | fs | (simple precisión)
abs.d fd, fs	R	0x11	0x11	0	fs	fd	5	fd = | fs | (doble precisión)
Transferencia								
mov.s fd, fs	R	0x11	0x10	0	fs	fd	6	fd = fs (simple precisión)
mov.d fd, fs	R	0x11	0x11	0	fs	fd	6	fd = fs (doble precisión)
l.s fdest, dirección	Ps							Carga un número en coma flotante de simple precisión desde etiqueta al registro fdest
l.d fdest,dirección	Ps							Carga un número en coma flotante de doble precisión desde etiqueta al registro fdest
s.s fdest, dirección	Ps							Guarda a partir de etiqueta el número en coma flotante de simple precisión que hay en fdest
s.d fdest, dirección	Ps							Guarda a partir de etiqueta el número en coma flotante de doble precisión que hay en fdest