Set de instrucciones del Simulador WinMIPS64

Instrucciones de Transferencia de Datos		
lb	r_d, Inm(r_i)	Copia en r_d un byte (8 bits) desde la dirección (Inm+r_i) (con extensión del signo)
lbu	r_d, Inm(r_i)	Copia en r_d un byte (8 bits) desde la dirección (Inm+r_i) (sin extensión del signo)
sb	r_f, Inm(r_i)	Guarda los 8 bits menos significativos de r_f en la dirección (Inm+r_i)
lh	r_d, Inm(r_i)	Copia en r_d un half-word (16 bits) desde la dir. (Inm+r_i) (con extensión del signo)
lhu	r_d, Inm(r_i)	Copia en r_d un half-word (16 bits) desde la dir. (Inm+r_i) (sin extensión del signo)
sh	r_f, Inm(r_i)	Guarda los 16 bits menos significativos de r_f a partir de la dirección (Inm+r_i)
lw	r_d, Inm(r_i)	Copia en r_d un word (32 bits) desde la dir. (Inm+r_i) (con extensión del signo)
lwu	r_d, Inm(r_i)	Copia en r_d un word (32 bits) desde la dir. (Inm+r_i) (sin extensión del signo)
sw	r_f, Inm(r_i)	Guarda los 32 bits menos significativos de r_f a partir de la dirección (Inm+r_i)
ld	r_d, Inm(r_i)	Copia en r_d un double word (64 bits) desde la dirección (Inm+r_i)
sd	r_f, Inm(r_i)	Guarda r_f a partir de la dirección (Inm+r_i)
l.d	f_d, Inm(r_i)	Copia en r_d un valor en punto flotante (64 bits) desde la dirección (Inm+r_i)
s.d	f_f, Inm(r_i)	Guarda f_f a partir de la dirección (Inm+r_i)
mov.d	f_d, f_f	Copia el valor del registro f_f al registro f_d
mtc1	r_f, f_d	Copia los 64 bits del registro entero r_f al registro f_d de punto flotante
mfc1	r_d, f_f	Copia los 64 bits del registro f_f de punto flotante al registro r_d entero
cvt.d.l	f_d, f_f	Convierte a punto flotante el valor entero copiado al registro f_f, dejándolo en f_d
cvt.l.d	f_d, f_f	Convierte a entero el valor en punto flotante contenido en f_f, dejándolo en f_d

Instrucciones Aritméticas		
dadd	r_d, r_f, r_g	Suma r_f con r_g, dejando el resultado en r_d (valores con signo)
daddi	r_d, r_f, N	Suma r_f con el valor inmediato N, dejando el resultado en r_d (valores con signo)
daddu	r_d, r_f, r_g	Suma r_f con r_g, dejando el resultado en r_d (valores sin signo)
daddui	r_d, r_f, N	Suma r_f con el valor inmediato N, dejando el resultado en r_d (valores con signo)
add.d	f_d, f_f, f_g	Suma f_f con f_g, dejando el resultado en f_d (en punto flotante)
dsub	r_d, r_f, r_g	Resta r_g a r_f, dejando el resultado en r_d (valores con signo)
dsubu	r_d, r_f, r_g	Resta r_g a r_f, dejando el resultado en r_d (valores sin signo)
sub.d	f_d, f_f, f_g	Resta f_g a f_f, dejando el resultado en f_d (en punto flotante)
dmul	r_d, r_f, r_g	Mutiplica r_f con r_g, dejando el resultado en r_d (valores con signo)
dmulu	r_d, r_f, r_g	Mutiplica r_f con r_g, dejando el resultado en r_d (valores sin signo)
mul.d	f_d, f_f, f_g	Multiplica f_f con f_g, dejando el resultado en f_d (en punto flotante)
ddiv	r_d, r_f, r_g	Divide r_f por r_g, dejando el resultado en r_d (valores con signo)
ddivu	r_d, r_f, r_g	Divide r_f por r_g, dejando el resultado en r_d (valores sin signo)
div.d	f_d, f_f, f_g	Divide f_f por f_g, dejando el resultado en f_d (en punto flotante)
slt	r_d, r_f, r_g	Compara r_f con r_g, dejando r_d=1 si r_f es menor que r_g (valores con signo)
slti	r_d, r_f, N	Compara r_f con el valor inmediato N, dejando r_d=1 si r_f es menor que N (valores signo)
c.lt.d	f_d, f_f	Compara f_f con f_g, dejando flag FP=1 si f_f es menor que f_g (en punto flotante)
c.le.d	f_d, f_f	Compara f_f con f_g, dejando flag FP=1 si f_f es menor o igual que f_g (en punto flotante)
c.lq.d	f_d, f_f	Compara f_f con f_g, dejando flag FP=1 si f_f es igual que f_g (en punto flotante)

Instrucciones Lógicas		
and	r_d, r_f, r_g	Realiza un AND entre r_f y r_g (bit a bit), dejando el resultado en r_d
andi	r_d, r_f, N	Realiza un AND entre r_f y el valor inmediato N (bit a bit), dejando el resultado en r_d
or	r_d, r_f, r_g	Realiza un OR entre r_f y r_g (bit a bit), dejando el resultado en r_d
ori	r_d, r_f, N	Realiza un OR entre r_f y el valor inmediato N (bit a bit), dejando el resultado en r_d
xor	r_d, r_f, r_g	Realiza un XOR entre r_f y r_g (bit a bit), dejando el resultado en r_d
xori	r_d, r_f, N	Realiza un XOR entre r_f y el valor inmediato N (bit a bit), dejando el resultado en r_d

Instrucciones de desplazamiento de bits		
dsll	r_d, r_f, N	Desplaza a izquierda N veces los bits del registro r_f, dejando el resultado en r_d
dsllv	r_d, r_f, r_N	Desplaza a izquierda r_N veces los bits del registro r_f, dejando el resultado en r_d
dsrl	r_d, r_f, N	Desplaza a derecha N veces los bits del registro r_f, dejando el resultado en r_d
dsrlv	r_d, r_f, r_N	Desplaza a derecha r_N veces los bits del registro r_f, dejando el resultado en r_d
dsra	r_d, r_f, N	Igual que **dsrl** pero mantiene el signo del valor desplazado
dsrav	r_d, r_f, r_N	Igual que **dsrlv** pero mantiene el signo del valor desplazado

Instrucciones de Transferencia de Control		
j	offN	Salta a la dirección rotulada offN
jal	offN	Salta a la dirección rotulada offN y copia en r_{31} la dirección de retorno
jr	r_d	Salta a la dirección contenida en el registro r_d
beq	r_d, r_f, offN	Si r_d es igual a r_f, salta a la dirección rotulada offN
bne	r_d, r_f, offN	Si r_d no es igual a r_f, salta a la dirección rotulada offN
beqz	r_d, offN	Si r_d es igual a 0, salta a la dirección rotulada offN
bnez	r_d, offN	Si r_d no es igual a 0, salta a la dirección rotulada offN
bc1f	offN	Salta a la dirección rotulada offN si flag FP=1 (ó true) (en punto flotante)
bc1t	offN	Salta a la dirección rotulada offN si flag FP=0 (ó false) (en punto flotante)

Instrucciones de Control	
nop	Operación nula
halt	Detiene el simulador