MIPS Instructions

Instruction	Description	Template	Calculation
add	Add	add $d, $s, $t	$d = $s + $t
addi	Add Immediate	addi $d, $s, i	$d = $s + i
sub	Subtract	sub $d, $s, $t	$d = $s - $t
div	Divide	div $s, $t	lo = $s / $t; hi = $s % $t
mult	Multiply	mult $s, $t	hi:lo = $s * $t
sll	Shift Left Immediate	sll $d, $t, i	$d = $t << i
sllv	Shift Left	sllv $d, $t, $s	$d = $t << $s
sra	Shift Right Immediate	sra $d, $t, i	$d = $t >> i
srav	Shift Right	srav $d, $t, $s	$d = $t >> $s
and	And	and $d, $s, $t	$d = $s & $t
andi	And Immediate	andi $d, $s, i	$d = $s & i
or	Or	or $d, $s, $t	$d = $s | $t
ori	Or Immediate	ori $d, $s, i	$d = $s | i
nor	Nor	nor $d, $s, $t	$d = ~($s | $t)
xor	Exclusive Or	xor $d, $s, $t	$d = $s ^ $t
xori	Exclusive Or Immediate	ori $d, $s, i	$d = $s ^ i
slt	Strictly Less Than	slt $d, $s, $t	$d = ($s < $t)
slti	Strictly Less Than Immediate	slti $d, $s, i	$d = ($s < i)
beq	Branch if Equal	beq $s, $t, label	if ($s == $t) pc += i << 2

bgtz	Branch if Greater Than Zero	bgtz $s, label	if ($s > 0) pc += i << 2
blez	Branch if Less Than or Equal to Zero	blez $s, label	if ($s <= 0) pc += i << 2
bne	Branch if Not Equal	bne $s, $t, label	if ($s != $t) pc += i << 2
llo	Load Low Half of Word	llo $d, i	LOWHALF($d) = i
lhi	Load High Half of Word	lhi $d, i	HIGHHALF($d) = i
lw	Load Word	lw $t, i($s)	$d = MEM[$s + i]
sw	Save Word	sw $t, i($s)	MEM[$s + i] = $t
mfhi	Move From High Reg	mfhi $d	$d = hi
mflo	Move From Low Reg	mflo $d	$d = lo
mthi	Move To High Reg	mthi $s	hi = $s
mtlo	Move To Low Reg	mtlo $s	lo = $s
j	Jump	j label	pc += i << 2
jal	Jump and Link	jal label	$31 = pc; pc += i << 2
jalr	Jump and Link Register	jalr $s	$31 = pc; pc = $s
jr	Jump Register	jr $s	pc = $s
trap	System Call	trap i	See trap list below

Traps

Trap Number	Description	Type
0	Halt the Program	System
1	Print the integer in $4 to the console	Console
4	Print the ASCIIZ string to console starting at the address in $4	Console
5	Read an integer from the user console and place value in $2	Console
8	Read a string from the user console and store at the address in $4. Length of buffer in $5 * Need to allocate one more word than used (for the terminating 0 in the ASCIIZ string)	Console
99	Places a random number from between the values in $4 and $5 into $2.	Math
101	Print character to console in $4	Console
102	Read character from user console and place value in $2	Console
103	Returns the last character the user entered into the console if empty returns 0	Console
104	Returns True(value greater than 0) if user is pressing the key value stored in register 4 and returns false(value is 0) if the user is not	Screen
105	Sleeps for the number of millis in register 4	System
106	Sleeps for the number of millis in register 4 minus the time it took since the trap was called	System
111	Breaks the program if breakpoints are enabled	System
130	Gives the lower half of the system millis and stores it in register 2	System
150	Sets the screen size where width is the number in register 4 and height is in register 5	Screen
151	Sets the pixel color at $4 -> x, $5 -> y, $6 -> color stored as a RGB int	Screen
152	Set the pixel color at index of $4, $6 -> color stored as RGB int	Screen

153	Updates the screen to the new values	Screen
154	hsv 0 - 255, h $4, s $5, v $6 - returns rgb values into register 4, 5, 6	Math
155	hsv 0 - 255, h $4, s $5, v $6 - returns color int $4	Math

Directives

Directive	Description
.word	Allows for the storage of 32 bit integers separated by commas
.hword	Allows for the storage of 16 bit integers separated by commas
.byte	Allows for the storage of 8 bit integers separated by commas
.ascii	Allows for the encoding of ascii characters defined between commas to be stored as a string of bytes in memory
.space N	Allocate N bytes of space
.org	Sets the origin in memory of the code below

Register Use

Registers	Description	Preserved During Function Call?
$0	Constant value of 0 - does not change	N/A
$1		
$2 - $3	Return values and Expression Evaluation	No
$4 - $7	Arguments	Yes
$8 - $15	Temporary Registers	No
$16 - $23	Saved Registers	Yes
$24 - $25	More Temporary Registers	No
$28	Global Pointer	Yes
$29	Stack Pointer	Yes
$30	Frame Pointer	Yes
$31	Return Address	Yes
lo, hi	For values larger than 32-bit integers	N/A
pc	Program Counter	N/A