# SPIM Quick Reference

## MIPS Registers and Usage Convention

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Number</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>zero</td>
<td>0</td>
<td>Constant 0</td>
</tr>
<tr>
<td>at</td>
<td>1</td>
<td>Reserved for assembler</td>
</tr>
<tr>
<td>v0</td>
<td>2</td>
<td>Expression evaluation and results of a function</td>
</tr>
<tr>
<td>v1</td>
<td>3</td>
<td>Expression evaluation and results of a function</td>
</tr>
<tr>
<td>a0</td>
<td>4</td>
<td>Argument 1</td>
</tr>
<tr>
<td>a1</td>
<td>5</td>
<td>Argument 2</td>
</tr>
<tr>
<td>a2</td>
<td>6</td>
<td>Argument 3</td>
</tr>
<tr>
<td>a3</td>
<td>7</td>
<td>Argument 4</td>
</tr>
<tr>
<td>t0</td>
<td>8</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t1</td>
<td>9</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t2</td>
<td>10</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t3</td>
<td>11</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t4</td>
<td>12</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t5</td>
<td>13</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t6</td>
<td>14</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t7</td>
<td>15</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>s0</td>
<td>16</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s1</td>
<td>17</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s2</td>
<td>18</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s3</td>
<td>19</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s4</td>
<td>20</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s5</td>
<td>21</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s6</td>
<td>22</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>s7</td>
<td>23</td>
<td>Saved temporary (preserved across call)</td>
</tr>
<tr>
<td>t8</td>
<td>24</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>t9</td>
<td>25</td>
<td>Temporary (not preserved across call)</td>
</tr>
<tr>
<td>k0</td>
<td>26</td>
<td>Reserved for OS kernel</td>
</tr>
<tr>
<td>k1</td>
<td>27</td>
<td>Reserved for OS kernel</td>
</tr>
<tr>
<td>gp</td>
<td>28</td>
<td>Pointer to global area</td>
</tr>
<tr>
<td>sp</td>
<td>29</td>
<td>Stack pointer</td>
</tr>
<tr>
<td>fp</td>
<td>30</td>
<td>Frame pointer</td>
</tr>
<tr>
<td>ra</td>
<td>31</td>
<td>Return address (used by function call)</td>
</tr>
</tbody>
</table>

## System Services

<table>
<thead>
<tr>
<th>Service</th>
<th>System Call Code</th>
<th>Arguments</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>print_int</td>
<td>1</td>
<td>$a0 = integer</td>
<td></td>
</tr>
<tr>
<td>Instruction</td>
<td>Value</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-------</td>
<td>------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>print_float</td>
<td>2</td>
<td>$f12 = float</td>
<td></td>
</tr>
<tr>
<td>print_double</td>
<td>3</td>
<td>$f12 = double</td>
<td></td>
</tr>
<tr>
<td>print_string</td>
<td>4</td>
<td>$a0 = string</td>
<td></td>
</tr>
<tr>
<td>read_int</td>
<td>5</td>
<td>integer (in $v0)</td>
<td></td>
</tr>
<tr>
<td>read_float</td>
<td>6</td>
<td>float (in $f0)</td>
<td></td>
</tr>
<tr>
<td>read_double</td>
<td>7</td>
<td>double (in $f0)</td>
<td></td>
</tr>
<tr>
<td>read_string</td>
<td>8</td>
<td>$a0 = buffer, $a1 = length</td>
<td></td>
</tr>
<tr>
<td>sbrk</td>
<td>9</td>
<td>$a0 = amount, $a1 = length</td>
<td></td>
</tr>
<tr>
<td>exit</td>
<td>10</td>
<td>address (in $v0)</td>
<td></td>
</tr>
</tbody>
</table>

**Assembler Directives**

- `.align n`
- `.ascii str`
- `.asciiz str`
- `.byte b1, ..., bn`
- `.data`
- `.double d1, ..., dn`
- `.extern sym size`
- `.float f1, ..., fn`
- `.globl sym`
- `.half h1, ..., hn`
- `.kdata`
- `.ktext`
- `.space n`
- `.text`
- `.word w1, ..., wn`

**SPIM Instructions and Common Macros**

**Arithmetic and Logical Instructions**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs Rdest, Rsrc</td>
<td>Absolute Value</td>
</tr>
<tr>
<td>add Rdest, Rsrc1, Src2</td>
<td>Addition (with overflow)</td>
</tr>
<tr>
<td>addi Rdest, Rsrc1, Imm</td>
<td>Addition Immediate (with overflow)</td>
</tr>
<tr>
<td>addu Rdest, Rsrc1, Src2</td>
<td>Addition (without overflow)</td>
</tr>
<tr>
<td>addiu Rdest, Rsrc1, Imm</td>
<td>Addition Immediate (without overflow)</td>
</tr>
<tr>
<td>and Rdest, Rsrc1, Src2</td>
<td>AND</td>
</tr>
<tr>
<td>andi Rdest, Rsrc1, Imm</td>
<td>AND Immediate</td>
</tr>
<tr>
<td>div Rsrc1, Rsrc2</td>
<td>Divide (with overflow)</td>
</tr>
<tr>
<td>divu Rsrc1, Rsrc2</td>
<td>Divide (without overflow)</td>
</tr>
<tr>
<td>div Rdest, Rsrc1, Src2</td>
<td>Divide (with overflow)</td>
</tr>
<tr>
<td>divu Rdest, Rsrc1, Src2</td>
<td>Divide (without overflow)</td>
</tr>
<tr>
<td>mul Rdest, Rsrc1, Src2</td>
<td>Multiply (without overflow)</td>
</tr>
<tr>
<td>mulo Rdest, Rsrc1, Src2</td>
<td>Multiply (with overflow)</td>
</tr>
<tr>
<td>mulou Rdest, Rsrc1, Src2</td>
<td>Unsigned Multiply (with overflow)</td>
</tr>
<tr>
<td>mult Rsrc1, Rsrc2</td>
<td>Multiply</td>
</tr>
<tr>
<td>multu Rsrc1, Rsrc2</td>
<td>Unsigned Multiply</td>
</tr>
<tr>
<td>neg Rdest, Rsrc</td>
<td>Negate Value (with overflow)</td>
</tr>
</tbody>
</table>
negu Rdest, Rsrc
nor Rdest, Rsrcl, Src2
not Rdest, Rsrc
or Rdest, Rsrcl, Src2
ori Rdest, Rsrcl, Imm
rem Rdest, Rsrcl, Src2
remu Rdest, Rsrcl, Src2
rol Rdest, Rsrcl, Src2
ror Rdest, Rsrcl, Src2
sll Rdest, Rsrcl, Src2
sllv Rdest, Rsrcl, Rsrcl2
sra Rdest, Rsrcl, Src2
srav Rdest, Rsrcl, Rsrcl2
srl Rdest, Rsrcl, Src2
srlv Rdest, Rsrcl, Rsrcl2
sub Rdest, Rsrcl, Src2
subu Rdest, Rsrcl, Src2
xor Rdest, Rsrcl, Src2
xori Rdest, Rsrcl, Imm

Constant-Manipulating Instructions
li Rdest, imm
lui Rdest, imm

Comparison Instructions
seq Rdest, Rsrcl, Src2
sge Rdest, Rsrcl, Src2
sgeu Rdest, Rsrcl, Src2
sgt Rdest, Rsrcl, Src2
sgtu Rdest, Rsrcl, Src2
sle Rdest, Rsrcl, Src2
sleu Rdest, Rsrcl, Src2
slt Rdest, Rsrcl, Src2
slti Rdest, Rsrcl, Imm
sltu Rdest, Rsrcl, Src2
sne Rdest, Rsrcl, Src2

Branch and Jump Instructions
b label
bczt label
bczf label
beq Rsrcl, Src2, label
beqz Rsrcl, label
bge Rsrcl, Src2, label
bgez Rsrcl, label
bgeu Rsrcl, Src2, label
bgez Rsrcl, label
bgezal Rsrcl, label
bgt Rsrcl, Src2, label

Negate Value (without overflow)
NOR
NOT
OR
OR Immediate
Remainder
Unsigned Remainder
Rotate Left
Rotate Right
Shift Left Logical
Shift Left Logical Variable
Shift Right Arithmetic
Shift Right Arithmetic Variable
Shift Right Logical
Shift Right Logical Variable
Subtract (with overflow)
Subtract (without overflow)
XOR
XOR Immediate
Load Immediate
Load Upper Immediate
Set Equal
Set Greater Than Equal
Set Greater Than Equal Unsigned
Set Greater Than
Set Greater Than Unsigned
Set Less Than Equal
Set Less Than Equal Unsigned
Set Less Than
Set Less Than Immediate
Set Less Than Unsigned
Set Less Than Unsigned Immediate
Set Not Equal
Branch instruction
Branch Coprocessor z True
Branch Coprocessor z False
Branch on Equal
Branch on Equal Zero
Branch on Greater Than Equal
Branch on Greater Than Equal Zero
Branch on Greater Than Equal Zero And Link
Branch on Greater Than
Branch on Greater Than
Branch on Greater Than Unsigned
Branch on Greater Than Zero
Branch on Less Than Equal
Branch on LTE Unsigned
Branch on Greater Than Equal Zero And Link
Branch on Less Than And Link
Branch on Less Than
Branch on Less Than Zero
Branch on Less Than Unsigned
Branch on Not Equal Zero
Jump
Jump and Link
Jump and Link Register
Jump Register
Load Address
Load Byte
Load Signed Byte
Load Double-Word
Load Halfword
Load Unsigned Halfword
Load Word
Load Word Coprocessor "z"
Load Word Left
Load Word Right
Unaligned Load Halfword
Unaligned Load Halfword Unsigned
Unaligned Load Word
Store Byte
Store Double-Word
Store Halfword
Store Word
Store Word Coprocessor "z"
Store Word Left
Store Word Right
Unaligned Store Halfword
Unaligned Store Word
Move
Move From hi
Move From lo
Move To hi
Move To lo
mfcz Rdest, CPsrc
mfc1.d Rdest, FRsrc1
mtcz Rsrc, CPdest

Floating Point Instructions
abs.d FRdest, FRsrc
abs.s FRdest, FRsrc
add.d FRdest, FRsrcl, FRsrc2
add.s FRdest, FRsrcl, FRsrc2
c.eq.d FRsrcl, FRsrc2
c.eq.s FRsrcl, FRsrc2
c.le.d FRsrcl, FRsrc2
c.le.s FRsrcl, FRsrc2
c.lt.d FRsrcl, FRsrc2
c.lt.s FRsrcl, FRsrc2
cvt.d.s FRdest, FRsrc
cvt.d.w FRdest, FRsrc
cvt.s.d FRdest, FRsrc
cvt.s.w FRdest, FRsrc
cvt.w.d FRdest, FRsrc
cvt.w.s FRdest, FRsrc
div.d FRdest, FRsrcl, FRsrc2
div.s FRdest, FRsrcl, FRsrc2
l.d FRdest, address
l.s FRdest, address
mov.d FRdest, FRsrc
mov.s FRdest, FRsrc
mul.d FRdest, FRsrcl, FRsrc2
mul.s FRdest, FRsrcl, FRsrc2
neg.d FRdest, FRsrc
neg.s FRdest, FRsrc
s.d FRdest, address
s.s FRdest, address
sub.d FRdest, FRsrcl, FRsrc2
sub.s FRdest, FRsrcl, FRsrc2

Exception and Trap Instructions
rfe
syscall
break n
nop

Taken from https://www.cs.tcd.ie/John.Waldron/itral/spim_ref.html created by reformattting portions of the LaTeX source file of the public domain documentation distributed with the SPIM simulator.