

**MIPS assembler directives**

(From *Computer Organization and Design - The Hardware/Software Interface* by Dave Patterson and John Hennessy, 2nd edition)

SPIM supports a subset of the assembler directives provided by the actual MIPS assembler:

<code>.align n</code>	Align the next datum on a $2^n$ byte boundary.  For example, <code>.align 2</code> aligns the next value on a word boundary. <code>.align 0</code> turns off automatic alignment of <code>.half</code> , <code>.word</code> , <code>.float</code> , and <code>.double</code> directives until the next <code>.data</code> or <code>.kdata</code> directive.
<code>.ascii str</code>	Store the string in memory, but do not null-terminate it.
<code>.asciiz str</code>	Store the string in memory and null-terminate it.
<code>.byte b1,..., bn</code>	Store the $n$ values in successive bytes of memory.
<code>.data &lt;addr&gt;</code>	The following data items should be stored in the data segment. If the optional argument <i>addr</i> is present, the items are stored beginning at address <i>addr</i> .
<code>.double d1,..., dn</code>	Store the $n$ floating point double precision numbers in successive memory locations.
<code>.extern sym size</code>	Declare that the datum stored at <i>sym</i> is <i>size</i> bytes large and is a global symbol. This directive enables the assembler to store the datum in a portion of the data segment that is efficiently accessed via register <code>\$gp</code> .
<code>.float f1,..., fn</code>	Store the $n$ floating point single precision numbers in successive memory locations.
<code>.globl sym</code>	Declare that symbol <i>sym</i> is global and can be referenced from other files.
<code>.half h1,..., hn</code>	Store the $n$ 16-bit quantities in successive memory halfwords.
<code>.kdata &lt;addr&gt;</code>	The following data items should be stored in the kernel data segment. If the optional argument <i>addr</i> is present, the items are stored beginning at address <i>addr</i> .
<code>.ktext &lt;addr&gt;</code>	The next items are put in the kernel text segment. In SPIM, these items may only be instructions or words (see the <code>.word</code> directive below). If the optional argument <i>addr</i> is present, the items are stored beginning at address <i>addr</i> .
<code>.space n</code>	Allocate $n$ bytes of space in the current segment (which must be the data segment in SPIM).

<code>.text &lt;addr&gt;</code>	The next items are put in the user text segment. In SPIM, these items may only be instructions or words (see the <code>.word</code> directive below). If the optional argument <i>addr</i> is present, the items are stored beginning at address <i>addr</i> .
<code>.word w1,..., wn</code>	Store the <i>n</i> 32-bit quantities in successive memory words. SPIM does not distinguish various parts of the data segment ( <code>.data</code> , <code>.rdata</code> and <code>.sdata</code> ).