MARS supported subset of MIPS assembler directives

`.align n`	Align the next datum on a 2^n byte boundary. For example, .align 2 aligns the next value on a word boundary. .align 0 turns off automatic alignment of .half, .word, .float, and .double directives until the next .data or .kdata directive.
`.ascii str`	Store the string in memory, but do not null-terminate it.
`.asciiz str`	Store the string in memory and null-terminate it.
`.byte b1,..., bn`	Store the *n* values in successive bytes of memory.
`.data <addr>`	The following data items should be stored in the data segment. If the optional argument *addr* is present, the items are stored beginning at address *addr* .
`.double d1,..., dn`	Store the *n* floating point double precision numbers in successive memory locations.
`.extern sym size`	Declare that the datum stored at sym is size 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 $gp.
`.float f1,..., fn`	Store the *n* floating point single precision numbers in successive memory locations.
`.globl sym`	Declare that symbol sym is global and can be referenced from other files.
`.half h1,..., hn`	Store the *n* 16-bit quantities in successive memory halfwords.
`.kdata <addr>`	The following data items should be stored in the kernel data segment. If the optional argument *addr* is present, the items are stored beginning at address *addr* .
`.ktext <addr>`	The next items are put in the kernel text segment. In SPIM, these items may only be instructions or words (see the .word directive below). If the optional argument *addr* is present, the items are stored beginning at address *addr* .
`.set noat` and `.set at`	The first directive prevents SPIM from complaining about subsequent instructions that use register $at. The second directive re-enables the warning. Since seudoinstructions expand into code that uses register $at, programmers must be very careful about leaving values in this register.
`.space n`	Allocate *n* bytes of space in the current segment (which must be the data segment in SPIM).
`.text <addr>`	The next items are put in the user text segment. In SPIM, these items may only be instructions or words (see the .word directive below). If the optional argument *addr* is present, the items are stored beginning at address *addr* .
`.word w1,..., wn`	Store the *n* 32-bit quantities in successive memory words. SPIM does not distinguish various parts of the data segment (.data, .rdata and .sdata).
since MARS 4.3	
`.eqv`	Substitute second operand for first. First operand is symbol, second operand is expression (like #define) (http://courses.missouristate.edu/KenVollmar/MARS/Help/MacrosHelp.html)
`.macro`	Begin macro definition. See .end_macro
`.end_macro`	End macro definition. See .macro
`.include`	Insert the contents of the specified file. Put filename in quotes. (http://courses.missouristate.edu/KenVollmar/MARS/Help/MacrosHelp.html)