

GAS Directives

Tony Richardson

April 7, 2006

An abbreviated list of GAS directives is shown below. See the *Using as* manual for the complete list. The descriptions here are extracted from the *Using as* manual.

.section *name*

Use the `.section` directive to assemble the following code into a section named *name*. Typical sections are `.text` (program code), `.data` (initialized data), and `.bss` (uninitialized data).

.include *file*

This directive provides a way to include supporting files at specified points in your source program. The code from *file* is assembled as if it followed the point of the `.include`; when the end of the included file is reached, assembly of the original file continues. You can control the search paths used with the `-I` command-line option. Quotation marks are required around *file*.

.equ *symbol, expression, .set symbol, expression*

This directive sets the value of *symbol* to *expression*. Directives `.equ` and `.set` are synonyms.

.global *symbol, .globl symbol*

`.global` makes the *symbol* visible to `ld`. If you define *symbol* in your partial program, its value is made available to other partial programs that are linked with it. Otherwise, *symbol* takes its attributes from a *symbol* of the same name from another file linked into the same program. Both spellings (`.globl` and `.global`) are accepted, for compatibility with other assemblers.

.ascii "string", .asciz "string", .string "string"

`.ascii` expects zero or more string literals separated by commas. It assembles each string (with no automatic trailing zero byte) into consecutive addresses. `.asciz` and `.string` are similar, but automatically add a trailing zero (or null) byte.

.byte *expressions, .short expressions, .int expressions, .quad expressions*

.word *expressions, .long expressions*

.single *expressions, .double expressions, .tfloat expressions*

.float *expressions*

Each of these directives emits a number (or numbers if *expressions* is a comma separated list) of the appropriate type and size. Directives `.byte`, `.short`, `.int`, `.quad` emit 1, 2, 4, and 8 byte integers. Directives `.word` and `.long` are synonymous with `.short` and `.int` respectively. Directives `.single`, `.double` and `.tfloat` emit 4, 8 and 10 byte floating point numbers. Directive `.float` is a synonym for `.single`. These directives may be used in the `.bss` section where they will cause space to be allocated. The actual value of the expression is ignored in the `.bss` section (but an expression list must still be included).

.fill *repeat, size, value*

repeat, *size* and *value* are absolute expressions. This emits *repeat* copies of *size* bytes. *Repeat* may be zero or more. *Size* may be zero or more, but if it is more than 8, then it is deemed to have the value 8, compatible with other people's assemblers. The contents of each repeat bytes is taken from an 8-byte number. The highest order 4 bytes are zero. The lowest order 4 bytes are value rendered in the byte-order of an integer on the computer as is assembling for. Each *size* bytes in a repetition is taken from the lowest order *size* bytes of this number. Again, this bizarre behavior is compatible with other people's assemblers. *size* and *value* are optional. If the second comma and *value* are absent, *value* is assumed zero. If the first comma and following tokens are absent, *size* is assumed to be 1. This directive may be used in the `.bss` section where it causes space to be allocated. *value* (if present) is ignored in the `.bss` section.