Lissom Assembler Directives Reference

v0r0: 01/09/2010
Initial description of directives

v0r1: 29/09/2010
Improved description of .text, .data and .section directives, updated list of unsupported directives.

1 Lissom assembler-specific directives

.section_adjustable *name*, *flags*, *word bitwidth, byte bitwidth, endianess*

name – Identifier that specifies section name, if a section with the same name was already defined, this directive sets as actual section the already existing section.

flags – String that specifies section type flags, curwently are supported: “x” - text (code), “d” - initialized read/write data, “b” - uninitialized read/write data.

word_bitwidth – Bitwidth of one word.

byte_bitwidth - Bitwidth of one least addressable unit (LAU), word bitwidth must be divisible by byte bitwidth.

endianess – either .big or .little, assembler and linker currently supports both, but program loader in simulator supports only big endianess.

.bit *bitwidth, values*

bitwidth – *D*ata bitwidth, must be a multiple of current section's byte bitwidth.

values – List of expressions separated by comma.

.listing-on, .listing-off

Turns on/off listing in form: address, binary coding.

.debug-on, .debug-off

Turns on/off parser's debug output, can be used to find why is an instruction incorrectly translated and to find collisions in assembly grammar.

Remaining directives are exactly the same as in GNU as (http://sourceware.org/binutils/docs-2.20/as/Pseudo-Ops.html#Pseudo-Ops), default section word bitwidth is 32 and byte bitwidth is 8.

2 List of supported GNU as directives

Here are only directive names listed, for more detailed description please see the GNU as user manual at http://sourceware.org/binutils/docs-2.20/as/Pseudo-Ops.html#Pseudo-Ops.

Target-independent

.ascii (8-bit characters), .asciiz (8-bit characters), .comm, .equ, .equiv, .eqv, .err, .error, .global, .globl, .hword (16-bit), .int (32-bit), .lcomm, .long (32-bit), .org, .short (16-bit), .skip, .sleb128, .space, .string (8-bit characters), .string8, .uleb128, .warning, word (32-bit), .line, .file, .loc

Target-dependent

For target-dependent directives are certain values of *word bitwidth, byte bitwidth* and *endianess* read from the ISAC model default memory map.

Section definitions use all words of *word bitwidth* bits, bytes of *byte bitwidth bits* and *endianess* from the model. Directives used to specify sections are these:

.bbs, .data, .rdata, .text, .usect

Only data byte bitwidth is affected by the model and this value is used by following directives:

.byte, .2byte, .4byte, .align, .balign, .fill, .p2align

3 List of non-supported GNU as directives (warning is printed)

.abort, .desc, .dim, .double, .endef, .fail, .float, .incbin, .include, .linkonce, .octa, .popsection, .pushsection, .quad, .reloc, .scl, .single, .string16, .string32, .string64, .subsection, .usect, .weak, .cpload, .cprestore

4 List of ignored GNU as directives (nothing is printed)

.arch, .end, .ent., .extern, .frame, .fmask, .gnu_attribute, .ident, .lflags, .ln, .mask, .previous, .size, .type, .func, .endfunc, .cfi_startproc, .cfi_endproc, .cfi_offset, .cfi_def_cfa_offset, .cfi_def_cfa_register

5 List of GNU as directives unknown to assembler (syntax error is printed)

.altmacro, .def, .eject, .else, .elseif, .endif, .exitm, .hidden, .if, .internal, .irp, .irpc, .list, .local, .macro, .mri, .noalmacro, .nolist, .print, .protected, .psize, .purgem, .rept, .sbttl .sizem, .stabd, .struct, .symver

6 Identifiers

Can be formed of literals, decimal digits, dots, underscores and dollar characters ($), a digit as the first character may not be a digit.

[$A-Za-z._]([$0-9A-Za-Z._])+

7 Expressions

Expression are inspired by the C language. Parentheses can be used as is common. Supported operators, sorted by priority:

1. +, - , ~, ! (unary),
2. *, /, %,
3. +, -,
4. <<, >>,
5. &,
6. ^,
7. |.

Relocation expressions for expressions the cannot be evaluated during assembly (those that use a label) can be in the form: label [>> shift [(+|-) offset [& mask]]].

8 Comments

Basic form of comments is the same as in C++, i.e. /* ... */ and //. The user may also specify a character that starts a one-line comment using the ISAC language section ASSEMBLER_SYNTAX.

9 Architecture-specific expression functions

These are auxiliary functions that can be used in expressions and were introduced for compatibility with specific target compiler outputs.

%hi(value)**, %lo**(value) - introduced for MIPS
#lhi(value)**, #llo**(value) - introduced for MSP430

Return high (31..16) / low (15 .. 0) bits of a value or of a symbol.