Expressions

The Assembler incorporates constant expressions. Expressions can consist of operands, operators and functions. All expressions are internally 32 bits in AVRASM, and 64 bits in AVRASM2.

Operands

The following operands can be used:

- User defined labels which are given the value of the location counter at the place they appear.
- User defined variables defined by the SET directive
- User defined constants defined by the EQU directive
- Integer constants: constants can be given in several formats, including
  - Decimal (default): 10, 255
  - Hexadecimal (two notations): 0x0a, $0a, 0xff, $ff
  - Binary: 0b00001010, 0b11111111
  - Octal (leading zero): 010, 077
- PC - the current value of the Program memory location counter
- Floating point constants - AVRASM2 only.

Operators

The Assembler supports a number of operators which are described here. The higher the precedence, the higher the priority. Expressions may be enclosed in parentheses, and such expressions are always evaluated before combined with anything outside the parentheses. The associativity of binary operators indicates the evaluation order of chained operators, left associativity meaning they are evaluated left to right, i.e., 2 - 3 - 4 is (2 - 3) - 4, while right associativity would mean 2-3-4 is 2 - (3 - 4). Some operators are not assoiciative, meaning chaining has no meaning.

The following operators are defined:

Symbol	Description
!	Logical Not
~	Bitwise Not
-	Unary Minus
*	Multiplication
/	Division
%	Modulo (AVRASM2 only)
+	Addition
-	Subtraction
<<	Shift left
>>	Shift right
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
==	Equal

!=	Not equal
&	Bitwise And
^	Bitwise Xor
|	Bitwise Or
&&	Logical And
||	Logical Or
?	Conditional operator (AVRASM2 only)

Logical Not

```
Symbol:        !
Description:   Unary operator which returns 1 if the expression was zero, and
returns 0 if the expression was nonzero
Precedence:    12
Associativity: None
Example:       ldi r16,!0xf0  ; Load r16 with 0x00
```

Bitwise Not

```
Symbol:        ~
Description:   Unary operator which returns the input expression with all bits
inverted
Precedence:    12
Associativity: None
Example:       ldi r16,~0xf0  ; Load r16 with 0x0f
```

Unary Minus

```
Symbol:        -
Description:   Unary operator which returns the arithmetic negation of an
expression
Precedence:    14
Associativity: None
Example:       ldi r16,-2  ; Load -2(0xfe) in r16
```

Multiplication

```
Symbol:        *
Description:   Binary operator which returns the product of two expressions
Precedence:    13
Associativity: Left
Example:       ldi r30,label*2 ; Load r30 with label*2
```

Division

```
Symbol:        /
Description:   Binary operator which returns the integer quotient of the left
expression divided by the right expression
Precedence:    13
Associativity: Left
Example:       ldi r30,label/2 ; Load r30 with label/2
```

Modulo (AVRASM2 Only)

```
Symbol:        %
Description:   Binary operator which returns the integer remainder of the left
expression divided by the right expression
Precedence:    13
Associativity: Left
Example:       ldi r30,label%2 ; Load r30 with label%2
```

Addition

```
Symbol:        +
Description:   Binary operator which returns the sum of two expressions
Precedence:    12
Associativity: Left
Example:       ldi r30,c1+c2  ; Load r30 with c1+c2
```

Subtraction

```
Symbol:        -
Description:   Binary operator which returns the left expression minus the right
expression
Precedence:    12
Associativity: Left
Example:       ldi r17,c1-c2  ;Load r17 with c1-c2
```

Shift left

```
Symbol:        <<
Description:   Binary operator which returns the left expression shifted left the
number given by the right expression
Precedence:    11
Associativity: Left
Example:       ldi r17,1<<bitmask  ;Load r17 with 1 shifted left bitmask times
```

Shift right

```
Symbol:        >>
Description:   Binary operator which returns the left expression shifted right
the number given by the right expression
Precedence:    11
Associativity: Left
Example:       ldi r17,c1>>c2  ;Load r17 with c1 shifted right c2 times
```

Less than

```
Symbol:        <
Description:   Binary operator which returns 1 if the signed expression to the
left is Less than the signed expression to the right, 0 otherwise
Precedence:    10
Associativity: None
Example:       ori r18,bitmask*(c1<c2)+1  ;Or r18 with an expression
```

Less or equal

```
Symbol:        <=
Description:   Binary operator which returns 1 if the signed expression to the
left is Less than or Equal to the signed expression to the right, 0 otherwise
Precedence:    10
Associativity: None
Example:       ori r18,bitmask*(c1<=c2)+1 ;Or r18 with an expression
```

Greater than

```
Symbol:        >
Description:   Binary operator which returns 1 if the signed expression to the
left is Greater than the signed expression to the right, 0 otherwise
Precedence:    10
Associativity: None
Example:       ori r18,bitmask*(c1>c2)+1  ;Or r18 with an expression
```

Greater or equal

```
Symbol:        >=
Description:   Binary operator which returns 1 if the signed expression to the
left is Greater than or Equal to the signed expression to the right, 0 otherwise
Precedence:    10
Associativity: None
Example:       ori r18,bitmask*(c1>=c2)+1 ;Or r18 with an expression
```

Equal

```
Symbol:        ==
Description:   Binary operator which returns 1 if the signed expression to the
left is Equal to the signed expression to the right, 0 otherwise
Precedence:    9
Associativity: None
Example:       andi r19,bitmask*(c1==c2)+1 ;And r19 with an expression
```

Not equal

```
Symbol:        !=
Description:   Binary operator which returns 1 if the signed expression to the
left is Not Equal to the signed expression to the right, 0 otherwise
Precedence:    9
Associativity: None
Example:       .SET flag=(c1!=c2)  ;Set flag to 1 or 0
```

Bitwise And

```
Symbol:        &
Description:   Binary operator which returns the bitwise And between two
expressions
Precedence:    8
Associativity: Left
```

```
Example:       ldi r18,High(c1&c2) ;Load r18 with an expression
```

Bitwise Xor

```
Symbol:        ^
Description:   Binary operator which returns the bitwise Exclusive Or between two
expressions
Precedence:    7
Associativity: Left
Example:       ldi r18,Low(c1^c2) ;Load r18 with an expression
```

Bitwise Or

```
Symbol:        |
Description:   Binary operator which returns the bitwise Or between two
expressions
Precedence:    6
Associativity: Left
Example:       ldi r18,Low(c1|c2) ;Load r18 with an expression
```

Logical And

```
Symbol:        &&
Description:   Binary operator which returns 1 if the expressions are both
nonzero, 0 otherwise
Precedence:    5
Associativity: Left
Example:       ldi r18,Low(c1&&c2)  ;Load r18 with an expression
```

Logical Or

```
Symbol:        ||
Description:   Binary operator which returns 1 if one or both of the expressions
are nonzero, 0 otherwise
Precedence:    4
Associativity: Left
Example:       ldi r18,Low(c1||c2)  ;Load r18 with an expression
```

Conditional operator (AVRASM2 only)

```
Symbol:        ? :
Syntax:        condtion? expression1 : expression2
Description:   Ternary operator which returns expression1 if condition is true,
expression2 otherwise.
Precedence:    3
Associativity: None
Example:       ldi r18, a > b? a : b  ; Load r18 with the maximum numeric value
of a and b.
Note:          This feature was introduced in AVRASM 2.1 and is not available in
2.0 or earlier versions.
```

Functions

The following functions are defined:

- LOW(expression) returns the low byte of an expression

- HIGH(expression) returns the second byte of an expression
- BYTE2(expression) is the same function as HIGH
- BYTE3(expression) returns the third byte of an expression
- BYTE4(expression) returns the fourth byte of an expression
- LWRD(expression) returns bits 0-15 of an expression
- HWRD(expression) returns bits 16-31 of an expression
- PAGE(expression) returns bits 16-21 of an expression
- EXP2(expression) returns 2 to the power of expression
- LOG2(expression) returns the integer part of log2(expression)

The following functions are only defined in AVRASM2:

- INT(expression) Truncates a floating point expression to integer (ie discards fractional part)
- FRAC(expression) Extracts fractional part of a floating point expression (ie discards integer part).
- Q7(expression) Converts a fractional floating point expression to a form suitable for the FMUL/FMULS/FMULSU instructions. (sign + 7-bit fraction)
- Q15(expression) Converts a fractional floating point expression to a form suitable for the FMUL/FMULS/FMULSU instructions. (sign +15-bit fraction)
- ABS() Returns the absolute value of a constant expression.
- DEFINED(symbol) Returns true if *symbol* is previously defined using .equ/.set/.def directives. Normally used in conjunction with .if directives (.if defined(foo)), but may be used in any context. It differs from other functions in that parentheses around its argument are not required, and that it only makes sense to use a single symbol as argument.
- STRLEN(string) returns the length of a string c+onstant, in bytes.