

# ASM11

A two-pass absolute assembler for the 68HC11

## Quick Reference Guide

Last Update: 06-Nov-98 for ASM11 v1.46

### Command-Line Syntax and Options

```
ASM11 [-option [...]] [[@]filespec [...]] [>errfile]
```

- *option(s)* may appear before, in between or after *filespec(s)*.
- *option(s)* specified apply to all files assembled, regardless of command line placement.
- Text file(s) containing list(s) of files to be processed may be specified by naming the text file on the command line, prefixed with a "@" character. These text files may not contain command line options.
- *filespec(s)* may include wildcard characters (?,\*). Wildcards are not allowed in *filespec(s)* that are prefixed with a "@".
- If the file extension for a source *filespec* is omitted, the extension ".ASM" is assumed (see description of the **-R.ext** option below).
- If assembler error output to display is enabled, it may be redirected to *errfile* using standard DOS output redirection syntax. This capability may be used in conjunction with, or as an alternative to the **-E+** option.
- The assembler will set the DOS ERRORLEVEL variable when it terminates as indicated:
  - 0 No error, assembly of last file was successful
  - 1 System error (hardware I/O failure, out of disk space, etc.), or -W option failure
  - 2 Error(s) generated (or Escape pressed) during assembly of last file
  - 3 Warning(s) generated during assembly of last file
  - 4 Assembler was not started (help screen displayed, **-w** option used with success)

Option	Default	Description
<b>-C</b> [±]	-C-	Label case sensitivity: + = case sensitive
<b>-E</b> [±]	-E-	Generate *.ERR file (one for each file assembled). *.ERR files are not generated for file(s) that do not contain errors.
<b>-EH</b> [±]	-EH+	If -E+ is in effect, hide (do not display) error messages on screen.
<b>-L</b> [±]	-L+	Create a *.LST file (one for each file assembled).
<b>-LS</b> [±]	-LS-	Create a *.SYM symbol list (one for each file assembled). May be useful for debuggers that do not support the P&E mapfile format.
<b>-M</b> [±]	-M+	Create a *.MAP (one for each file assembled). *.MAP files created may be used with debuggers that support the P&E map file format.
<b>-MT</b> x	-MTP	Specifies type of MAP file to be generated (if <b>-M+</b> in effect): <b>-MTA</b> : Generate parsable ASCII map file <b>-MTP</b> : Generate P&E-style map file

<b>-R<math>n</math></b>	-R74	Specifies maximum length of S-record files. The length count $n$ includes all characters in an S-record, including the leading "S" and record type, but not the CR/LF line terminator.
<b>-R.ext</b>	-R.ASM	Specifies the default extension to assume for source files specified on the command line which do not directly specify an extension.
<b>-REL</b> [ $\pm$ ]	-REL+	Allows generation of "BRA/BSR instead of JMP/JSR" optimization warnings when enabled.
<b>-RTS</b> [ $\pm$ ]	-RTS-	Allows generation of "JSR followed by RTS" subroutine call optimization warnings when enabled.
<b>-S</b> [ $\pm$ ]	-S+	Generate *.S19 object file (one for each file assembled).
<b>-SH</b> [ $\pm$ ]	-SH-	Include dummy "S0" record (header) in object file (only if <b>-S+</b> ).
<b>-T<math>n</math></b>	-T8	Specifies tab field width to use in *.LST files. Tab characters embedded in the source file are converted to spaces in the listing file such that columns are aligned to every $n$ th character.
<b>-X</b> [ $\pm$ ]	-X+	Allow recognition of extra, non-68HC11-standard mnemonics in source files.
<b>-W</b>	(none)	Write options specified on command line to the ASM11 executable. The user-specified options become the default values used by ASM11 in subsequent invocations. <i>Filespec(s)</i> on the command line are ignored. Assembly of source files does not take place if this option is specified.

## Source File Pseudo-Ops

Pseudo-Op	Description
<b>DB</b> <i>string expr[,...]</i>	Define Byte(s). <i>expr</i> may be a constant numeric, a label reference, an expression, or a string. <b>DB</b> encodes a single byte in the object file at the current location counter for each <i>expr</i> encountered (using the LSB of the result) or one byte for each character in <i>strings</i> .
<b>DS</b> <i>blocksize</i>	Define Storage. The assembler's location counter is incremented by <i>blocksize</i> . No code is generated.
<b>DW</b> <i>expr[,...]</i>	Define Word(s). <i>expr</i> may be a constant numeric, a label or an expression. <i>expr</i> is always interpreted as a word (16-bit) quantity, and is stored in the object file at the current location counter, high byte followed by low byte.
<b>END</b> [ <i>expr</i> ]	Provided for compatibility. The END directive cannot be used to terminate assembly; ASM11 always processes the source file to the end of file. If <i>expr</i> is specified, the word result is encoded in the S9 record of the object file.
<i>label</i> <b>EQU</b> <i>expr</i>	Assigns the value of <i>expr</i> to <i>label</i> .
<b>FCB</b> <i>string expr[,...]</i>	Form Constant Byte(s). Same as <b>DB</b> .
<b>FCC</b> <i>string expr[,...]</i>	Form Constant Character(s). Same as <b>DB</b> .
<b>FCS</b> <i>string expr[,...]</i>	Form Constant String. Similar to <b>FCC</b> , but automatically adds a terminating null (0) byte to the end of the string defined.
<b>FDB</b> <i>expr[,...]</i>	Form Double Byte(s). Same as <b>DW</b> .
<b>ORG</b> <i>expr</i>	Sets the assembler's location counter. Code generated after this directive will be assembled starting at the location specified by <i>expr</i> .
<b>RMB</b> <i>blocksize</i>	Reserve Memory Byte(s). Same as <b>DS</b> .

## Source File Processing Directives

- All processing directives must be prefixed with a **\$** or **#** character. ASM11 will recognize either character as the start of a processing directive.
- If a directive has a corresponding command-line option, the directive in the source file will override the command line directive at the point in which the source file directive is encountered.

Directive	Description
<b>\$CASEOFF</b>	When <b>\$CASEOFF</b> is in effect, all symbol references that follow are converted to uppercase internally before they are searched for or placed in the symbol table. Equivalent to the <b>-C-</b> command line option.
<b>\$CASEON</b>	When <b>\$CASEON</b> is in effect, symbol references are NOT internally converted to uppercase before they are searched for or placed in the symbol table. Equivalent to the <b>-C+</b> command line option.
<b>\$ELSE</b>	When used in conjunction with conditional assembly directives ( <b>\$IF</b> , <b>\$IFDEF</b> , <b>\$IFNDEF</b> , <b>\$IFEXIST</b> , <b>\$IFNEXIST</b> , <b>\$IFZ</b> , <b>\$IFNZ</b> ), code following the <b>\$ELSE</b> directive is assembled if the conditional it is paired with evaluates to a not-true result.
<b>\$ENDIF</b>	Marks the end of a conditional-assembly block. Conditional assembly statements may be nested if they are properly blocked with <b>\$ENDIF</b> directives.
<b>\$ERROR</b> [ <i>text</i> ]	When encountered in the source, the assembler issues a error message in the same form as internally-generated errors, using the <i>text</i> specified, prefixed with "USER: "
<b>\$EXTRAOFF</b>	Disables recognition of ASM11's extended instruction set for source lines that follow this directive. Equivalent to the <b>-X-</b> command line option.
<b>\$EXTRAON</b>	Enables recognition of ASM11's extended instruction set for source lines that follow this directive. Equivalent to the <b>-X+</b> command line option.
<b>\$FATAL</b> [ <i>text</i> ]	Similar to the <b>\$ERROR</b> directive, but generates an assembler fatal error message and terminates assembly.
<b>\$IF</b> <i>expr1 cond expr2</i>	Evaluates <i>expr1</i> and <i>expr2</i> (which may be any valid ASM11 expression) and compares them using the specified <i>cond</i> conditional operator. If the condition is true, the code following the <b>\$IF</b> operator is assembled, up to its matching <b>\$ELSE</b> or <b>\$ENDIF</b> directive. <i>Cond</i> may be any one of: < <= = >= > <> The condition is always evaluated using unsigned arithmetic. If a symbol referenced in <i>expr1</i> or <i>expr2</i> is not defined, the statement will always evaluate as false.

<b>\$IFDEF</b> <i>expr</i>	Attempts to evaluate <i>expr</i> , and if successful, assembles the code that follows, up to the matching <b>\$ELSE</b> or <b>\$ENDIF</b> directive. This directive is usually used to test if a specified symbol has been defined. Symbol(s) referenced in <i>expr</i> must be defined before the directive for the result to evaluate true (e.g., forward references will evaluate as false).
<b>\$IFEXISTS</b> <i>fpath</i>	Checks for the existence of the file specified by <i>fpath</i> (using the same rules as those used for <b>\$INCLUDE</b> directives) and assembles the code which follows if the specified <i>fpath</i> exists.
<b>\$INCLUDE</b> <i>fpath</i>	INCLUDEs the specified <i>fpath</i> file in the assembly stream, as if the contents of the file were physically present in the source at the point where the <b>\$INCLUDE</b> directive is encountered. INCLUDEs may be nested, up to 50 levels (the main source file counts as one level). Relative <i>fpath</i> specifications are always referenced to the directory in which the main source file resides, including any relative <b>\$INCLUDE</b> <i>fpath</i> references in nested include files.  Note: As of version 1.46, ASM11 will only generate a standard error (not an assembly-terminating fatal error) if a file specified in a <b>\$INCLUDE</b> directive is not found. The <b>\$IFEXISTS</b> and <b>\$IFNEXISTS</b> directives may be used in conjunction with <b>\$FATAL</b> if termination of assembly is desired under such conditions.
<b>\$IFNDEF</b> <i>expr</i>	Evaluates <i>expr</i> and assembles the code that follows if the expression could NOT be evaluated; usually as the result of a reference to an undefined symbol. This directive is the functional opposite of the <b>\$IFDEF</b> directive.
<b>\$IFNEXISTS</b> <i>fpath</i>	The opposite of <b>\$IFEXISTS</b> ; code following this directive is assembled if the specified <i>fpath</i> does NOT exist.
<b>\$IFNZ</b> <i>expr</i>	Evaluates <i>expr</i> and assembles the code that follows if the expression evaluates to a non-zero value. <b>\$IFNZ</b> always evaluates as false if <i>expr</i> references undefined or forward-defined symbols.
<b>\$IFZ</b> <i>expr</i>	Evaluates <i>expr</i> and assembles the code that follows if the expression is equal to zero. <b>\$IFNZ</b> always evaluates as false if <i>expr</i> references undefined or forward-defined symbols.
<b>\$LISTOFF</b>	Turns off generation of source and object data in the *.LST file for all lines which follow this directive. Useful for excluding the contents of <b>\$INCLUDE</b> files in the *.LST file.
<b>\$LISTON</b>	Enables generation of source and object data in the *.LST file for the source code following this directive. Has no effect if list file generation is disabled ( <b>-L-</b> command line option in effect).

<b>\$MAPOFF</b>	Suppresses generation of source-line information in the *.MAP file for the code following this directive. Symbols which are defined following this directive are still included in the *.MAP file.
<b>\$MAPON</b>	Enables generation of source-line information in the *.MAP file for the code following this directive. <b>\$MAPON</b> is the default state when assembly is started when map file generation is enabled ( <b>-M+</b> command line option).
<b>\$MESSAGE</b> <i>text</i>	Displays <i>text</i> on screen during the first pass of assembly when this directive is encountered in the source. Messages are not written to the error file.
<b>\$OPTRELOFF</b>	Disable "BRA/BSR instead of JMP/JSR" optimization warnings. Equivalent to the <b>-REL-</b> command line option.
<b>\$OPTRELON</b>	Enable warning generation when an absolute branch or subroutine call (JMP or JSR) is encountered that could be successfully implemented using the relative form of the same instruction (BRA or BSR). This option is on by default. Equivalent to the <b>-REL+</b> command line option.
<b>\$OPTRTSOFF</b>	Disable RTS-after-JSR/BSR optimization warning (default). Equivalent to the <b>-RTS-</b> command line option.
<b>\$OPTRTSON</b>	Enable warning generation when a subroutine call (JSR or BSR) is immediately followed by a RTS. This option is off by default. Command-line option <b>-RTS+</b> does the same thing.
<b>\$S19FLUSH</b>	Forces the immediate termination of an S-record line when encountered, rather than waiting for the record to reach the size specified by the <b>-Sn</b> command line directive. This directive may be used to make identification of the end of code blocks easier when viewing the *.S19 file.
<b>\$TABSIZE</b> <i>n</i>	Specifies the field width of tab stops used in the source file. Proper use of this directive ensures that the *.LST files generated by ASM11 are formatted in the same way as your source files appear in your text editor. This directive overrides the setting of the <b>-Tn</b> command line option for the source file(s) in which it is encountered.
<b>\$WARNING</b> [ <i>text</i> ]	Similar to the <b>\$ERROR</b> directive, but generates an assembler warning message instead of an error message.

## Expression Operators and Other Special Characters Recognized by ASM11

- Expressions are evaluated in the order they are written.  
All operators have equal precedence.
- Avoid inserting spaces between values and operators.

<b>Operator</b>	<b>Description</b>
+	Addition
-	Subtraction When used as a unary operator, the 2's complement of the value to the right is returned.
*	Multiplication Can also be used to represent the current location counter.
/	Division
\	Modulus (remainder of integer division)
>	Shift right – operand to the left is shifted right by the count to the right. Also used to specify extended addressing mode.
<	Shift left – operand to the left is shifted left by the count to the right. Also used to specify direct addressing mode.
&	Bitwise AND
	Bitwise OR
^	Bitwise XOR (exclusive OR)
~	Swap high and low bytes (unary): ~\$1234 = \$3412 Useful for converting word values from little-endian to big-endian form
[	Extract low 8 bits (unary): [\$1234 = \$34
]	Extract high 8 bits (unary): ]\$1234 = \$12
\$	Interpret numeric constant that follows as a hexadecimal number. Can also be used to represent the current location counter.
%	Interpret numeric constant that follows as a binary number
' ` ""	Any one of these characters may be used to enclose a string or character entity. The character used at the start of the string must be used to end it.
#	Specifies immediate addressing mode
@	Specifies direct addressing mode (same as "<")

## ASM11 Extended Instruction Set

The instructions listed below are not actually new instructions, rather, internal macros which generate one or more 68HC11 CPU instructions. These instructions are only recognized if the extended instruction set option is enabled (**-X+** command line option or **\$EXTRAON** processing directive).

<b>Mnemonic/Syntax</b>	<b>Description</b>	
<b>LDA</b> <i>operand</i>	Same as:	<b>LDAA</b> <i>operand</i>
<b>LDB</b> <i>operand</i>	Same as:	<b>LDAB</b> <i>operand</i>
<b>STA</b> <i>operand</i>	Same as:	<b>STAA</b> <i>operand</i>
<b>STB</b> <i>operand</i>	Same as:	<b>STAB</b> <i>operand</i>
<b>ORA</b> <i>operand</i>	Same as:	<b>ORAA</b> <i>operand</i>
<b>ORB</b> <i>operand</i>	Same as:	<b>ORAB</b> <i>operand</i>
<b>PSHD</b>	Push D:	<b>PSHB / PSHA</b>
<b>PULD</b>	Pull D:	<b>PULA / PULB</b>
<b>CMPX</b> <i>operand</i>	Same as:	<b>CPX</b> <i>operand</i>
<b>CMPY</b> <i>operand</i>	Same as:	<b>CPY</b> <i>operand</i>
<b>CLRD</b>	Clear D:	<b>CLRA / CLRB</b>
<b>CLR X</b>	Clear X:	<b>LDX #0</b>
<b>CLR Y</b>	Clear Y:	<b>LDY #0</b>
<b>COMD</b>	1's Complement D:	<b>COMA / COMB</b>
<b>NEGD</b>	2's Complement D:	<b>COMA / COMB / ADDD #1</b>
<b>XGAB</b>	Exchange A and B:	<b>PSHA / TBA / PULB</b>
<b>INCD</b>	Increment D:	<b>ADDD #1</b>
<b>DECD</b>	Decrement D:	<b>SUBD #1</b>
<b>LBRA</b> <i>addr16</i>	Long relative branch: <i>(24 bytes/76 cycles)</i> <b>Warning!</b> Generates considerable code, use with care	
<b>LBSR</b> <i>addr16</i>	Long relative subroutine call: <i>(34 bytes/99 cycles)</i> <b>Warning!</b> Generates considerable code, use with care	
<b>CLS</b>	Clear S flag:	<b>PSHA / TPA / ANDA #\$7F / TAP / PULA</b>
<b>CLX</b>	Clear X flag:	<b>PSHA / TPA / ANDA #\$BF / TAP / PULA</b>
<b>SES</b>	Set S flag:	<b>PSHA / TPA / ORAA #\$80 / TAP / PULA</b>
<b>WAIT</b>	Enter WAIT mode:	<b>CLI / WAI</b>
<b>OS</b> <i>byteval</i>	Operating system call:	<b>SWI / DB</b> <i>byteval</i>
<b>OSW</b> <i>wordval</i>	Operating system call:	<b>SWI / DW</b> <i>wordval</i>