ASSEMBLER CROSS REFERENCE

by the Nibble Staff

TIPS 'N TECHNIQUES

Here's a handy *chart to help you convert published assembly listings to work with other assemblers.*

This cross-reference will help guide assembly language programmers through the maze of different pseudo-opcodes used by several popular assemblers. While all assemblers for the Apple // family use the same opcodes for 6502 instructions, such as LDA and STX, the pseudo-ops used vary from one assembler to another.

The table covers most common pseudo-ops that have a direct bearing on the code the assembler generates. Directives that control the output of the assembled listing, such as pagination or disk commands, and special commands for generating relocatable modules are not included.

The final test to determine if you have substituted the proper directive in your source code is to check the object code that is generated by your assembler against the object code that is printed in the magazine. If it doesn't match, you can almost always force the proper code in your program by using the directive DFB or its equivalent.

Assembler Cross-Reference

Function	MicroSPARC Assembler	Merlin/Big Mac Merlin Pro	Apple's DOS Tool Kit	LISA	S-C Macro Assembler
Origin of assembled code	ORG	ORG (see note 1)	ORG	ORG	.OR
Store object code in memory here	See note 2	OBJ (see note 1)	OBJ (optional)	OBJ	.TA
Assign a label	*label* EQU *addr*	*label* EQU *addr* *label* = *addr*	*label* EQU *addr*	*label* EQU *addr*	*label* .EQ *addr*
Assign a page zero label	*label* EQU *addr*	*label* EQU *addr* *label* = *addr*	*label* EQU *addr*	*label* EPZ *addr*	*label* .EQ *addr*
Insert hex bytes AE, BD in memory	DFC $AE,$BD HEX in MACLIB	HEX AE,BD HEX AEBD	DFB $AE,$BD	HEX AEBD	.HS (see note 3) .HS AEBD
Store the address of a label, low byte first	DFC *label,label*/ ADDR in MACLIB	DA *label*	DW *label*	ADR *label*	.DA *label*
Store an address, high byte first	DFC *label*/,*label*	DDB *label*	DDB *label*	DBY *label*	.DA /*label*,#*label*
Define a data value, low byte only	DFC #*label*	DFB *label* DFB <*label* DFB #<*label* DB *label* (Pro only)	DFB >*label* DB >*label*	BYT *label*	.DA #*label*
Define a data value, high byte only	DFC #*label*/	DFB >*label* DFB #>*label* DFB #/*label* DB >*label* (Pro only)	DFB *label* DFB <*label*	HBY *label*	.DA /*label*
Operate on the low byte of a label	Opcode #*label*	Opcode #<*label* Opcode #*label*	Opcode #*label*	Opcode #*label*	Opcode #*label*
Operate on the high byte of a label	Opcode #*label*/	Opcode #>*label* Opcode #/*label*	Opcode #/*label* Opcode #<*label*	Opcode #/*label*	Opcode /*label*
Define a constant	DFC *number*	DFB *number*	DFB *number*	Use HEX or BYT	.DA #*number*
Define storage	DFS	DS (see note 4)	DS	DFS	.BS
Define the start of constant storage	DSC	Use new ORG at start and end	DSECT storage code DEND	Use new ORG at start and end	Use new .PH address at start and .EP at end
Put an ASCII string into object code, high bit set	ASC "*string*"	ASC '*string*' also ( )+? as delimiters (see note 5)	ASC "*string*" also "?.!:)({[}#$%& as delimiters	ASC "string"	.AS −"*string*"
ASCII with high bit clear	DCI "*string*" USE MACLIB Last bit set high	ASC "*string*" also !#$%& as delimiters	MSB OFF ASC "*string*" or other delimiters	ASC '*string*'	.AS "string"
Insert the length of a string, then the string	STR "*string*" USE MACLIB	STR "*string*" STR '*string*' also !#$%& as delimiters	N/A	STR "*string*" STR '*string*'	*labela* .DA #*labelb* .AS /*string*/ *labelb* .EQ* − *labela* − 1
Insert a string with the final byte opposite to others	DCI "*string*" USE MACLIB high bit clear except last	DCI "*string*" also !#$%& as delimiters	DCI "*string*" high bit clear except last	DCI "*string*" DCI '*string*'	.AT "*string*"

Function	MicroSPARC Assembler	Merlin/Big Mac Merlin Pro	Apple's DOS Tool Kit	LISA	S-C Macro Assembler
Insert an inverse string	INV "*string*"	INV "*string*" also !#$%& as delimiters	N/A (see note 6)	INV "*string*"	N/A (see note 6)
Insert a flashing string	FLS "*string*" USE MACLIB	FLS "*string*" also !#$%& as delimiters	N/A (see note 6)	BLK "*string*"	N/A (see note 6)
Extended mnemonics	BLT (USE MACLIB) or BCC	BLT (Pro only) or BCC	BCC	BLT or BCC	BCC
	BGE (USE MACLIB) or BCS	BGE (Pro only) or BCS	BCS	BGE or BCS	BCS
	BTR (USE MACLIB) or BNE	BNE	BNE	BTR or BNE	BNE
	BFL (USE MACLIB) or BEQ	BEQ	BEQ	BFL or BEQ	BEQ
	XOR (USE MACLIB) or EOR	EOR	EOR	XOR or EOR	EOR
Current program counter	. (period) or *(ProDOS version only)	*	*	*	*
Comment in a label field	; or *(ProDOS version only)	*	* or ;	* or ;	* or ;
Comment in another field	;	None required but usually ;	;	;	; or space
Specify constant hex number	*#$number*	*#$number*	*#$number*	*#$number*	*#$number*
Specify constant decimal number	*#number*	*#number*	*#number*	*!#number* or *#number*	*#number*
Octal number	N/A	N/A	*@number*	N/A	N/A
Binary number	*%number*	*%number*	N/A	*%number*	*%number*
Exclusive OR	N/A	!	N/A	^	N/A
OR	!	. (period)	N/A	|	N/A
AND	^	&	N/A	&	N/A
Equal	=	N/A	N/A	=	=
Not equal	#	N/A	N/A	#	N/A
Greater than	>	N/A	N/A	N/A	>
Less than	<	N/A	N/A	N/A	<
Local labels	[*local*]*local* (see note 7)	:*local* (see note 8)	N/A	^ *local* 0-9	.0 to .99
Start of macro definition	MAC	MAC	N/A	N/A (USR available)	.MA
End of macro definition	EMC	EOM or < < <	N/A	N/A	.EM
Put macro in code	*macro.name*	*macro.name* (Pro only) PMC *mac.name* > > > *mac.name*	N/A	N/A	> *macro.name*
Shift Accumulator	ASL LSR	ASL LSR	ASL A LSR A	ASL LSR	ASL LSR
Rotate Accumulator	ROL ROR	ROL ROR	ROL A ROR A	ROL ROR	ROL ROR
Conditional assembly	AIF ALS (else) AEN (end if)	DO ELSE FIN	DO ELSE FIN	.IF .EL (else) .FI (end if)	.DO .ELSE .FIN
Supports 65C02	Yes (ProDOS version)	Yes (Pro only)	Yes (ProDOS version)	No	Yes (v. 2.0)
Supports 65816	No	Yes (Pro only)	No	No	Yes (v. 2.0)
Supports DOS 3.3	Yes (DOS version)	Yes	Yes	Yes	Yes
Supports ProDOS	Yes (ProDOS version)	Yes (Pro only)	No	No	Yes (ProDOS version)

1. Merlin Pro supports relocatable code modules for use with a relocating linker. The ORG and OBJ commands are not used with relocatable modules.

2. The MicroSPARC and Tool Kit Assemblers send the assembled code to disk, so OBJ or its equivalent is not needed. With the MicroSPARC Assembler, you can assemble to memory by changing the assembly default parameters.

3. The S-C Macro Assembler command .HS ignores imbedded periods (.), so they can be used to separate hexadecimal bytes. For example: .HS AE.BD.

4. Merlin Pro supports the syntax "DS /" or "DS \.*number*", where the backslash causes zeros to fill memory to the next memory page.

5. To insert hex bytes after the string, Merlin Pro supports the syntax: ASC "string".8D.

6. The DOS Tool Kit and S-C Macro Assemblers do not have special pseudo-ops for inverse or flashing text. You can translate the characters in the message into hex bytes and use the DFB or .HS directives, respectively.

7. With the MicroSPARC Assembler, local labels can either have forward or backward reference. Always define the label with] in the label field, and in references, use] for forward and [for backward.

8. In Merlin, the]*local* syntax is designed primarily for use in macros and as label variables. It can also be used for forward references in local variables, but not for backward references. In Merlin Pro,]*var* is used as a label variable and :*local* is used as a true local variable, with forward and backward references available.

Reprinted with permission of Nibble Magazine, 45 Winthrop Street, Concord, MA 01742, (617) 371-1660. A 12-issue subscription to Nibble is $26.95. Copyright 1986 by MicroSPARC Inc.