Motorola 68000 Instruction Set (1/2)

Instruction	Description	Assembler Syntax	Data Size	X N Z V C
ABCD	Add BCD with extend	Dx,Dy	B--	* U * U *
		-(Ax),-(Ay)		
ADD	ADD binary	Dn,<ea>	BWL	* * * * *
		<ea>,Dn		
ADDA	ADD binary to An	<ea>,An	-WL	- - - - -
ADDI	ADD Immediate	#x,<ea>	BWL	* * * * *
ADDQ	ADD 3-bit immediate	#<1-8>,<ea>	BWL	* * * * *
ADDX	ADD eXtended	Dy,Dx	BWL	* * * * *
		-(Ay),-(Ax)		
AND	Bit-wise AND	<ea>,Dn	BWL	- * * 0 0
		Dn,<ea>		
ANDI	Bit-wise AND with Immediate	#<data>,<ea>	BWL	- * * 0 0
ASL	Arithmetic Shift Left	#<1-8>,Dy	BWL	* * * * *
		Dx,Dy		
		<ea>		
ASR	Arithmetic Shift Right	...	BWL	* * * * *
Bcc	Conditional Branch	Bcc.S <label>	BW-	- - - - -
		Bcc.W <label>		
BCHG	Test a Bit and CHanGe	Dn,<ea>	B-L	- - * - -
		#<data>,<ea>		
BCLR	Test a Bit and CLeaR	...	B-L	- - * - -
BSET	Test a Bit and SET	...	B-L	- - * - -
BSR	Branch to SubRoutine	BSR.S <label>	BW-	- - - - -
		BSR.W <label>		
BTST	Bit TeST	Dn,<ea>	B-L	- - * - -
		#<data>,<ea>		
CHK	CHecK Dn Against Bounds	<ea>,Dn	-W-	- * U U U
CLR	CLeaR	<ea>	BWL	- 0 1 0 0
CMP	CoMPare	<ea>,Dn	BWL	- * * * *
CMPA	CoMPare Address	<ea>,An	-WL	- * * * *
CMPI	CoMPare Immediate	#<data>,<ea>	BWL	- * * * *
CMPM	CoMPare Memory	(Ay)+,(Ax)+	BWL	- * * * *
DBcc	Looping Instruction	DBcc Dn,<label>	-W-	- - - - -
DIVS	DIVide Signed	<ea>,Dn	-W-	- * * * 0
DIVU	DIVide Unsigned	<ea>,Dn	-W-	- * * * 0
EOR	Exclusive OR	Dn,<ea>	BWL	- * * 0 0
EORI	Exclusive OR Immediate	#<data>,<ea>	BWL	- * * 0 0
EXG	Exchange any two registers	Rx,Ry	--L	- - - - -
EXT	Sign EXTend	Dn	-WL	- * * 0 0
ILLEGAL	ILLEGAL-Instruction Exception	ILLEGAL		- - - - -
JMP	JuMP to Affective Address	<ea>		- - - - -
JSR	Jump to SubRoutine	<ea>		- - - - -
LEA	Load Effective Address	<ea>,An	--L	- - - - -
LINK	Allocate Stack Frame	An,#<displacement>		- - - - -
LSL	Logical Shift Left	Dx,Dy	BWL	* * * 0 *
		#<1-8>,Dy		
		<ea>		
LSR	Logical Shift Right	...	BWL	* * * 0 *
MOVE	Between Effective Addresses	<ea>,<ea>	BWL	- * * 0 0
Instruction	**Description**	**Assembler Syntax**	**Data Size**	**X N Z V C**

Motorola 68000 Instruction Set (2/2)

Instruction	Description	Assembler Syntax	Data Size	X N Z V C
MOVE	To CCR	<ea>,CCR	-W-	I I I I I
MOVE	To SR	<ea>,SR	-W-	I I I I I
MOVE	From SR	SR,<ea>	-W-	- - - - -
MOVE	USP to/from Address Register	USP,An	--L	- - - - -
		An,USP		
MOVEA	MOVE Address	<ea>,An	-WL	- - - - -
MOVEM	MOVE Multiple	<register list>,<ea>		- - - - -
		-WL		
		<ea>,<register list>		
MOVEP	MOVE Peripheral	Dn,x(An)	-WL	- - - - -
		x(An),Dn		
MOVEQ	MOVE 8-bit immediate	#<-128.+127>,Dn	--L	- * * 0 0
MULS	MULtiply Signed	<ea>,Dn	-W-	- * * 0 0
MULU	MULtiply Unsigned	<ea>,Dn	-W-	- * * 0 0
NBCD	Negate BCD	<ea>	B--	* U * U *
NEG	NEGate	<ea>	BWL	* * * * *
NEGX	NEGate with eXtend	<ea>	BWL	* * * * *
NOP	No OPeration	NOP		- - - - -
NOT	Form one's complement	<ea>	BWL	- * * 0 0
OR	Bit-wise OR	<ea>,Dn	BWL	- * * 0 0
		Dn,<ea>		
ORI	Bit-wise OR with Immediate	#<data>,<ea>	BWL	- * * 0 0
PEA	Push Effective Address	<ea>	--L	- - - - -
RESET	RESET all external devices	RESET		- - - - -
ROL	ROtate Left	#<1-8>,Dy	BWL	- * * 0 *
		Dx,Dy		
		<ea>		
ROR	ROtate Right	...	BWL	- * * 0 *
ROXL	ROtate Left with eXtend	...	BWL	* * * 0 *
ROXR	ROtate Right with eXtend	...	BWL	* * * 0 *
RTE	ReTurn from Exception	RTE		I I I I I
RTR	ReTurn and Restore	RTR		I I I I I
RTS	ReTurn from Subroutine	RTS		- - - - -
SBCD	Subtract BCD with eXtend	Dx,Dy	B--	* U * U *
		-(Ax),-(Ay)		
Scc	Set to -1 if True, 0 if False	<ea>	B--	- - - - -
STOP	Enable & wait for interrupts	#<data>		I I I I I
SUB	SUBtract binary	Dn,<ea>	BWL	* * * * *
		<ea>,Dn		
SUBA	SUBtract binary from An	<ea>,An	-WL	- - - - -
SUBI	SUBtract Immediate	#x,<ea>	BWL	* * * * *
SUBQ	SUBtract 3-bit immediate	#<data>,<ea>	BWL	* * * * *
SUBX	SUBtract eXtended	Dy,Dx	BWL	* * * * *
		-(Ay),-(Ax)		
SWAP	SWAP words of Dn	Dn	-W-	- * * 0 0
TAS	Test & Set MSB & Set N/Z-bits	<ea>	B--	- * * 0 0
TRAP	Execute TRAP Exception	#<vector>		- - - - -
TRAPV	TRAPV Exception if V-bit Set	TRAPV		- - - - -
TST	TeST for negative or zero	<ea>	BWL	- * * 0 0
UNLK	Deallocate Stack Frame	An		- - - - -
Instruction	**Description**	**Assembler Syntax**	**Data Size**	**X N Z V C**

Symbol	Meaning
*	Set according to result of operation
-	Not affected
0	Cleared
1	Set
U	Outcome (state after operation) undefined
I	Set by immediate data
<ea>	Effective Address Operand
<data>	Immediate data
<label>	Assembler label
<vector>	TRAP instruction Exception vector (0-15)
<rg.lst>	MOVEM instruction register specification list
<displ.>	LINK instruction negative displacement
...	Same as previous instruction

Legend	
Dn	Data Register (n is 0-7)
An	Address Register (n is 0-7)
b	08-bit constant
w	16-bit constant
l	32-bit constant
x	8-, 16-, 32-bit constant
Rx	Index Register Specification, one of:
Dn.W	Low 16 bits of Data Register
Dn.L	All 32 bits of Data Register
An.W	Low 16 bits of Address Register
An.L	All 32 bits of Address Register

Condition Codes for Bcc Instructions. Condition Codes set after CMP D0,D1 Instruction		
Relationship	**Unsigned**	**Signed**
D1 < D0	CS – Carry Bit Set	LT - Less Than
D1 <= D0	LS – Lower or Same	LE - Less than or Equal
D1 = D0	EQ – Equal (Z-bit Set)	EQ - Equal (Z-bit Set)
D1 != D0	NE - Not Equal (Z-bit Clear)	NE - Not Equal (Z-bit Clear)
D1 > D0	HI - HIgher than	GT - Greater Than
D1 >= D0	CC – Carry Bit Clear	GE - Greater than or Equal
	PL - PLus (N-bit Clear)	
	MI – Minus (N-bit Set)	
	VC - V-bit Clear (No Overflow)	
	VS - V-bit Set (Overflow)	