

## Motorola

## 68000 MICROPROCESSOR Instruction Set Summary

<--> D4	- 1	64	- D5	<-->
<--> D3	- 2	63	- D6	<-->
<--> D2	- 3	62	- D7	<-->
<--> D1	- 4	61	- D8	<-->
<--> D0	- 5	60	- D9	<-->
<-- ~AS	- 6	59	- D10	<-->
<-- ~UDS	- 7	58	- D11	<-->
<-- ~LDS	- 8	57	- D12	<-->
<-- R/~W	- 9	56	- D13	<-->
--> ~DTACK	- 10	55	- D14	<-->
<-- ~BG	- 11	54	- D15	<-->
--> ~BGACK	- 12	53	- GND	
--> ~BR	- 13	52	- A23	-->
Vcc	- 14	51	- A22	-->
--> CLK	- 15	50	- A21	-->
GND	- 16	49	- Vcc	
		48	- A20	-->
<--> ~HALT	- 17	47	- A19	-->
<--> ~RESET	- 18	46	- A18	-->
<-- ~VMA	- 19	45	- A17	-->
<-- E	- 20	44	- A16	-->
--> ~VPA	- 21	43	- A15	-->
--> ~BERR	- 22	42	- A14	-->
--> ~IPL2	- 23	41	- A13	-->
--> ~IPL1	- 24	40	- A12	-->
--> ~IPL0	- 25	39	- A11	-->
<-- FC2	- 26	38	- A10	-->
<-- FC1	- 27	37	- A9	-->
<-- FC0	- 28	36	- A8	-->
<-- A1	- 29	35	- A7	-->
<-- A2	- 30	34	- A6	-->
<-- A3	- 31	33	- A5	-->
<-- A4	- 32			

Written by Jonathan Bowen  
 Programming Research Group  
 Oxford University Computing Laboratory  
 8-11 Keble Road  
 Oxford OX1 3QD  
 England

Tel +44-865-273840

Created January 1983

Updated April 1985

Issue 1.4

Copyright (C) J.P.Bowen 1985

Mnemonic		XNZVC	BWL	Description	Notes
ABCD	s,d	*?***	X	Add BCD format	d=BCD{d+s+X}
ADD	s,d	*****	XXX	Add binary	d=d+s
ADDA	s,An	-----	XX	Add Address	An=An+s
ADDI	#e,d	*****	XXX	Add Immediate	d=d+e
ADDQ	#q,d	*****	XXX	Add Quick	d=d+q
ADDX	s,d	*****	XXX	Add Extended	d=d+s+X
AND	s,d	-**00	XXX	Logical AND	d=d&s
ANDI	#e,d	-**00	XXX	Logical AND Immediate	d=d&e
ASLr	d	*****	XXX	Arithmetic Shift	d=d*2 or d=d/2
Bcc	l	-----	XX	Branch conditionally	If cc BRA
BCHG	s,d	--*--	XX	Bit test and Change	BTST,d<s>=Z
BCLR	d	--*--	XX	Bit test and Clear	BTST,d<s>=0
BRA	l	-----	XX	Branch Always	PC=l
BSET	d	--*--	XX	Bit test and Set	BTST,d<s>=1
BSR	l	-----	XX	Branch to Subroutine	-[SP]=PC,PC=l
BTST	d	--*--	XX	Bit Test	Z=~d<s>
CHK	s,Dn	-*???	X	Check register	If 0>Dn>s \$[18H]
CLR	d	-0100	XXX	Clear operand	d=0
CMP	s,Dn	-****	XXX	Compare	Dn-s
CMPA	s,An	-****	XXX	Compare Address	An-s
CMPI	#e,d	-****	XXX	Compare Immediate	d-e
CMPM	s,d	-****	XXX	Compare Memory	d-s
DBcc	Dn,l	-----		Decrement and Branch	If~cc&Dn-1~-1 BRA
DIVS	s,Dn	-***0	X	Signed Division	Dn={Dn%s,Dn/s}
DIVU	s,Dn	-***0	X	Unsigned Division	Dn={Dn%s,Dn/s}
EOR	Dn,d	-**00	XXX	Exclusive OR	d=dxDn
EORI	#e,d	-**00	XXX	Exclusive OR Immediate	d=dxe
EXG	r,r	-----	X	Exchange registers	r<->r
EXT	Dn	-**00	XX	Extend sign	Dn<hi>=Dn<7or15>
JMP	d	-----		Jump	PC=d
JSR	d	-----		Jump to Subroutine	-[SP]=PC,PC=d
LEA	s,An	-----	X	Load Effective Address	An=EA{s}
LINK	An,#nn	-----		Link and allocate	-[SP]=An=SP=SP+nn
LSLr	d	***0*	XXX	Logical Shift	d->{C,d,0}<-
MOVE	s,d	-**00	XXX	Move data	d=s
MOVE	s,CCR	*****	X	Move to CCR	CCR=s
MOVE	s,SR	*****	X	Move to SR	SR=s
MOVE	SR,d	-----	X	Move from SR	d=SR
MOVE	USP,An	-----	X	Move User SP	USP=An or An=USP
MOVEA	s,An	-----	XX	Move Address	An=s
MOVEM	s,d	-----	XX	Move Multiple register	rr=s or d=rr
MOVEP	s,d	-----	XX	Move Peripheral data	d=Dn or Dn=s
MOVEQ	#q,d	-**00	X	Move Quick	d=q
MULS	s,Dn	-**00	X	Signed Multiply	Dn<0:31>=Dn*s
MULU	s,Dn	-**00	X	Unsigned Multiply	Dn<0:31>=Dn*s
NBCD	d	*?***	X	Negate BCD format	d=BCD{-d-X}
NEG	d	*****	XXX	Negate	d=-d
NEGX	d	*****	XXX	Negate with Extend	d=-d-X
NOP		-----		No Operation	
NOT	d	-**00	XXX	Logical NOT	d=~d
OR	s,d	-**00	XXX	Inclusive OR	d=dvs

ORI	#e,d	-**00	XXX	Inclusive OR Immediate	d=dve
PEA	s	-----	X	Push Effective Address	-[SP]=EA{s}
RESET		-----		Reset external devices	Reset line=0
ROlr	d	-**0*	XXX	Rotate	d-->{d}<-
ROXlr	d	**0*	XXX	Rotate with Extend	d-->{d}<- ,X=C
RTE		*****		Return from Exception	SR=[SSP]+,RTS
RTR		*****		Return and Restore	SR<0:4>=[SP]+,RTS
RTS		-----		Return from Subroutine	PC=[SP]+
SBCD	s,d	*?***	X	Subtract BCD format	d=BCD{d-s-X}
Scc	d	-----	X	Set conditionally	d=0 or d=-1
STOP	#nn	*****		Load status and Stop	SR=nn, wait
SUB	s,d	*****	XXX	Subtract binary	d=d-s
SUBA	s,An	-----	XX	Subtract Address	An=An-s
SUBI	#e,d	*****	XXX	Subtract Immediate	d=d-e
SUBQ	#q,d	*****	XXX	Subtract Quick	d=d-q
SUBX	s,d	*****	XXX	Subtract with Extend	d=d-s-X
SWAP	Dn	-**00	X	Swap register halves	Dn<hi><->Dn<lo>
TAS	d	-**00	X	Test And Set	d<7>=1
TRAP	#n	-----		Trap (n=0-15)	\$(80H+4*n]
TRAPV		-----		Trap on Overflow	If V=1 \$(1CH]
TST	d	-**00	XXX	Test	d
UNLK	An	-----		Unlink	SP=An,An=[SP]+
-----					
DC	e(,....)		XXX	Define Constant	
DS	e		XXX	Define Storage	

Mnemonic	XNZVC	BWL	Description
CCR	-*01?		Unaffected/affected/reset/set/unknown
T			Trace mode flag (Bit 15)
S			Supervisor/user mode select (Bit 13)
In			Interrupt mask flag #n (Bits 8-10,n=0-2)
X	X		Extend flag (Bit 4)
N	N		Negative flag (Bit 3)
Z	Z		Zero flag (Bit 2)
V	V		Overflow flag (Bit 1)
C	C		Carry flag (Bit 0)
-----			
.B		X	Byte attribute (8-bit, .S for branch)
.W		X	Word attribute (16-bit)
.L		X	Long word attribute (32-bit)
-----			
Dn			Data register direct addressing
An			Address register direct addressing
[An]			Register indirect addressing
[An]+			Post-increment register indirect addr.
-[An]			Pre-decrement register indirect addr.
n[An]			Offset register indirect addressing
n[An,r]			Index register indirect addressing
nn			Short absolute data addressing
nnnn			Long absolute data addressing
nn			Program counter relative addressing
nn[r]			Program counter with index addressing

#e	Immediate data addressing
ABSOLUTE_LONG	Long Absolute addressing (or ABS_LONG)
ABSOLUTE_SHORT	Short Absolute addressing (or ABS_SHORT)
EVEN	Set program counter to Even address
NO_RORG	Disable Relative addressing (or PC_DEP)
RORG	Enable Relative addressing (or PC_INDEP)
An	Address register (16/32-bit, n=0-7)
CCR	Condition Code Register (8-bit, low SR)
Dn	Data register (8/16/32-bit, n=0-7)
PC	Program Counter (24-bit)
SP	Active Stack Pointer (equivalent to A7)
SR	Status Register (16-bit)
SSP	Supervisor Stack Pointer (32-bit)
USP	User Stack Pointer (32-bit)
BCD{ } EA{ }	Binary Coded Decimal/Effective Address
cc	Condition = (T/F/HI/LS/CC/CS/NE/EQ/ VC/VB/PL/MI/GE/LT/GT/LE)
d s	Destination/source
e n nn nnnn	Any/8-bit/16-bit/32-bit expression
l	Branch displacement label (8/16-bit)
lr	Left/right direction = (L/R)
q	Quick expression (1-8)
r	Any register An or Dn
rr	Multiple registers (-=range,/=separator)
+ - * / %	Add/subtract/multiply/divide/remainder
& ~ v x	AND/NOT/inclusive OR/exclusive OR
->{ }<- <->	Rotate left or right/exchange operands
[ ] -[ ] [ ]+	Indirect/autoincrement/autodecr. address
< > <:> <hi> <lo>	Bit number/bit range/high half/low half
{ } {,}	Combination of operands
\$	Software trap -[SP]=PC,-[SP]=SR,PC=...
0000H to 0007H	Reset vector (initial SSP and PC) (0-1)
0008H to 000BH	Bus error vector (2)
000CH to 000FH	Address error vector (3)
0010H to 0013H	Illegal instruction vector (4)
0014H to 0017H	Zero divide vector (5)
0018H to 001BH	CHK instruction vector (6)
001CH to 001FH	TRAPV instruction vector (7)
0020H to 0023H	Privilege violation vector (8)
0024H to 0027H	Trace vector (9)
0028H to 002FH	Line 1010/1111 emulator vectors (10-11)
003CH to 003FH	Uninitialised interrupt vector (15)
0060H to 0063H	Spurious interrupt vector (24)
0064H to 007FH	Level 1-7 interrupt auto-vectors (25-31)
0080H to 00BFH	TRAP #0-15 instruction vectors (32-47)
0100H to 03FFH	Unassigned, reserved (12-14,16-23,48-63)
	User interrupt vectors (64-255)