Meaning of M68000 Instructions

MNEMONIC	DESCRIPTION	OPERATION	X	N	Z	V	C
ABCD	Add decimal with extend	(Destination)$_{10}$ + (source)$_{10}$ + x → destination	●	U	●	U	●
ADD	Add binary	(Destination) + (source) → destination	●	●	●	●	●
ADDA	Add address	(Destination) + (source) → destination	—	—	—	—	—
ADDI	Add immediate	(Destination) + immediate data → destination	●	●	●	●	●
ADDQ	Add quick	(Destination) + immediate data → destination	●	●	●	●	●
ADDX	Add extended	(Destination) + (source) + x → destination	●	●	●	●	●
AND	AND logical	(Destination) Λ (source) → destination	—	●	●	0	0
ANDI	AND immediate	(Destination) Λ immediate data → destination	—	●	●	0	0
ASL, ASR	Arithmetic shift	(Destination) shifted by ⟨count⟩ → destination	●	●	●	●	●
B$_{CC}$	Branch conditionally	If $_{CC}$ then PC + d → PC	—	—	—	—	—
BCHG	Test a bit and change	~(⟨bit number⟩) OF destination → Z ~(⟨bit number⟩) OF destination → ⟨bit number⟩ OF destination	—	—	●	—	—
BCLR	Test a bit and clear	~(⟨bit number⟩) OF destination → Z 0 → ⟨bit number⟩ OF destination	—	—	●	—	—
BRA	Branch always	PC + displacement → PC	—	—	—	—	—
BSET	Test a bit and set	~(⟨bit number⟩) OF destination → Z 1 → ⟨bit number⟩ OF destination	—	—	●	—	—
BSR	Branch to subroutine	PC → − (SP), PC + d → PC	—	—	—	—	—
BTST	Test a bit	~(⟨bit number⟩) OF destination → Z	—	—	●	—	—
CHK	Check register against bounds	If Dn ⟨0 or Dn⟩(⟨ea⟩) then TRAP	—	●	U	U	U
CLR	Clear an operand	0 → Destination	—	0	1	0	0
CMP	Compare	(Destination) − (source)	—	●	●	●	●
CMPA	Compare address	(Destination) − (source)	—	●	●	●	●
CMPI	Compare immediate	(Destination) − immediate data	—	●	●	●	●
CMPM	Compare memory	(Destination) − (source)	—	●	●	●	●
DB$_{CC}$	Test condition, decrement, and branch	If ~$_{CC}$ then Dn − 1 → Dn; if Dn ≠ −1 then PC + d → PC	—	—	—	—	—
DIVS	Signed divide	(Destination)/(source) → destination	—	●	●	●	0
DIVU	Unsigned divide	(Destination)/(source) → destination	—	●	●	●	0
EOR	Exclusive OR logical	(Destination) ⊕ (source) → destination	—	●	●	0	0
EORI	Exclusive OR immediate	(Destination) ⊕ immediate data → destination	—	●	●	0	0
EXG	Exchange register	Rx ⟷ Ry	—	—	—	—	—
EXT	Sign extend	(Destination) sign-extended → destination	—	●	●	0	0
JMP	Jump	Destination → PC	—	—	—	—	—
JSR	Jump to subroutine	PC → − (SP); destination → PC	—	—	—	—	—
LEA	Load effective address	Destination → An	—	—	—	—	—
LINK	Link and allocate	An → − (SP); SP → An; SP + displacement → SP	—	—	—	—	—
LSL, LSR	Logical shift	(Destination) shifted by ⟨count⟩ → destination	●	●	●	0	●
MOVE	Move data from source to destination	(Source) → destination	—	●	●	0	0
MOVE to CCR	Move to condition code	(Source) → CCR	●	●	●	●	●
MOVE to SR	Move to the status register	(Source) → SR	●	●	●	●	●
MOVE from SR	Move from the status register	SR → destination	—	—	—	—	—
MOVE USP	Move user stack pointer	USP → An or An → USP	—	—	—	—	—
MOVEA	Move address	(Source) → destination	—	—	—	—	—
MOVEM	Move multiple registers	Registers → destination (Source) → registers	—	—	—	—	—
MOVEP	Move peripheral data	(Source) → destination	—	—	—	—	—
MOVEQ	Move quick	Immediate data → destination	—	●	●	0	0
MULS	Signed multiply	(Destination) × (source) → destination	—	●	●	0	0
MULU	Unsigned multiply	(Destination) × (source) → destination	—	●	●	0	0
NBCD	Negate decimal with extend	0 − (Destination)$_{10}$ − x → destination	●	U	●	U	●
NEG	Negate	0 − (Destination) → destination	●	●	●	●	●
NEGX	Negate with extend	0 − (Destination) − x → destination	●	●	●	●	●
NOP	No operation	—	—	—	—	—	—
NOT	Logical complement	~(Destination) → destination	—	●	●	0	0
OR	Inclusive OR logical	(Destination) V (source) → destination	—	●	●	0	0
ORI	Inclusive OR immediate	(Destination) V immediate data → destination	—	●	●	0	0

MNEMONIC	DESCRIPTION	OPERATION	X	N	Z	V	C
PEA	Push effective address	Destination $\rightarrow$ − (SP)	—	—	—	—	—
RESET	Reset external devices	—	—	—	—	—	—
ROL, ROR	Rotate (without extend)	(Destination) rotated by $\langle$count$\rangle$ $\rightarrow$ destination	—	●	●	0	●
ROXL, ROXR	Rotate with extend	(Destination) rotated by $\langle$count$\rangle$ $\rightarrow$ destination	●	●	●	0	●
RTE	Return from exception	(SP) + $\rightarrow$ SR; (SP) + $\rightarrow$ PC	●	●	●	●	●
RTR	Return and restore condition codes	(SP) + $\rightarrow$ CC; (SP) + $\rightarrow$ PC	●	●	●	●	●
RTS	Return from subroutine	(SP) + $\rightarrow$ PC	—	—	—	—	—
SBCD	Subtract decimal with extend	$(\text{Destination}_{10}) - (\text{source})_{10} - x \rightarrow$ destination	●	U	●	U	●
S_{CC}	Set according to condition	If $_{cc}$ then 1's $\rightarrow$ destination else 0's $\rightarrow$ destination	—	—	—	—	—
STØP	Load status register and stop	Immediate data $\rightarrow$ SR; STOP	●	●	●	●	●
SUB	Subtract binary	(Destination) − (source) $\rightarrow$ destination	●	●	●	●	●
SUBA	Subtract address	(Destination) − (source) $\rightarrow$ destination	—	—	—	—	—
SUBI	Subtract immediate	(Destination) − immediate data $\rightarrow$ destination	●	●	●	●	●
SUBQ	Subtract quick	(Destination) − immediate data $\rightarrow$ destination	●	●	●	●	●
SUBX	Subtract with extend	(Destination) − (source) − x $\rightarrow$ destination	●	●	●	●	●
SWAP	Swap register halves	Register (31:16) $\longleftrightarrow$ register (15:0)	—	●	●	0	0
TAS	Test and set an operand	(Destination) tested $\rightarrow$ CC; 1 $\rightarrow$ [7] OF destination	—	●	●	0	0
TRAP	Trap	PC $\rightarrow$ − (SSP); SR $\rightarrow$ − (SSP); (vector) $\rightarrow$ PC	—	—	—	—	—
TRAPV	Trap on overflow	If V then TRAP	—	—	—	—	—
TST	Test an operand	(Destination) tested $\rightarrow$ CC	—	●	●	0	0
UNLK	Unlink	An $\rightarrow$ SP; (SP) + $\rightarrow$ An	—	—	—	—	—

$\oplus$	Logical exclusive OR	●	Affected
$\wedge$	Logical AND	—	Unaffected
$\vee$	Logical OR	0	Cleared
~	Logical complement	1	Set
		U	Undefined