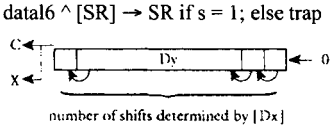
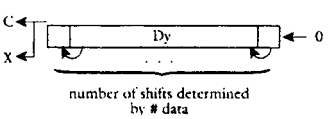
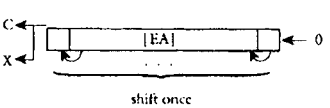
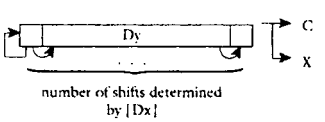
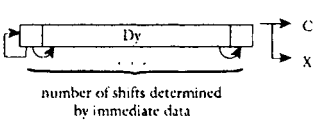
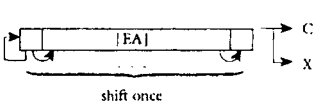
Fundamentals of Digital Logic and Microcomputer Design. M. Rafiquzzaman

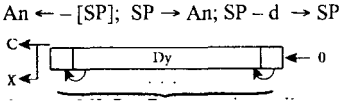
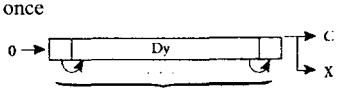
APPENDIX

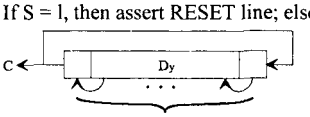
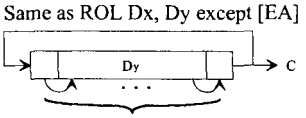
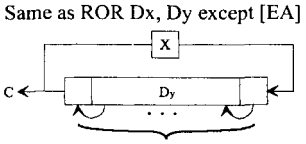
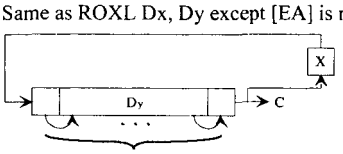
G

68000 INSTRUCTION SET

Instruction	Size	Length (words)	Operation
ABCD – (Ay), – (Ax)	B	1	– [Ay] 10 + – [Ax] 10 + X → [Ax]
ABCD Dy, Dx	B	1	[Dy]10 + [Dx]10 +X → Dx
ADD (EA), (EA)	B,W, L	1	[EA] + [EA] → EA
ADDA (EA), An	W, L	1	[EA] + An → An
ADDI #data, (EA)	B,W, L	2 for B, W 3 for L	data + [EA] → EA
ADDQ #data, (EA)	B, W, L	1	data + [EA] → EA
ADDX – (Ay), – (Ax)	B, W, L	1	– [Ay] + – [Ax] + X → [Ax]
ADDX Dy, Dx	B, W, L	1	Dy + Dx + X → Dx
AND (EA), (EA)	B, W, L	1	[EA] ^ [EA] → EA
ANDI #data, (EA)	B,W, L	2 for B, W 3 for L	data ^ [EA] → EA
ANDI #data8, CCR	B	2	data8 ^ [CCR] → CCR
ANDI #data16, SR	W	2	data16 ^ [SR] → SR if s = 1; else trap
ASL Dx, Dy	B, W, L	1	
ASL #data, Dy	B, W, L	1	
ASL (EA)	B, W, L	1	
ASR Dx, Dy	B, W, L	1	
ASR #data, Dy	B, W, L	1	
ASR (EA)	B,W, L	1	

Instruction	Size	Length (words)	Operation
BCC d	B, W	1 for B 2 for W	Branch to PC + d if carry = 0; else next instruction
BCHG Dn, (EA)	B, L	1	[bit of [EA], specified by Dn]' → Z [bit of [EA] specified by Dn]' → bit of [EA]
BCHG #data. (EA)	B, L	2	Same as BCHG Dn, [EA] except bit number is specified by immediate data
BCLR Dn (EA)	B, L	1	[bit of [EA]]' → Z 0 → bit of [EA] specified by Dn
BCLR #data, (EA)	B, L	2	Same as BCLR Dn, [EA] except the bit is specified by immediate data
BCS d	B, W	1 for B 2 for W	Branch to PC + d if carry = 1; else next instruction
BEQ d	B, W	1 for B 2 for W	Branch to PC + d if Z = 1; else next instruction
BGE d	B, W	1 for B 2 for W	Branch to PC + d if greater than or equal; else next instruction
BGT d	B, W	1 for B 2 for W	Branch to PC + d if greater than; else next instruction
BHI d	B, W	1 for B 2 for W	Branch to PC + d if higher; else next instruction
BLE d	B, W	1 for B 2 for W	Branch to PC + d if less or equal; else next instruction
BLS d	B, W	1 for B 2 for W	Branch to PC + d if low or same; else next instruction
BLT d	B, W	1 for B 2 for W	Branch to PC + d if less than; else next instruction
BMI d	B,W	1 for B 2 for W	Branch to PC +d if N = 1; else next instruction
BNE d	B, W	1 for B 2 for W	Branch to PC +d if Z = 0; else next instruction
BPL d	B, W	1 for B 2 for W	Branch to PC + d if N = 0; else next instruction
BRA d	B, W	1 for B 2 for W	Branch always to PC + d
BSET Dn, (EA)	B, L	1	[bit of [EA]]' → Z 1 → bit of [EA] specified by Dn
BSET #data, (EA)	B, L	2	Same as BSET Dn, [EA] except the bit is specified by immediate data
BSR d	B, W	1 for B 2 for W	PC → – [SP] PC + d → PC
BTST Dn, (EA)	B, L	1	[bit of [EA] specified by Dn]' → Z
BTST #data, (EA)	B, L	2	Same as BTST Dn, [EA] except the bit is specified by data
BVC d	B,W	1 for B 2 for W	Branch to PC + d if V = 0; else next instruction
BVS d	B, W	1 for B 2 for W	Branch to PC + d if V = 1; else next instruction
CHK (EA), Dn	W	1	If Dn < 0 or Dn > [EA], then trap
CLR(EA)	B, W, L	1	0 → EA
CMP (EA), Dn	B, W, L	1	Dn – [EA] → Affect all condition codes except X
CMP (EA), An	W, L	1	An – [EA] → Attect all condition codes except X
CMPI #data, (EA)	B, W, L	2 for B, W 3 for L	[EA] – data → Affect all flags except X-bit

Instruction	Size	Length (words)	Operation
CMPM (Ay) +, (Ax) +	B, W, L	1	[Ax]+ - [Ay]+ → Affect all flags except X; update Ax and Ay
DBCC Dn, d	W	2	If condition false, i.e., C = 1, then Dn - 1 —> Dn; if Dn ≠ – 1, then PC + d → PC; else PC + 2 → PC
DBCS Dn, d	W	2	Same as DBCC except condition is C = 1
DBEQ Dn, d	W	2	Same as DBCC except condition is Z = 1
DBF Dn, d	W	2	Same as DBCC except condition is always false
DBGE Dn, d	W	2	Same as DBCC except condition is greater or equal
DBGT Gn, d	W	2	Same as DBCC except condition is greater than
DBHIDn, d	\V	2	Same as DBCC except condition is high
DBLE Dn, d	W	2	Same as DBCC except condition is less than or equal
DBLS Dn, d	W	2	Same as DBCC except condition is low or same
DBLT Dn, d	W	2	Same as DBCC except condition is less than
DBM1 Dn, d	W	2	Same as DBCC except condition is N = 1
DBNE Dn, d	W	2	Same as DBCC except condition Z = 0
DBPL Dn, d	W	2	Same as DBCC except condition N = 0
DBT Dn, d	W	2	Same as DBCC except condition is always true
DBVC Dn, d	W	2	Same as DBCC except condition is V = 0
DBVS Dn, d	W	2	Same as DBCC except condition is V = 1
DIVS (EA), Dn	W	1	Signed division [Dn]32/[EA]16 → [Dn] 0-15 = quotient [Dn] 16-31 = remainder
DIVU (EA), Dn	W	1	Same as DIVS except division is unsigned
EOR Dn, (EA)	B,W, L	1	Dn ⊕ [EA] → EA
EORI #data, (EA)	B,W, L	2 for B, W 3 for L	data ⊕ [EA] → EA
EORI #d8, CCR	B	2	d8 ⊕ CCR → CCR
EORI #dl6, SR	W	2	dl6 ⊕ SR → SR if S = 1; else trap
EXG Rx, Ry	L	1	Rx ↔ Ry
EXTDn	W, L	1	Extend sign bit of Dn from 8-bit to 16-bit or from 16-bit to 32-bit depending on whether the operand size is B or W
JMP (EA)	Unsized	1	[EA] → PC Unconditional jump using address in operand
JSR (EA)	Unsized	1	PC → – [SP]; [EA] → PC Jump to subroutine using address in operand
LEA (EA), An	L	1	[EA] → An
LINK An, # -d	Unsized	2	An ← – [SP]; SP → An; SP – d → SP
LSL Dx, Dy	B,W, L	1	
LSL #data, Dy	B,W, L	1	Same as LSL Dx, Dy except immediate data specify the number of shifts from 0 to 7
LSL (EA)	B,W, L	1	Same as LSL Dx, Dy except left shift is performed only once
LSR Dx, Dy	B,W, L	1	
LSR #data, Dy	B,W, L	1	Same as LSR except immediate data specifies the number of shifts from 0 to 7
LSR (EA)	B,W,L	1	Same as LSR, Dx, Dy except the right shift is performed only once

Instruction	Size	Length (words)	Operation
MOVE (EA), (EA)	B,W, L	1	[EA] source → [EA] destination
MOVE (EA), CCR	W	1	[EA] → CCR
MOVE CCR, (EA)	W	1	CCR → [EA]
MOVE (EA), SR	W	1	If S = 1, then [EA] → SR; else TRAP
MOVE SR, (EA)	W	1	If S = 1, then SR → [EA]; else TRAP
MOVE An, USP	L	1	If S = 1, then An → USP; else TRAP
MOVE USP, An	W, L	1	[USP] → An
MOVEM register list, (EA)	W,L	2	Register list → [EA]
MOVEM (EA), register list	W, L	2	[EA] → register list
MOVEP Dx, d (Ay)	W, L	2	Dx → d[Ay]
MOVEP d (Ay), Dx	W, L	2	d[Ay] → Dx
MOVEQ #d8, Dn	L	1	d8 sign extended to 32-bit → Dn
MULS(EA)16, (Dn)16	W	1	Signed 16 × 16 multiplication [EA]16 * [Dn]16 → [Dn]32
MULU(EA)16, (Dn)16	W	1	Unsigned 16 × 16 multiplication [EA]16 * [Dn]16 → [Dn]32
NBCD (EA)	B	1	0 – [EA]10 – X → EA
NEC (EA)	B,W, L	1	0 – [EA] → EA
NEGX (EA)	B,W, L	1	0 – [EA] – X → EA
NOP	Unsized	1	No operation
NOT (EA)	B,W, L	1	[EA]' → EA
OR (EA), (EA)	B,W, L	1	[EA]V[EA] → EA
ORI #data, (EA)	B,W, L	2 for B, W 3 for L	data V[EA] → EA
ORI #d8, CCR	B	2	d8VCCR → CCR
ORI #dl6, SR	W	2	If S = 1, then dl6VSR -> SR; else TRAP
PEA (EA)	L	1	[EA] 16 sign extend to 32 bits → – [SP]
RESET	Unsized	1	If S = 1, then assert RESET line; else TRAP
ROL Dx, Dy	B, W, L	1	
ROL #data, Dy	B, W, L	1	Same as ROL Dx, Dy except immediate data specifies number of times to be rotated from 0 to 7
ROL (EA)	B, W, L	1	Same as ROL Dx, Dy except [EA] is rotated once
ROR Dx, Dy	B, W, L	1	
ROR #data, Dy	B, W, L	1	Same as ROR Dx, Dy except the number of rotates is specified by immediate data from 0 to 7
ROR (EA)	B, W, L	1	Same as ROR Dx, Dy except [EA] is rotated once
ROXL Dx, Dy	B, W, L	1	
ROXL #data, Dy	B, W, L	1	Same as ROXL Dx, Dy except immediate data specifies number of rotates from 0 to 7
ROXL (EA)	B, W, L	1	Same as ROXL Dx, Dy except [EA] is rotated once
ROXR Dx, Dy	B, W, L	1	

Instruction	Size	Length (words)	Operation
ROXR #data, Dy	B,W, L	1	Same as ROXR Dx, Dy except immediate data specifies number of rotates from 0 to 7
ROXR (EA)	B,W, L	1	Same as ROXR Dx, Dy except [EA] is rotated once
RTE	Unsized	1	If S = 1, then [SP] + → SR; [SP] + → PC, else TRAP
RTR	Unsized	1	[SP] + → CC; [SP] + → PC
RTS	Unsized	1	[SP] + → PC
SBCD -(Ay), -(Ax)	B	1	– (Ax)10 – (Ay)10 – X → (Ax)
SBCD Dy, Dx	B	1	[Dx]10 – [Dy]10 – X → Dx
SCC (EA)	B	1	If C = 0, then 1s → [EA] else 0s → [EA]
SCS (EA)	B	1	Same as SCC except the condition is C = 1
SEQ (EA)	B	1	Same as SCC except if Z = 1
SF (EA)	B	1	Same as SCC except condition is always false
SGE (EA)	B	1	Same as SCC except if greater or equal
SGT (EA)	B	1	Same as SCC except if greater than
SHI (EA)	B	1	Same as SCC except if high
SLE (EA)	B	1	Same as SCC except if less or equal
SLS(EA)	B	1	Same as SCC except if low or same
SLT (EA)	B	1	Same as SCC except if less than
SMI (EA)	B	1	Same as SCC except if N = 1
SNE (EA)	B	1	Same as SCC except if Z = 0
SPL(EA)	B	1	Same as SCC except if N = 0
ST (EA)	B	1	Same as SCC except condition always true
STOP #data	Unsized	2	If S= 1, then data → SR and stop; TRAP if executed in user mode
SUB (EA), (EA)	B, W, L	1	[EA] – [EA] → EA
SUBA (EA), An	W,L	1	An – [EA] → An
SUBI #data, (EA)	B, W, L	2 for B, W 3 for L	[EA] – data → EA
SUBQ #data, (EA)	B, W, L	1	[EA] – data → EA
SUBX – (Ay), – (Ax)	B, W, L	1	– [Ax] – [Ay] – X → [Ax]
SUBX Dy, Dx	B, W, L	1	Dx – Dy – X → Dx
SVC (EA)	B	1	Same as SCC except if V = 0
SVS (EA)	B	1	Same as SCC except if V = 1
SWAP Dn	W	1	Dn [31:16] ↔ Dn [15:0]
TAS (EA)	B	1	[EA] tested; N and Z are affected accordingly; 1 → bit 7 of [EA]
TRAP #vector	Unsized	1	PC → – [SSP], SR → – [SSP], (vector) → PC; 16 TRAP
TRAPV	Unsized	1	If V = 1, then TRAP; else next instruction
TST (EA)	B,W, L	1	[EA] – 0 → condition codes affected; no result provided
UNLK An	Unsized	1	An → SP; [SP]+ → An