Instruction **Instruction Format**

ABCD Add decimal with extend

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Rx			1	0	0	0	0	R/M	Ry		

ABCD Dy, Dx R/M = 0 data register to data register
ABCD −(Ay), −(Ax) R/M = 1 memory to memory

ADD Add binary

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
ADD <ea>, Dn	000	001	010
ADD Dn, <ea>	100	101	110

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ADDA Add address
ADDA <ea>, An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	An			Opmode			Effective Address					
										Mode			Register		

Opmode	word	long
	011	111

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

ADDI Add immediate

ADDI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ADDQ Add quick

ADDQ #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Data			0	Size		Effective Address					
										Mode			Register		

Data - 3 bits, {1-7, 0}, which represent the range {1-7, 8}

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ADDX Add extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Rx			1	Size		0	0	R/M	Ry		

Size	byte	word	long
	00	01	10

ADDX Dy, Dx R/M = 0 data register to data register

ADDX -(Ay), -(Ax) R/M = 1 memory to memory

AND AND logical

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
AND <ea>, Dn	000	001	010
AND Dn, <ea>	100	101	110

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ANDI AND immediate
ANDI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ANDI AND immediate to CCR
ANDI #<data>, CCR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	Byte Data (8 bits)							

ANDI AND immediate to SR
ANDI #<data>, SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0
Word Data (16 bits)															

ASL, ASR Arithmetic shift

ASd Dx, Dy

ASd #<data>, Dy

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/Dx			dr	Size		i/r	0	0	Dy		

dr = 0 shift left
dr = 1 shift right

i/r = 0 immediate shift count
i/r = 1 register shift count

Count If immediate, 3 bits, {1-7, 0}, which represent the range {1-7, 8}
Dx If register, shift count in register Dx

Size	byte	word	long
	00	01	10

ASd <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	0	0	dr	1	1	Effective Address					
										Mode			Register		

dr = 0 shift left
dr = 1 shift right

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

Bcc Branch conditionally

Bcc <label>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	Condition				8-bit displacement							
16-bit displacement if 8-bit displacement = 0															

<label> = PC + displacement where displacement is 2's complement

Condition:

CC	0100	MI	1011
CS	0101	GE	1100
NE	0110	LT	1101
EQ	0111	GT	1110
VC	1000	LE	1111
VS	1001	HI	0010
PL	1010	LS	0011

BCHG Test a bit and change

BCHG Dn, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Dn			1	0	1	Effective Address					
										Mode			Register		

BCHG #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	0	1	Effective Address					
										Mode			Register		
0	0	0	0	0	0	0	0	bit number							

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

BCLR Test a bit and clear

BCLR Dn, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Dn			1	1	0	Effective Address					
										Mode			Register		

BCLR #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	1	0	Effective Address					
										Mode			Register		
0	0	0	0	0	0	0	0	bit number							

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

BRA Branch always

BRA <label>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	0	8-bit displacement							
16-bit displacement if 8-bit displacement = 0															

<label> = PC + displacement where displacement is 2's complement

BSET Test a bit and set

BSET Dn, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Dn			1	1	1	Effective Address					
										Mode			Register		

BSET #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	1	1	Effective Address					
										Mode			Register		
0	0	0	0	0	0	0	0	bit number							

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

BSR

Branch to subroutine

BSR <label>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	1	8-bit displacement							
16-bit displacement if 8-bit displacement = 0															

<label> = PC + displacement where displacement is 2's complement

BTST

Test a bit

BTST Dn, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Dn			1	0	0	Effective Address					
										Mode			Register		

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

BTST #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	0	0	Effective Address					
										Mode			Register		
0	0	0	0	0	0	0	0	bit number							

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

CHK

Check register against bounds

CHK <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	Dn			1	1	0	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

CLR Clear an operand

CLR <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	1	0	Size		Effective Address					
										Mode			Register		

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

CMP Compare

CMP <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
	000	001	010

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

CMPA Compare address

CMP <ea>, An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	An			Opmode			Effective Address					
										Mode			Register		

Opmode	word	long
	011	111

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

CMPI Compare immediate

CMPI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

CMPM Compare memory

CMPM (Ay)+, (Ax)+

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Ax			1	Size		0	0	1	Ay		

Size	byte	word	long
	00	01	10

DBcc Test condition, decrement, branch

DBcc Dn, <label>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Condition				1	1	0	0	1	Dn		
displacement															

<label> = PC + displacement where displacement is 2's complement

Condition:

CC	0100	MI	1011
CS	0101	GE	1100
NE	0110	LT	1101
EQ	0111	GT	1110
VC	1000	LE	1111
VS	1001	HI	0010
PL	1010	LS	0011
F	0001	T	0000

DIVS Signed divide

DIVS <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Dn			1	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

DIVU Unsigned divide

DIVU <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Dn			0	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

EOR Exclusive OR logical

EOR Dn, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
	100	101	110

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

EORI Exclusive OR immediate

EORI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

EORI Exclusive OR immediate to CCR

EORI #<data>, CCR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	Byte Data (8 bits)							

EORI Exclusive OR immediate to SR

EORI #<data>, SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	0	1	1	1	1	1	0	0
Word Data (16 bits)															

EXG Exchange registers

EXG Rx, Ry

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Rx			1	Opmode					Ry		

Opmode - specifies whether exchanging

Opmode	Dn	An	Dn/An*
	01000	01001	10001

* if exchange is between Dn and An, then Rx=Dn and Ry=An

EXT Sign extend

EXT Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	Opmode			0	0	0	Dn		

Opmode	B -> W	W -> L
	010	011

ILLEGAL Illegal instruction

ILLEGAL

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	0

JMP Jump

JMP <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	--	--	d(PC)	111	010
−(An)	--	--	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

JSR Jump to subroutine

JSR <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	1	0	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	--	--	d(PC)	111	010
−(An)	--	--	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

LEA Load effective address

LEA <ea>, An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	An			1	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	--	--	d(PC)	111	010
−(An)	--	--	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

LINK Link and allocate

LINK An, #<displacement>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	1	0	An		
displacement (2's complement)															

LSL, LSR Logical shift

LSd Dx, Dy

LSd #<data>, Dy

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/Dx			dr	Size		i/r	0	1	Dy		

dr = 0 shift left
dr = 1 shift right

i/r = 0 immediate shift count
i/r = 1 register shift count

Count If immediate, 3 bits, {1-7, 0}, which represent the range {1-7, 8}
Dx If register, shift count in register Dx

Size	byte	word	long
	00	01	10

LSd <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	0	1	dr	1	1	Effective Address					
										Mode			Register		

dr = 0 shift left
dr = 1 shift right

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

MOVE Move data

MOVE <ea>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	Size		Destination						Source					
				Register			Mode			Mode			Register		

Size	byte	word	long
	01	11	10

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

MOVE Move to CCR

MOVE <ea>, CCR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	0	0	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

MOVE Move to SR

MOVE <ea>, SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	0	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

MOVE Move from SR

MOVE SR, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	1	1	Effective Address					
										Mode			Register		

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

MOVE MOVE USP

MOVE USP, An

MOVE An, USP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	0	dr	An		

dr = 0 transfer address register to USP

dr = 1 tramsfer USP to address register

MOVEA

Move address
MOVEA <ea>, An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	Size		Destination Register			0	0	1	Source Mode			Source Register		

Size	word	long
	11	10

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

MOVEM

Move multiple registers
MOVEM <reg list>, <ea>
MOVEM <ea>, <reg list>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	dr	0	0	1	Sz	Effective Address Mode			Effective Address Register		
Register List Mask															

dr = 0 register to memory
dr = 1 memory to register

Sz = 0 word transfer
Sz = 1 longword transfer

Source effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	--	--	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	--	--	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

Register List Mask
Addressing mode -(An):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D0	D1	D2	D3	D4	D5	D6	D7	A0	A1	A2	A3	A4	A5	A6	A7

All other addressing modes:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0

=> low order bit is the first register to be transferred

MOVEP Move peripheral data

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Dx			Opmode			0	0	1	Ay		
displacement															

Opmode - specifies direction and size of operation

Opmode	word	long
MOVEP d(Ay), Dx	100	101
MOVEP Dx, d(Ay)	110	111

MOVEQ Move quick
MOVEQ #<data>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	Dn			0	data							

MULS Signed multiply
MULS <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Dn			1	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

MULU Unsigned multiply
MULU <ea>, Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Dn			0	1	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

NBCD Negate decimal with extend
NBCD <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	0	Effective Address					
										Mode			Register		

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

NEG Negate

NEG <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	0	0	Size		Effective Address					
										Mode			Register		

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

NEGX Negate with extend

NEGX <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	Size		Effective Address					
										Mode			Register		

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

NOP No operation

NOP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	0	1

NOT Logical complement

NOT <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	0	Size		Effective Address					
										Mode			Register		

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

OR Inclusive OR logical

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
OR <ea>, Dn	000	001	010
OR Dn, <ea>	100	101	110

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ORI Inclusive OR immediate
ORI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

ORI OR immediate to CCR
ORI #<data>, CCR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	Byte Data (8 bits)							

ORI OR immediate to SR
ORI #<data>, SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0
Word Data (16 bits)															

PEA Push effective address

PEA <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	1	Effective Address					
										Mode			Register		

Source effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	--	--	d(PC)	111	010
−(An)	--	--	d(PC,Rn)	111	011
d(An)	101	#	#<data>	--	--

RESET Reset external devices

RESET

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	0	0

ROR, ROL Rotate

ROd Dx, Dy

ROd #<data>, Dy

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/Dx			dr	Size		i/r	1	1	Dy		

dr = 0 rotate left
dr = 1 rotate right

i/r = 0 immediate rotate count
i/r = 1 register rotate count

Count If immediate, 3 bits, {1-7, 0}, which represent the range {1-7, 8}
Dx If register,rotate count in register Dx

Size	byte	word	long
	00	01	10

ROd <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	1	1	dr	1	1	Effective Address					
										Mode			Register		

dr = 0 rotate left
dr = 1 rotate right

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

RORX, ROLX Rotate with extend

ROXd Dx, Dy
ROXd #<data>, Dy

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/Dx			dr	Size		i/r	1	0	Dy		

dr = 0 rotate left
dr = 1 rotate right

i/r = 0 immediate rotate count
i/r = 1 register rotate count

Count If immediate, 3 bits, {1-7, 0}, which represent the range {1-7, 8}
Dx If register,rotate count in register Dx

Size	byte	word	long
	00	01	10

ROXd <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	1	0	dr	1	1	Effective Address					
										Mode			Register		

dr = 0 rotate left
dr = 1 rotate right

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

RTE Return from exception

RTE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	1	1

RTR Return and restore CCR

RTR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	1	1

RTS Return from subroutine

RTS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	0	1

SBCD Subtract decimal with extend

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Rx			1	0	0	0	0	R/M	Ry		

SBCD Dy, Dx
SBCD −(Ay), −(Ax)

R/M = 0 data register to data register
R/M = 1 memory to memory

Scc Set according to condition

Scc <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Condition				1	1	Effective Address					
										Mode			Register		

Condition:

CC	0100	MI	1011
CS	0101	GE	1100
NE	0110	LT	1101
EQ	0111	GT	1110
VC	1000	LE	1111
VS	1001	HI	0010
PL	1010	LS	0011
F	0001	T	0000

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

STOP Load status register and stop

STOP #<data>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	1	0
Immediate data															

SUB Subtract binary

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	Dn			Opmode			Effective Address					
										Mode			Register		

Opmode	byte	word	long
SUB <ea>, Dn	000	001	010
SUB Dn, <ea>	100	101	110

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

Destination effective address:

	Mode	Register		Mode	Register
Dn	--	--	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

SUBA Subtract address

SUBA <ea>, An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	An			Opmode			Effective Address					
										Mode			Register		

Opmode	word	long
	011	111

Source effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	111	010
−(An)	100	#	d(PC,Rn)	111	011
d(An)	101	#	#<data>	111	100

SUBI Subtract immediate

SUBI #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	Size		Effective Address					
										Mode			Register		
Word Data (16 bits)								Byte Data (8 bits)							
Long Data (32 bits, including previous word)															

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

SUBQ Subtract quick

SUBQ #<data>, <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Data			1	Size		Effective Address					
										Mode			Register		

Data - 3 bits, {1-7, 0}, which represent the range {1-7, 8}

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	001	#	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

SUBX Subtract with extend

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	Rx			1	Size		0	0	R/M	Ry		

Size	byte	word	long
	00	01	10

SUBX Dy, Dx — R/M = 0 — data register to data register
SUBX -(Ay), -(Ax) — R/M = 1 — memory to memory

SWAP Swap register halves
SWAP Dn

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	1	0	0	0	Dn		

TAS Test and set an operand
TAS <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	1	0	1	1	Effective Address					
										Mode			Register		

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

TRAP Trap
TRAP #<vector #>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	0	Vector #			

TRAPV Trap on overflow
TRAPV

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	1	0

TST Test an operand
TST <ea>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	1	0	Size		Effective Address					
										Mode			Register		

Size	byte	word	long
	00	01	10

Destination effective address:

	Mode	Register		Mode	Register
Dn	000	#	d(An,Rn)	110	#
An	--	--	(xxx).W	111	000
(An)	010	#	(xxx).L	111	001
(An)+	011	#	d(PC)	--	--
−(An)	100	#	d(PC,Rn)	--	--
d(An)	101	#	#<data>	--	--

UNLK Unlink and deallocate
UNLK An

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	1	1	An		