PROGRAMMER'S REFERENCE MANUAL

(Includes CPU32 Instructions)



Instruction Set Summary

Table 3-1. Notational Conventions

	Single And Dauble Onemand Onemations
+	Arithmetic addition or nostincrement indicator
-	Arithmetic authraction or productional indicator
1	Allument submaction of predecientern markardi.
×	Arithmetic multiplication.
+	Arithmetic division or conjunction symbol.
ł	Invert; operand is logically complemented.
V	Logical AND
Λ	Logical OR
0	Logical exclusive OR
1	Source operand is moved to destination operand.
1	Two operands are exchanged.
<do></do>	Any double-operand operation.
<observand>tested</observand>	Operand is compared to zero and the condition codes are set appropriately.
sign-extended	All bits of the upper portion are made equal to the high-order bit of the lower portion.
	Other Operations
TRAP	Equivalent to Format +Offset Word \rightarrow (SSP); SSP $-2 \rightarrow$ SSP; PC \rightarrow (SSP); SSP $-4 \rightarrow$ SSP; SR \rightarrow (SSP); SSP $-2 \rightarrow$ SSP; (Vector) \rightarrow PC
STOP	Enter the stopped state, waiting for interrupts.
<oberand>10</oberand>	The operand is BCD; operations are performed in decimal.
f <condition> :hen <operations> else <operations></operations></operations></condition>	Test the condition. If true, the operations after "then" are performed. If the condition is false and the optional "else" clause is present, the operations after "else" are performed. If the condition is false and else is omitted, the instruction performs no operation. Refer to the Bcc instruction description as an example.
	Register Specifications
An	Any Address Register n (example: A3 is address register 3)
Ax, Ay	Source and destination address registers, respectively.
õ	Data register D7–D0, used during compare.
Dh, DI	Data register's high- or low-order 32 bits of product.
υ	Any Data Register n (example: D5 is data register 5)
Dr, Dd	Data register's remainder or quotient of divide.
Da	Data register D7-D0, used during update.
Dx, Dy	Source and destination data registers, respectively.
MRn	Any Memory Register n.
윤	Any Address or Data Register
Rx, Ry	Any source and destination registers, respectively.
x	Index Register

Instruction Set Summary

Š 5

	Table 3-1. Notational Conventions (Continued)
	Data Format And Type
+ inf	Positive Infinity
<fmt></fmt>	Operand Data Format: Byte (B), Word (W), Long (L), Single (S), Double (D), Extended (X), or Packed (P).
B, W, L	Specifies a signed integer data type (twos complement) of byte, word, or long word.
۵	Double-precision real data format (64 bits).
×	A twos complement signed integer (–64 to +17) specifying a number's format to be stored in tr packed decimal format.
۵	Packed BCD real data format (96 bits, 12 bytes).
S	Single-precision real data format (32 bits).
×	Extended-precision real data format (96 bits, 16 bits unused).
luf –	Negative Infinity
	Subfields and Qualifiers
# <xxx> or #<data></data></xxx>	Immediate data following the instruction word(s).
0	Identifies an indirect address in a register.
Ξ	Identifies an indirect address in memory.
pq	Base Displacement
200	Index into the MC68881/MC68882 Constant ROM
['] p	Displacement Value, n Bits Wide (example: d ₁₆ is a 16-bit displacement).
LSB	Least Significant Bit
NST	Least Significant Word
MSB	Most Significant Bit
MSW	Most Significant Word
ро	Outer Displacement
SCALE	A scale factor (1, 2, 4, or 8 for no-word, word, long-word, or quad-word scaling, respectively).
SIZE	The index register's size (W for word, L for long word).
{offset:width}	Bit field selection.
	Register Names
CCR	Condition Code Register (lower byte of status register)
DFC	Destination Function Code Register
FPcr	Any Floating-Point System Control Register (FPCR, FPSR, or FPIAR)
FPm, FPn	Any Floating-Point Data Register specified as the source or destination, respectively.
IC, DC, IC/DC	Instruction, Data, or Both Caches
MMUSR	MMU Status Register
PC	Program Counter
Rc	Any Non Floating-Point Control Register
SFC	Source Function Code Register
SR	Status Register

MOTOROLA

3-2

Instruction Set Summary

Table 3-1. Notational Conventions (Concluded)

	Register Codes
*	General Case
O	Carry Bit in CCR
20	Condition Codes from CCR
FC	Function Code
z	Negative Bit in CCR
D	Undefined, Reserved for Motorola Use.
>	Overflow Bit in CCR
×	Extend Bit in CCR
Z	Zero Bit in CCR
	Not Affected or Applicable.
	Stack Pointers
ISP	Supervisor/Interrupt Stack Pointer
MSP	Supervisor/Master Stack Pointer
SP	Active Stack Pointer
SSP	Supervisor (Master or Interrupt) Stack Pointer
USP	User Stack Pointer
	Miscellaneous
<ea>></ea>	Effective Address
<label></label>	Assemble Program Label
	List of registers, for example D3–D0.
FB	Lower Bound
٤	Bit m of an Operand
F	Bits m through n of Operand
an	Upper Bound

Instruction Set Summary

Table 3-18. Integer Unit Condition Code Computations

Operations	×	z	Z	>	ပ	Special Definition
ABCD	*	Э	٠	D	<i>~</i> .	C = Decimal Carry Z = Z \(\text{A} \text{Fm} \(\text{A} \cdots \cdot \text{R0} \)
АБВ, АБВІ, АБВQ	*	*	*	٠	~-	$V = Sm \Lambda Dm \Lambda Rm V Sm \Lambda Dm \Lambda Rm$ $C = Sm \Lambda Dm V Rm \Lambda Dm V Sm \Lambda Rm$
ADDX	*	*	٠.	٠.	~	V = Sm A Dm A Rm V Sm A Dm A Rm C = Sm A Dm V Rm A Dm V Sm A Rm Z = Z A Rm A A R0
AND, ANDI, EOR, EORI, MOVEQ, MOVE, OR, ORI, CLR, EXT, EXTB, NOT, TAS, TST	1	*	*	0	0	
CHK	ı	*	n	n	⊃	
CHK2, CMP2	1	Э	<i>د</i>	Э	<i>~</i>	Z = (R = LB) V (R = UB) C = (LB ≤ UB) Λ (R < LB) V (R > UB)) V (UB < LB) Λ (R > UB) Λ (R < LB)
SUB, SUB, SUBQ	*	*	*	خ	<i>~</i> .	$V = \overline{Sm} \Lambda Dm \Lambda \overline{Rm} V Sm \Lambda \overline{Dm} \Lambda Rm$ $C = Sm \Lambda \overline{Dm} V Rm \Lambda \overline{Dm} V Sm \Lambda Rm$
SUBX	*	*	خ	خ	~	V = Sm A Dm A Rm V Sm A Dm A Rm C = Sm A Dm V Rm A Dm V Sm A Rm Z = Z A Rm AA R0
CAS, CAS2, CMP, CMPA, CMPI, CMPM	Ι	*	*	?	<i>~</i> .	$V = \overline{Sm} \Lambda Dm \Lambda \overline{Rm} V Sm \Lambda \overline{Dm} \Lambda Rm$ $C = Sm \Lambda \overline{Dm} V Rm \Lambda \overline{Dm} V Sm \Lambda Rm$
DIVS, DUVU	ı	*	*	5	0	V = Division Overflow
MULS, MULU	Ι	*	*	?	0	V = Multiplication Overflow
SBCD, NBCD	*	U	خ	U	خ	C = Decimal Borrow $Z = Z \Lambda \overline{Rm} \Lambda \Lambda \overline{R0}$
NEG	*	*	*	?	٠.	V = Dm A Rm C = Dm V Rm
NEGX	*	*	خ	?	<i>د</i>	V = Dm
BTST, BCHG, BSET, BCLR	I	I	خ	I	1	Z = <u>Dn</u>
BFTST, BFCHG, BFSET, BFCLR	ı	٠	ć.	0	0	N = Dm Z = Dn A Dm-7 A A D0
BFEXTS, BFEXTU, BFFFO	I	ن	خ	0	0	N = Sm Z = Sm A <u>Sm-1</u> AA <u>S0</u>
BFINS	ı	5	خ	0	0	N = Dm Z = Dm A Dm-1 AA D0
ASL	*	*	*	<i>د</i> -	~	V = Dm A Dm=7 VV Dm=r V Dm A (DM -1 V+ Dm - r) C = Dm=r+1
ASL (r = 0)	ı	*	*	0	0	
LSL, ROXL	*	*	*	0	<i>ر</i> .	C = Dm - r + 1

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

Instruction Set Summary

Table 3-18. Integer Unit Condition Code Computations (Continued)

Operations	×	z	Z	>	ပ	Special Definition
LSR (r = 0)	I	*	*	0	0	
ROXL (r = 0)	I	*	*	0	خ	X = C
ROL	Ι	*	*	0	خ	C = Dm - r + 1
ROL (r = 0)	Ι	*	*	0	0	
ASR, LSR, ROXR	*	*	*	0	خ	C = Dr – 1
ASR, LSR (r = 0)	Ι	*	*	0	0	
ROXR (r = 0)	I	*	*	0	۲.	X = C
ROR	I	*	*	0	۲.	C = Dr - 1
ROR (r = 0)	1	*	*	0	0	

? = Other—See Special Definition

N = Result Operand (MSB)

 \overline{Rm} = Not Result Operand (MSB) Rm = Result Operand (MSB)

R = Register Tested r = Shift Count

 $Z = \overline{Rm} \Lambda...\Lambda \overline{R0}$

Dm = Destination Operand (MSB) Sm = Source Operand (MSB)

Table 3-19. Conditional Tests

															_	_
Test	_	0	ZV2	CVZ	O	O	Z	Z	>	>	z	z	NAVVNAV	NAVVNAV	NAVAZVNAVAZ	ZVNAVVNAV
Encoding	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Condition	True	False	High	Low or Same	Carry Clear	Carry Set	Not Equal	Equal	Overflow Clear	Overflow Set	Plus	Minus	Greater or Equal	Less Than	Greater Than	Less or Equal
Mnemonic	*_	*	Ī	ST	CC(HI)	CS(LO)	ЯN	EQ	۸C	ΝS	П	M	GE	H	GT	Щ

$$\begin{split} \overline{N} &= \text{Logical Not N} \\ \overline{V} &= \text{Logical Not V} \\ \overline{Z} &= \text{Logical Not Z} \\ *\text{Not available for the Bcc instruction.} \end{split}$$

Instruction Set Summary

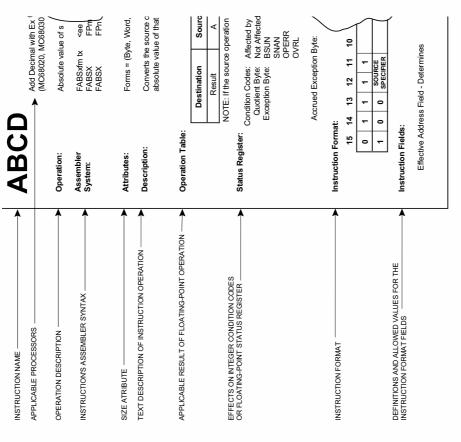


Figure 3-3. Instruction Description Format

Add Decimal with Extend (M68000 Family) ABCD

ABCD

Source10 + Destination10 + X → Destination Operation:

ABCD Dy,Dx Assembler

ABCD - (Ay), - (Ax) Syntax:

Size = (Byte) Attributes:

Description: Adds the source operand to the destination operand along with the extend bit, and stores the result in the destination location. The addition is performed using binarycoded decimal arithmetic. The operands, which are packed binary-coded decimal numbers, can be addressed in two different ways:

- 1. Data Register to Data Register. The operands are contained in the data registers specified in the instruction.
- 2. Memory to Memory: The operands are addressed with the predecrement addressing mode using the address registers specified in the instruction

This operation is a byte operation only.

Condition Codes

ပ	*	
>	n	
Z	*	
z	n	
×	*	

- X Set the same as the carry bit.
 N Undefined.
 Z Cleared if the result is nonzero; unchanged otherwise.
 V Undefined.
 C Set if a decimal carry was generated; cleared otherwise.

Normally, the Z condition code bit is set via programming before the start of an operation. This allows successful tests for zero results upon completion of multiple-precision operations.

Integer Instructions

ABCD

ABCD

Add Decimal with Extend (M68000 Family)

Instruction Fields:

Register Rx field—Specifies the destination register

If R/M = 0, specifies a data register.

If R/M = 1, specifies an address register for the predecrement addressing mode.

R/M field—Specifies the operand addressing mode.

0 — The operation is data register to data register.1 — The operation is memory to memory.

Register Ry field—Specifies the source register.

If RM=0, specifies a data register. If RM=1, specifies an address register for the predecrement addressing mode.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

ADD

Add (M68000 Family)

ADD

Source + Destination → Destination Operation:

ADD < ea > ,Dn Assembler

ADD Dn, < ea > Syntax: Size = (Byte, Word, Long) Attributes:

stores the result in the destination location. The size of the operation may be specified as byte, word, or long. The mode of the instruction indicates which operand is the source and which is the destination, as well as the operand size. Description: Adds the source operand to the destination operand using binary addition and

Condition Codes:

ပ	*
>	*
Z	*
z	*
×	*

X — Set the same as the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow is generated; cleared otherwise.
C — Set if a carry is generated; cleared otherwise.

Instruction Format:

0		ĸ.
-	ESS	REGISTE
7	'E ADDR	_
က	FFECTIV	
4	Ī	MODE
2		
9	با	ų
7		
∞		
6	6	<u>د</u>
10	GTEGICIE	
Ξ		
12	,	-
13		>
4	,	-
15	,	-

ADD

Add (M68000 Family)

ADD

Integer Instructions

Instruction Fields:

Register field—Specifies any of the eight data registers.

Opmode field

Operation	< ea > + Dn → Dn	Dn + < ea > → < ea >
Long	010	110
Word	001	101
Byte	000	100

Effective Address field—Determines addressing mode.

a. If the location specified is a source operand, all addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regist
D	000	reg. number:Dn	(xxx).W	111	000
An*	001	reg. number:An	J.(xxx)	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

011	011	011	
111	111	111	
(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
_	_	_	
reg. number:Ar	reg. number:Ar	reg. number:An	
110	110	110	
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)	

4-4

^{*}Word and long only
**Can be used with CPU32.

ADD

ADD

b. If the location specified is a destination operand, only memory alterable (M68000 Family)

addressing modes can be used as listed in the following tables:

Registe	000	100	I			I	I
Mode	11	111	I			ı	_
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	ı	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	ı	ı	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

ı	ı	1
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

I

*Can be used with CPU32

NOTE

The Dn mode is used when the destination is a data register; the destination < ea > mode is invalid for a data register ADDA is used when the destination is an address register. ADDI and ADDQ are used when the source is immediate data. Most assemblers automatically make this distinction.

Integer Instructions

ADDA

ADDA

Add Address (M68000 Family)

Source + Destination → Destination

Operation:

ADDA < ea > , An Assembler Syntax:

Size = (Word, Long) Attributes:

Description: Adds the source operand to the destination address register and stores the result in the address register. The size of the operation may be specified as word or long. The entire destination address register is used regardless of the operation size.

Condition Codes:

Not affected.

Instruction Format:

0		ER
-	ESS	REGISTE
7	E ADDR	_
က	FFECTIVI	
4	Ш	MODE
2		
9		ш
7	OPMODE	
80	Ĺ	
6	EGISTER	
10		
£		2
12	-	
13	-	>
14	,	-
15	,	-

Instruction Fields:

Register field—Specifies any of the eight address registers. This is always the destination.

Opmode field—Specifies the size of the operation.

011—Word operation; the source operand is sign-extended to a long operand and the operation is performed on the address register using all 32 bits.

111—Long operation.

4-6

ADDA

Add Address (M68000 Family)

ADDA

Effective Address field—Specifies the source operand. All addressing modes can be used as listed in the following tables:

Register	000	001	100			010	011
Mode	111	111	111			111	111
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Reç	reg. nu	reg. n	reg. n	reg. n	reg. n	reg. n	reg. n
Addressing Mode Mode Reç	000 reg. nu	001 reg. nu	010 reg. n	011 reg. n	100 reg. n	101 reg. n	110 reg. n

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

011 011 011

7 11 111

*Can be used with CPU32

ADDI

Add Immediate (M68000 Family)

ADDI

Integer Instructions

Immediate Data + Destination → Destination Operation:

Assembler Syntax:

ADDI # < data > , < ea >

Size = (Byte, Word, Long) Attributes:

Description: Adds the immediate data to the destination operand and stores the result in the destination location. The size of the operation may be specified as byte, word, or long. The size of the immediate data matches the operation size.

Condition Codes:

ပ	*
>	*
Z	*
z	*
×	*

X — Set the same as the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow is generated; cleared otherwise.
C — Set if a carry is generated; cleared otherwise.

Instruction Format:

0				
-	SS	REGISTER		
7	ADDRESS	<u>~</u>	TE DATA	
ဗ	EFFECTIVE		8-BIT BY	
4	EFF	MODE	8	
2				
9	22.13	Ц		4
7	٥	ō ๋		NG DAT
œ	c	>		2-BIT LO
6	,	-	4	33
9	,	-	6-BIT WORD DAT	
=	c	>	-BIT WC	
12	-	>	16	
13		>		
4		>		
15		>		
	_		_	_

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

ADDI

Add Immediate (M68000 Family)

ADDI

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	1			I	1
Mode	11	111	ı			1	1
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Register	000 reg. number:Dn	ı	010 reg. number:An	011 reg. number:An	100 reg. number:An	101 reg. number: An	110 reg. number:An

MC68020, MC68030, and MC68040 only

1		1
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

I

Immediate field—Data immediately following the instruction. If size = 00, the data is the low-order byte of the immediate word. If size = 01, the data is the entire immediate word.

If size = 10, the data is the next two immediate words.

ADDQ

ADDQ

Integer Instructions

Immediate Data + Destination → Destination Add Quick (M68000 Family) Operation:

ADDQ # < data > , < ea > Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

location. The size of the operation may be specified as byte, word, or long. Word and long operations are also allowed on the address registers. When adding to address registers, the condition codes are not altered, and the entire destination address Description: Adds an immediate value of one to eight to the operand at the destination register is used regardless of the operation size.

Condition Codes:

ပ	
>	
Z	*
z	*
×	*

X — Set the same as the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.
C — Set if a carry occurs; cleared otherwise.

The condition codes are not affected when the destination is an address register.

Instruction Format:

0		띪
-	ESS	REGISTE
2	E ADDR	_
က	FECTIVI	
4	Ш	MODE
2		
9	L.	171
7	٥	0
80	0	
6		
9	F	<u> </u>
£		
12	,	-
13		>
4	,	-
12		>

4-10

^{*}Can be used with CPU32

ADDQ

Add Quick (M68000 Family)

ADDQ

Instruction Fields:

Data field—Three bits of immediate data representing eight values (0-7), with the immediate value zero representing a value of eight.

Size field—Specifies the size of the operation

00—Byte operation

01—Word operation 10—Long operation Effective Address field—Specifies the destination location. Only alterable addressing modes can be used as listed in the following tables:

Register 000 00 I

lode	111	111				-	1
Addressing Mode Mode	W.(xxx)	T(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	001	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

I	I	١
(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn**	([bd,An,Xn],od)	([bd,An],Xn,od)

1 1 I

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-12

Integer Instructions

ADDX

Add Extended (M68000 Family) **ADDX**

Source + Destination + X → Destination Operation:

Assembler

ADDX – (Ay), – (Ax) Syntax: Size = (Byte, Word, Long)

Attributes:

stores the result in the destination location. The operands can be addressed in two Description: Adds the source operand and the extend bit to the destination operand and

1. Data register to data register—The data registers specified in the instruction contain the operands.

different ways:

2. Memory to memory—The address registers specified in the instruction address the operands using the predecrement addressing mode.

The size of the operation can be specified as byte, word, or long.

Condition Codes:

ပ	*
>	*
Z	*
z	*
×	*

X — Set the same as the carry bit.

N — Set if the result is negative; cleared otherwise.
 Z — Cleared if the result is nonzero; unchanged otherwise.
 V — Set if an overflow occurs; cleared otherwise.
 C — Set if a carry is generated; cleared otherwise.

Normally, the Z condition code bit is set via programming before the start of an operation. This allows successful tests for zero results upon completion of multiple-precision operations.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Word and long only.
**Can be used with CPU32.

ADDX

Add Extended (M68000 Family)

ADDX

Instruction Format:

Instruction Fields:

Register Rx field—Specifies the destination register.

If R/M = 0, specifies a data register. If R/M = 1, specifies an address register for the predecrement addressing mode.

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

R/M field—Specifies the operand address mode.

0 — The operation is data register to data register.

1 — The operation is memory to memory.

Register Ry field—Specifies the source register.

If R/M = 0, specifies a data register. If R/M = 1, specifies an address register for the predecrement addressing mode.

Integer Instructions

AND

AND

AND Logical (M68000 Family)

Source L Destination → Destination Operation:

AND Dn, < ea > AND < ea > ,Dn Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

be specified as byte, word, or long. The contents of an address register may not be **Description:** Performs an AND operation of the source operand with the destination operand and stores the result in the destination location. The size of the operation can used as an operand.

Condition Codes:

00 0 > Ν z ×

X — Not affected.

N — Set if the most significant bit of the result is set; cleared otherwise. Z — Set if the result is zero; cleared otherwise. V — Always cleared.

C — Always cleared.

Instruction Format:

EFFECTIVE ADDRESS

2

MODE

OPMODE

REGISTER

0

0

Instruction Fields:

Register field—Specifies any of the eight data registers.

Opmode field

Dn Λ < ea > \rightarrow < ea > < ea > Λ Dn → Dn Operation Long 010 Word 6 1 **Byte** 000 100

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

AND

AND Logical (M68000 Family)

AND

Effective Address field—Determines addressing mode.

If the location specified is a source operand, only data addressing modes can be used as listed in the following tables:

Register 000 001

Mode	111	111	111			111	111
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	ı	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	ď	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

100

010 011

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn*	(lbd,An,Xn],od)	([bd,An],Xn,od)	

011 011

1 111 111

AND

AND Logical (M68000 Family)

AND

Integer Instructions

b. If the location specified is a destination operand, only memory alterable address-ing modes can be used as listed in the following tables:

Register 000 9

de Mode	111	111	I			I	I
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d _x .PC.Xn)
Register	I	ı	reg. number:An	reg. number:An	reg. number:An	reg. number:An	rea. number:An
Mode	ı	I	010	011	100	101	110
Addressing Mode Mode	Б	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(ds.An.Xn)

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

([bd,PC],Xn,od) ([bd,PC,Xn],od) (bd,PC,Xn)*

*Can be used with CPU32.

The Dn mode is used when the destination is a data register; the destination < ea > mode is invalid for a data register.

Most assemblers use ANDI when the source is immediate data.

MOTOROLA

^{*}Can be used with CPU32.

ANDI

AND Immediate (M68000 Family)

ANDI

Immediate Data ∧ Destination → Destination Operation:

Assembler Syntax:

ANDI # < data > , < ea >

Size = (Byte, Word, Long) Attributes:

Description: Performs an AND operation of the immediate data with the destination operand and stores the result in the destination location. The size of the operation can be specified as byte, word, or long. The size of the immediate data matches the operation size.

Condition Codes:

ပ	0
>	0
Z	*
z	
×	

X — Not affected.
N — Set if the most significant bit of the result is set; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

0		~		
-	SS	REGISTER		
2	ADDRESS	<u>~</u>	3-BIT BYTE DATA	
ო	EFFECTIVE,		-BIT BY	
4	监	MODE	8	
2				
9	1710	Ц		A
7		ō		NG DAT
œ	c	>		2-BIT LC
6	,	-	Ā	33
10		>	RD DAT	
Ε	c	>	6-BIT WORD DATA	
12	c	>	16	
13	c	>		
4	c	>		
15	٥	>		

AND Immediate (M68000 Family) ANDI

Integer Instructions

ANDI

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation

10 — Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register 00 001

Mode	111	111	I			1	ı
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_	_	_					
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	1	010	011	100	101	110
Addressing Mode Mode	ď	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

I

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([pd,An,Xn],od)	([bd,An],Xn,od)

1 1

*Can be used with CPU32

Immediate field—Data immediately following the instruction.

If size = 00, the data is the low-order byte of the immediate word. If size = 01, the data is the entire immediate word. If size = 10, the data is the next two immediate words.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

ANDI to CCR

CCR AND Immediate (M68000 Family)

Source ∆ CCR → CCR Operation:

ANDI # < data > ,CCR Assembler Syntax: Size = (Byte) Attributes: Description: Performs an AND operation of the immediate operand with the condition codes and stores the result in the low-order byte of the status register.

Condition Codes:

>

X — Cleared if bit 4 of immediate operand is zero; unchanged otherwise.
N — Cleared if bit 3 of immediate operand is zero; unchanged otherwise.
Z — Cleared if bit 2 of immediate operand is zero; unchanged otherwise.
V — Cleared if bit 1 of immediate operand is zero; unchanged otherwise.
C — Cleared if bit 0 of immediate operand is zero; unchanged otherwise.

Instruction Format:

0	0	
-	0	
7	-	1
က	1	TE DAT
4	-	з-вп ву
2	-	3
9	0	
_	0	
œ	0	0
တ	1	0
10	0	0
7	0	0
12	0	0
13	0	0
4	0	0
15	0	0

Supervisor (Privileged) Instructions

ANDI to SR

AND Immediate to the Status Register (M68000 Family)

ANDI to SR

If Supervisor State Then Source L SR \rightarrow SR

Operation:

ELSE TRAP

ANDI # < data > ,SR Assembler Syntax:

size = (word) Attributes: **Description:** Performs an AND operation of the immediate operand with the contents of the status register and stores the result in the status register. All implemented bits of the status register are affected.

Condition Codes:



X—Cleared if bit 4 of immediate operand is zero; unchanged otherwise.

N—Cleared if bit 3 of immediate operand is zero; unchanged otherwise.

Z—Cleared if bit 2 of immediate operand is zero; unchanged otherwise.

V—Cleared if bit 1 of immediate operand is zero; unchanged otherwise. C—Cleared if bit 0 of immediate operand is zero; unchanged otherwise.

Instruction Format:

0	0	
_	0	
5	-	
က	-	
4	-	
2	-	
9	-	≰
7	0	JRD DAT
80	0	BIT WC
6	-	16
10	0	
7	0	
12	0	
13	0	
4	0	
2	0	

4-20

ASL, ASR

Arithmetic Shift (M68000 Family)

ASL, ASR

Destination Shifted By Count → Destination Operation:

ASd # < data > ,Dy Assembler Syntax:

where d is direction, L or R ASd < ea >

Size = (Byte, Word, Long) Attributes:

The carry bit receives the last bit shifted out of the operand. The shift count for the Description: Arithmetically shifts the bits of the operand in the direction (L or R) specified shifting of a register may be specified in two different ways:

- 1. Immediate—The shift count is specified in the instruction (shift range, 1 8).
- 2. Register—The shift count is the value in the data register specified in instruction

The size of the operation can be specified as byte, word, or long. An operand in memory can be shifted one bit only, and the operand size is restricted to a word.

Bits shifted out of the high-order bit go to both the carry and the extend bits; zeros are shifted into the low-order bit. The overflow bit indicates if any sign changes occur dur-For ASL, the operand is shifted left; the number of positions shifted is the shift count. ing the shift

OPERAND ASL:

Integer Instructions

ASL, ASR

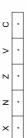
Arithmetic Shift (M68000 Family)

ASL, ASR

For ASR, the operand is shifted right; the number of positions shifted is the shift count. Bits shifted out of the low-order bit go to both the carry and the extend bits; the sign bit (MSB) is shifted into the high-order bit.



Condition Codes:



- X Set according to the last bit shifted out of the operand; unaffected for a shift count of zero.
- N Set if the most significant bit is changed at any time during the shift operation; V Set if the most significant bit is changed at any time during the shift operation;
 - Set according to the last bit shifted out of the operand; cleared for a shift count 0

Instruction Format

REGISTER SHIFTS

2

9

9 7

12

13

4 15

-	-	-	0	COUNT? REGISTER	ъ	SIZE	ı/r	0	0	REGISTER	$\overline{}$
Instruction Fields:	ıction	Field	:: S								
J	Sount/	Regis	ster fie	Count/Register field—Specifies shift count or register that contains the shift count: If $i/r = 0$ this field contains the shift count. The values $1 - 7$ represent counts of $1 - 1$.	ft cou	nt or registe	er that	conta	ins th	e shift count:	

U, this field contains the shift count. The valu7; a value of zero represents a count of eight

If i/r = 1, this field specifies the data register that contains the shift count (modulo 64)

MOTOROLA

ASL, ASR

Arithmetic Shift (M68000 Family)

ASL, ASR

dr field—Specifies the direction of the shift.

0 — Shift right

1 — Shift left

Size field—Specifies the size of the operation. 00 — Byte operation 01 — Word operation 10 — Long operation

If i/r = 0, specifies immediate shift count. If i/r = 1, specifies register shift count.

Register field—Specifies a data register to be shifted.

Instruction Format:

MEMORY SHIFTS

0		٣
-	ESS	REGISTER
7	E ADDR	_
က	FECTIVE	
4	H	MODE
2		
9	,	-
7	,	-
80	- 17	5
6		>
10		0
7		0
12		>
13	,	_
4	,	_
15	,	_

Instruction Fields:

dr field—Specifies the direction of the shift.

0 — Shift right

1 — Shift left

Integer Instructions

ASL, ASR

Arithmetic Shift (M68000 Family)

ASL, ASR

Effective Address field—Specifies the operand to be shifted. Only memory alterable addressing modes can be used as listed in the following tables:

Register 000 00

y Mode Mode # 7

Addressing Moo	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	ı	ı	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

1

I

1 |

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

*Can be used with CPU32.

I	_	I	
I	1	I	
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	

MOTOROLA

4-23

Bcc

Branch Conditionally (M68000 Family)

Bcc

If Condition True Operation:

Then PC + d_n → PC

Assembler Syntax:

Bcc < label >

Size = (Byte, Word, Long*) Attributes: *(MC68020, MC68030, and MC68040 only)

displacement (long word immediately following the instruction) is used. Condition code oc specifies one of the following conditional tests (refer to Table 3-19 for more information on these conditional tests): + displacement. The program counter contains the address of the instruction word for the Bcc instruction plus two. The displacement is a twos-complement integer that represents the relative distance in bytes from the current program counter to the destination program counter. If the 8-bit displacement field in the instruction word is zero, a 16-bit displacement (the word immediately following the instruction) is used. If the 8-bit displacement field in the instruction word is all ones (\$FF), the 32-bit **Description:** If the specified condition is true, program execution continues at location (PC)

Mnemonic	Condition	Mnemonic	Condition
CC(HI)	Carry Clear	ST	Low or Sam
CS(FO)	Carry Set	LT	Less Than
EQ	Equal	M	Minus
GE	Greater or Equal	Ä	Not Equal
GT	Greater Than	PL	SnIA
Ξ	High	ΛC	Overflow Cle
믜	Less or Equal	NS	Overflow Se

ar

Condition Codes:

Not affected.

Integer Instructions

Bcc

Branch Conditionally (M68000 Family)

Bcc

Instruction Format

8-BIT DISPLACEMENT 16-BIT DISPLACEMENT IF 8-BIT DISPLACEMENT = \$00 32-BIT DISPLACEMENT IF 8-BIT DISPLACEMENT = \$FF 2 7 6 CONDITION 10 0 12 3 4 15

nstruction Fields:

Condition field—The binary code for one of the conditions listed in the table.

8-Bit Displacement field—Twos complement integer specifying the number of bytes between the branch instruction and the next instruction to be executed if the condition is met. 16-Bit Displacement field—Used for the displacement when the 8-bit displacement field contains \$00. 32-Bit Displacement field—Used for the displacement when the 8-bit displacement field contains \$FF.

NOTE

uses the 16-bit displacement format because the 8-bit displacement field contains \$00 (zero offset). A branch to the immediately following instruction automatically

MOTOROLA

4-25

BCHG

BCHG

Operation:

Test a Bit and Change (M68000 Family)

TEST (< number > of Destination) \to Z; TEST (< number > of Destination) \to < bit number > of Destination

Assembler

BCHG Dn, < ea > BCHG # < data > , < ea > Syntax:

Size = (Byte, Long) Attributes:

the destination is a memory location, the operation is a byte operation, and the bit number is modulo 8. In all cases, bit zero refers to the least significant bit. The bit appropriately, then inverts the specified bit in the destination. When the destination is a data register, any of the 32 bits can be specified by the modulo 32-bit number. When Description: Tests a bit in the destination operand and sets the Z condition code

- number for this operation may be specified in either of two ways:

 1. Immediate—The bit number is specified in a second word of the instruction.
- 2. Register—The specified data register contains the bit number.

Condition Codes:

ပ	I
>	I
Z	*
z	I
×	I

X — Not affected.
N — Not affected.
Z — Set if the bit tested is zero; cleared otherwise.
V — Not affected.
C — Not affected.

Integer Instructions

BCHG

Test a Bit and Change (M68000 Family)

BCHG

Instruction Format:

BIT NUMBER DYNAMIC, SPECIFIED IN A REGISTER

0		ĸ
-	SS	EGISTE
2	ADDRE	<u>~</u>
8	FECTIVE	
4	EF	MODE
2		
9	,	-
7	c	>
80	,	-
6		Ľ
10	TECICIE	
7		Ľ.
12	O	>
13	o	>
4	c	>
15	c	>

Instruction Fields:

Register field—Specifies the data register that contains the bit number.

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

jister

Addressing Mode Mode	Mode	Podietor	Addressing Mode Mode	Mode	Dog
anom Guissaina		isosisisi i	anous Suiceaina	000	604
Dn*	000	reg. number:Dn	W.(xxx)	111	0
An	I	I	(xxx).L	111	ō
(An)	010	reg. number:An	# <data></data>	I	'
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	I	!
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	I	

MC68020, MC68030, and MC68040 only

(bd,An,Xn)**	110	reg. number:An		
(bo,[nX,nA,bd])	110	reg. number:An	-	
([bd,An],Xn,od)	110	reg. number:An		
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	4 4	1		

1	I	I	
I	I	I	
(bd,PC,Xn)	([bd,PC,Xn],od)	(lpd,PC],Xn,od)	

MOTOROLA

4-27

^{*}Long only; all others are byte only. **Can be used with CPU32.

BCHG

Test a Bit and Change (M68000 Family)

BCHG

Instruction Format:

BIT NUMBER STATIC, SPECIFIED AS IMMEDIATE DATA

0		œ	
-	SS	EGISTE	
2	ADDRE	æ	
ъ	EFFECTIVE		MBER
4	H	MODE	BIT NU
2			
9	,		
7			
80	0		0
6	0		0
10	0		0
£	-		0
12	0		0
13	0		0
14	c	>	0
15		>	0

Instruction Fields:

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	I
Mode	11	11	ı			I	I
Addressing Mode Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	Dn*	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

([bd,PC],Xn,od)	reg. number:An	110	([bd,An],Xn,od)	
([bd,PC,Xn],od)	reg. number:An	110	([bd,An,Xn],od)	
(bd,PC,Xn)	reg. number:An	110	(bd,An,Xn)**	

([bd,An,Xn],od) ([bd,An],Xn,od)

I 1

(bd,An,Xn)**

1

Bit Number field—Specifies the bit number.

Integer Instructions

BCLR

BCLR

TEST (< bit number > of Destination) \rightarrow Z; $0 \rightarrow$ < bit number > of Des-Test a Bit and Clear (M68000 Family) Operation:

BCLR Dn, < ea > BCLR # < data > , < ea > Assembler Syntax:

tination

Size = (Byte, Long) Attributes:

the destination, any of the 32 bits can be specified by a modulo 32-bit number. When a memory location is the destination, the operation is a byte operation, and the bit number is modulo 8. In all cases, bit zero refers to the least significant bit. The bit appropriately, then clears the specified bit in the destination. When a data register is Description: Tests a bit in the destination operand and sets the Z condition code

number for this operation can be specified in either of two ways:

1. Immediate—The bit number is specified in a second word of the instruction.

2. Register—The specified data register contains the bit number.

Condition Codes:

O	Ι
>	Ι
Z	*
z	Ι
×	ı

X — Not affected.
N — Not affected.
Z — Set if the bit tested is zero; cleared otherwise.
V — Not affected.
C — Not affected.

4-29

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

^{*}Long only; all others are byte only. **Can be used with CPU32.

Test a Bit and Clear (M68000 Family)

BCLR

Instruction Format:

BIT NUMBER DYNAMIC, SPECIFIED IN A REGISTER

0		œ	
-	SSES	EGISTER	
7	: ADDRE	<u>ır</u>	
3	FECTIVE		
4	EF	MODE	
2			
9	C	>	
7	,	-	
80	,	-	
6		Ľ	
10	EGISTER		
1		_	
12	c	>	
13	c	>	
14		>	
15	-	>	

Instruction Fields:

Register field—Specifies the data register that contains the bit number.

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	I
Mode	111	111	1			ı	ı
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_							
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	I	010	011	100	101	110
Addressing Mode Mode	Dn*	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,An,Xn)**	110	reg. number:An	(bd,PC,Xn)
([bd,An,Xn],od)	110	reg. number:An	([bd,PC,Xn],o
([bd,An],Xn,od)	110	reg. number:An	([bd,PC],Xn,o

(12,4,5)	[bd,PC,Xn],od)	[bd,PC],Xn,od)
_		
reg. number:An	reg. number:An	reg. number:An
110	110	110
1,An,Xn)**	An,Xn],od)	An],Xn,od)

|

1 |

Integer Instructions

BCLR

Test a Bit and Clear (M68000 Family)

BCLR

Instruction Format:

BIT NUMBER STATIC, SPECIFIED AS IMMEDIATE DATA

0		œ	
-	SS	EGISTE	
2	ADDRES	₩	
က	ECTIVE		MBER
4	EFFE	MODE	BIT NU
2			
9		>	
7	1		
œ	0		0
6	0		0
10	0		0
#	-		0
12	0		0
13	0		0
4	c	>	0
15	c	>	0

Instruction Fields:

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register

001 000

Mode	Mode	111	111	ı			ı	I
Addisoning Mode	Addressing Mode Mode	W.(xxx)	L(xxx).L	/ <data< th=""><th></th><th></th><th>(d₁₆,PC)</th><th>(d₈,PC,Xn)</th></data<>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_								
Dogiotor	Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	Mode	000	1	010	011	100	101	110
Address and	Addressing Mode Mode	Dn*	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

1 I

|

Bit Number field—Specifies the bit number.

MOTOROLA

^{*}Long only; all others are byte only. **Can be used with CPU32.

^{*}Long only; all others are byte only. **Can be used with CPU32.

BRA

BRA

Branch Always (M68000 Family)

PC+d, → PC Operation Assemble

Size = (Byte, Word, Long*) Attributes Syntax:

BRA < label >

*(MC68020, MC68030, MC68040 only)

Description: Program execution continues at location (PC) + displacement. The program The displacement is a twos complement integer that represents the relative distance in bytes from the current program counter to the destination program counter. If the 8-bit displacement field in the instruction word is zero, a 16-bit displacement (the word immediately following the instruction) is used. If the 8-bit displacement field in the instruction word is all ones (\$FF), the 32-bit displacement (long word immediately counter contains the address of the instruction word of the BRA instruction plus two. following the instruction) is used.

Condition Codes:

Not affected.

Instruction Format:

0			
-			
7	ENT		
က	ACEME		
4	IT DISPI	00\$	T = \$FF
2	8-B	MENT =	FMFNT =
9		SPLACE	ISPI ACE
7		3-BIT DIS	AIT D
œ	0	ENT IF 8	FNT IF 8-
6	0	LACEM	I ACEMEN
10	0	BIT DISF	RIT DISPI
£	0	16-E	32-F
12	0		
13	1		
4	-		
15	0		
			_

Instruction Fields:

- 8-Bit Displacement field—Twos complement integer specifying the number of bytes between the branch instruction and the next instruction to be executed
- 16-Bit Displacement field—Used for a larger displacement when the 8-bit displacement is equal to \$00.
- 32-Bit Displacement field—Used for a larger displacement when the 8-bit displacement is equal to \$FF.

NOTE

A branch to the immediately following instruction automatically uses the 16-bit displacement format because the 8-bit displacement field contains \$00 (zero offset)

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

nteger Instructions

BSET

Test a Bit and Set (M68000 Family)

BSET

TEST (< bit number > of Destination) \rightarrow Z; 1 \rightarrow < bit number > of Des-Operation:

tination

BSET Dn, < ea > BSET # < data > , < ea > Assembler Syntax:

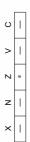
Size = (Byte, Long) Attributes:

appropriately, then sets the specified bit in the destination operand. When a data register is the destination, any of the 32 bits can be specified by a modulo 32-bit **Description**: Tests a bit in the destination operand and sets the Z condition code number. When a memory location is the destination, the operation is a byte operation and the bit number is modulo 8. In all cases, bit zero refers to the least significant bit. The bit number for this operation can be specified in either of two ways:

Immediate—The bit number is specified in the second word of the instruction.

Register—The specified data register contains the bit number.

Condition Codes:



X — Not affected.
N — Not affected.
Z — Set if the bit tested is zero; cleared otherwise.
V — Not affected.
C — Not affected.

4-56

BSET

Test a Bit and Set (M68000 Family)

BSET

Instruction Format:

BIT NUMBER DYNAMIC, SPECIFIED IN A REGISTER

Instruction Fields:

Register field—Specifies the data register that contains the bit number.

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	1
Mode	111	111	ı			1	ı
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_	Ü		:An	:An	:An	:An	:An
Register	reg. number:Dn	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	I	010	011	100	101	110
Addressing Mode Mode	Dn*	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd, PC, Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
		_
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

| 1

|

([pq,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	
110	110	
,Xn],od)	(poʻuXʻ[

^{*}Long only; all others are byte only. **Can be used with CPU32.

Integer Instructions

BSET

Test a Bit and Set (M68000 Family)

BSET

Instruction Format:

15 0

4 3 2 1

EFFECTIVE ADDRESS
IODE | REGISTER

9

MODE

_

REGISTER 9 7

0 12

0 13

0

4 15

BIT NUMBER STATIC, SPECIFIED AS IMMEDIATE DATA

0		~	
←	ESS	EGISTE	
2	ADDRE	ď	
က	FFECTIVE		BER
4	EFFE	MODE	TNUMB
2			<u>-</u> B
9	,	-	
7	,	-	
œ	c	>	
6	c	>	0
10	c	>	0
£	,	-	0
12	c	>	0
13	c	>	0
4		-	0
	_		_

Instruction Fields:

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register 000 100

Made	Mode	111	111	I			I	I
Addustration Manda	Addressing Mode Mode	W.(xxx)	L(xxx).L	/ <data< th=""><th></th><th></th><th>(d₁₆,PC)</th><th>(d₈,PC,Xn)</th></data<>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_								
	register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Medi	Mode	000	ı	010	011	100	101	110
Addison and	Addressing Mode Mode	Dn*	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

1 I

|

Bit Number field—Specifies the bit number.

MOTOROLA

4-57

^{*}Long only; all others are byte only. **Can be used with CPU32.

BSR

Branch to Subroutine (M68000 Family) BSR

 $SP-4 \rightarrow SP$; $PC \rightarrow (SP)$; $PC + d_n \rightarrow PC$ Operation:

BSR < label > Assembler Syntax: Size = (Byte, Word, Long*) Attributes: *(MC68020, MC68030, MC68040 only)

the instruction word plus two. Program execution then continues at location (PC) + displacement. The displacement is a twos complement integer that represents the counter. If the 8-bit displacement field in the instruction word is zero, a 16-bit displacement (the word immediately following the instruction) is used. If the 8-bit displacement field in the instruction word is all ones (\$FF), the 32-bit displacement **Description:** Pushes the long-word address of the instruction immediately following the BSR instruction onto the system stack. The program counter contains the address of relative distance in bytes from the current program counter to the destination program (long word immediately following the instruction) is used.

Condition Codes:

Not affected.

Instruction Format:

0			
-			
2	Þ		
က	ACEMEN		
4	8-BIT DISPL	009	3FF
2	8-BI) = L	MENT = \$
9		PLACEMEN	PLACEN
7		-BIT DIS	BIT DIS
80	-	16-BIT DISPLACEMENT IF 8-BIT	EMENT IF 8-BIT I
6	0	LACEME	AC
10	0	SIT DISP	32-BIT DISPL
7	0	16-E	32-BIT
12	0		
13	-		
14	-		
15	0		

Integer Instructions

BSR

Branch to Subroutine (M68000 Family)

BSR

8-Bit Displacement field—Twos complement integer specifying the number of bytes between the branch instruction and the next instruction to be executed. nstruction Fields:

16-Bit Displacement field—Used for a larger displacement when the 8-bit displacement is equal to \$00. 32-Bit Displacement field—Used for a larger displacement when the 8-bit displacement is equal to \$FF.

NOTE

uses the 16-bit displacement format because the 8-bit A branch to the immediately following instruction automatically displacement field contains \$00 (zero offset).

4-59

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

BTST

Test a Bit (M68000 Family)

BTST

TEST (< bit number > of Destination) \rightarrow Z Operation:

BTST Dn, < ea > BTST # < data > , < ea > Assembler

Size = (Byte, Long) Syntax:

Attributes:

Description: Tests a bit in the destination operand and sets the Z condition code appropriately. When a data register is the destination, any of the 32 bits can be specified by a modulo 32- bit number. When a memory location is the destination, the operation is a byte operation, and the bit number is modulo 8. In all cases, bit zero refers to the least significant bit. The bit number for this operation can be specified in either of two ways:

- 1. Immediate—The bit number is specified in a second word of the instruction.
- 2. Register—The specified data register contains the bit number.

Condition Codes:

ပ	I	
>	I	
Z	*	
z	I	
×	I	

X — Not affected.
N — Not affected.
Z — Set if the bit tested is zero; cleared otherwise.
V — Not affected.
C — Not affected.

Integer Instructions

BTST

Test a Bit (M68000 Family)

BTST

Instruction Format:

BIT NUMBER DYNAMIC, SPECIFIED IN A REGISTER

0		22
-	ESS	REGISTE
2	EADDRI	_
3	FECTIVI	
4	님	MODE
2		
9		>
7	c	>
80	,	-
6		Ľ
10	TEGICAL	
7		Ľ
12	c	>
13	c	>
4	c	>
15	c	>

Instruction Fields:

Register field—Specifies the data register that contains the bit number.

Effective Address field—Specifies the destination location. Only data addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Registe
Dn*	000	reg. number:Dn	W.(xxx)	111	000
An	I	1	L(xxx).L	111	100
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

011	011	011
=	111	111
(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)

^{*}Long only; all others are byte only. **Can be used with CPU32.

MOTOROLA

4-61

BTST

Test a Bit (M68000 Family)

BTST

Instruction Format:

BIT NUMBER STATIC, SPECIFIED AS IMMEDIATE DATA

0		R	
-	DRESS	REGISTER	
5	E ADDRE	<u></u>	
က	ECTIV		JMBER
4	EFFE	MODE	BIT N
2			
9	0		
7	0		
80	0		0
6	0		0
10	0		0
=	1		0
12	0		0
13	c	>	0
4		>	0
12		>	0

Instruction Fields:

Effective Address field—Specifies the destination location. Only data addressing modes can be used as listed in the following tables:

Register	000	100	ı			010	011
Mode	111	11	I			111	111
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	Ъ	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

	111	111
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)	([bd,An,Xn],od	([bd,An],Xn,od)

110 011

011

*Can be used with CPU32.

Bit Number field—Specifies the bit number.

Check Register Against Bounds (M68000 Family) CHK

CHK

Integer Instructions

If Dn < 0 or Dn > Source Then TRAP Operation:

CHK < ea > ,Dn Assembler Syntax:

Size = (Word, Long*) Attributes:

*(MC68020, MC68030, MC68040 only)

Description: Compares the value in the data register specified in the instruction to zero and to the upper bound (effective address operand). The upper bound is a twos complement integer. If the register value is less than zero or greater than the upper bound, a CHK instruction exception (vector number 6) occurs.

Condition Codes:

O	n
>	n
Z	n
z	*
×	I

X — Not affected. N — Set if Dn < 0; cleared if Dn > effective address operand; undefined otherwise. Z — Undefined. V — Undefined.

C — Undefined.

Instruction Format:

0		~	
-	SS	EGISTE	
7	E ADDRE	Ľ	
က	FECTIVE		
4	EF	MODE	
2			
9	c	>	
7	SIZE		
œ	SIS		
စ	R		
9	REGISTER		
7			
12	0		
13	0		
4	,	-	
15		-	

MOTOROLA

CHK

Check Register Against Bounds (M68000 Family)

Clear an Operand (M68000 Family)

CLR

CHK

0 → Destination

Operation: Assembler

CLR

Integer Instructions

Instruction Fields:

Register field—Specifies the data register that contains the value to be checked.

Size field—Specifies the size of the operation.

11— Word operation 10— Long operation

Effective Address field—Specifies the upper bound operand. Only data addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Register
D	000	reg. number:Dn	W.(xxx)	111	000
An	ı	I	L(xxx).L	11	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number: An			
(d ₁₆ ,An)	101	reg. number: An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number: An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
r:An	r:An	r:An
reg. number:An	reg. number:An	reg. number:An
reg.	reg.	reg.
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([pq,An],Xn,od)

011

111

011 011

111 1

*Can be used with CPU32.

Condition Codes:

Description: Clears the destination operand to zero. The size of the operation may be specified as byte, word, or long.

Size = (Byte, Word, Long)

Attributes:

CLR < ea >

Syntax:

X — Not affected.

N — Always cleared.
Z — Always set.
V — Always cleared.
C — Always cleared.

Instruction Format:

1 0	RESS	REGISTER
3 2	ECTIVE ADD	
4	EFFE	MODE
2		
9	SIZE	
7	ō	ō
80	0	
6	,	-
10	c	>
7	0	
12	0	
13	0	
4	,	-
15	c	>

MOTOROLA

CLR

Clear an Operand (M68000 Family)

Instruction Fields:

Size field—Specifies the size of the operation.

00—Byte operation 01—Word operation 10—Long operation

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	001	I			I	I
Mode	11	111	I			1	I
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	ı	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
	reg.		reg.	reg	reg.	reg	reg
Addressing Mode Mode	000 reg.	1	010 reg.	011 reg.	100 reg.	101 reg.	110 reg.

MC68020, MC68030, and MC68040 only

(pa,An,An)	2	reg. number.An	(Da, P.C, Arr)
([bd,An,Xn],od)	110	reg. number:An	([bd,PC,Xn],od)
([bd,An],Xn,od)	110	reg. number:An	([bd,PC],Xn,od)

I

*Can be used with CPU32.

In the MC68000 and MC68008 a memory location is read before

it is cleared.

NOTE

Integer Instructions

CMP

CMP

Compare (M68000 Family)

Destination – Source → cc Operation:

Assembler

CMP < ea > , Dn Syntax: Size = (Byte, Word, Long) Attributes:

Description: Subtracts the source operand from the destination data register and sets the condition codes according to the result; the data register is not changed. The size of the operation can be byte, word, or long.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×		

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.

C — Set if a borrow occurs; cleared otherwise.

Instruction Format:

2 1 0	ADDRESS	REGISTER	
Э	FECTIVE		
4	EFF	MODE	
2			
9	۱	Ц	
7	7		
80			
6	۱	צ	
10	TOIOL	2	
£		_	
12	,	-	
13	-		
4		>	
15	_	-	

Instruction Fields:

Register field—Specifies the destination data register.

Opmode field

Dn - < ea > Operation Long 010 Word 001 **Byte** 000

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

CMP

Compare (M68000 Family)

CMP

Effective Address field—Specifies the source operand. All addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regis
Dn	000	reg. number:Dn	W.(xxx)	111	000
An*	001	reg. number:An	J.(xxx)	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

111 011 011

111 11

NOTE

CMPA is used when the destination is an address register. CMPI is used when the source is immediate data. CMPM is used for memory-to-memory compares. Most assemblers automatically make the distinction.

Integer Instructions

Compare Address (M68000 Family) **CMPA**

CMPA

Destination – Source → cc

Assembler

Operation:

CMPA < ea > , An Syntax:

Description: Subtracts the source operand from the destination address register and sets size of the operation can be specified as word or long. Word length source operands the condition codes according to the result; the address register is not changed. The Size = (Word, Long) Attributes:

Condition Codes:

are sign- extended to 32 bits for comparison.

O	*
>	*
Z	*
z	*
×	1

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow is generated; cleared otherwise.
C — Set if a borrow is generated; cleared otherwise.

Instruction Format:

0		监
-	ESS	REGISTE
7	E ADDR	_
က	EFFECTIVI	
4	Ш	MODE
2		
9		Ш
7	100	
œ		
6		Ľ,
10	L	
7		_
12	,	-
13	,	-
4	c	>
15	,	_

4-76

^{*}Word and Long only.
**Can be used with CPU32.

CMPA

Compare Address (M68000 Family)

CMPA

Instruction Fields:

Register field—Specifies the destination address register.

Opmode field—Specifies the size of the operation.

011—Word operation; the source operand is sign-extended to a long operand, and the operation is performed on the address register using all 32 bits.

111— Long operation.

Effective Address field—Specifies the source operand. All addressing modes can be used as listed in the following tables:

Register	000	001	100			010	011
Mode	111	111	111			111	111
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	reg. number:An	reg. number: An	reg. number:An	reg. number: An	reg. number: An	reg. number: An
Mode	000	001	010	011	100	101	110
Addressing Mode Mode	Du	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)	([bd,An,Xn],od)	([bd,An],Xn,od)

011 011

1

011

7 =

CMPI

CMPI

Integer Instructions

Compare Immediate (M68000 Family)

Destination – Immediate Data → cc

Operation: Assembler

CMPI # < data > , < ea > Syntax:

Size = (Byte, Word, Long) Attributes:

Description: Subtracts the immediate data from the destination operand and sets the size of the operation may be specified as byte, word, or long. The size of the immediate condition codes according to the result; the destination location is not changed. The data matches the operation size.

Condition Codes:

O	*
>	*
Z	*
z	*
×	ı

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.
C — Set if a borrow occurs; cleared otherwise.

Instruction Format:

				_
>		~		
-	SS	REGISTER		
N	E ADDRE	<u>~</u>		
3	EFFECTIVE		YTE DATA	
4	EFF	MODE	8-BIT BYT	
Ω				
٥	1710	<u> </u>		ΤA
_	٥	0		NG DA
œ		>		32-BIT LONG DATA
ກ	c	>		33
2	,	-	Ā	
F	,	-	JRD DAT	
Z	c	>	16-BIT WORD DATA	
5	c	>	19	
4	c	>		
13	c	-		

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Can be used with CPU32

CMPI

Compare Immediate (M68000 Family)

CMPI

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation

10 — Long operation

Effective Address field—Specifies the destination operand. Only data addressing modes can be used as listed in the following tables:

Register	000	001	1			010	011
Mode	111	111	1			111	111
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)*	(d ₈ ,PC,Xn)*
ster	oer:Dn		nber:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Register	reg. number:Dn	I	reg. number:An	reg. nur	reg. nur	reg. nur	reg. nui
Addressing Mode Mode Regis	000 reg. num		010 reg. nun	011 reg. nur	100 reg. nur	101 reg. nur	110 reg. nur

MC68020, MC68030, and MC68040 only

	011	011	011
	111	111	111
	(bd,PC,Xn)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
	٦	Æ	:An
•	reg. number:Ar	reg. number:An	reg. number:An
	110 reg. number:/	110 reg. number:	110 reg. number

^{*}PC relative addressing modes do not apply to MC68000, MC680008, or MC6801 **Can be used with CPU32.

Immediate field—Data immediately following the instruction.

If size = 00, the data is the low-order byte of the immediate word. If size = 01, the data is the entire immediate word. If size = 10, the data is the next two immediate words.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-80

MOTOROLA

Integer Instructions

CMPM

Compare Memory (M68000 Family) CMPM

Destination – Source → cc Operation:

Assembler

CMPM (Ay) + ,(Ax) + Syntax: Size = (Byte, Word, Long) Attributes:

operands are always addressed with the postincrement addressing mode, using the address registers specified in the instruction. The size of the operation may be Description: Subtracts the source operand from the destination operand and sets the condition codes according to the results; the destination location is not changed. The specified as byte, word, or long.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×	I	

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow is generated; cleared otherwise.
C — Set if a borrow is generated; cleared otherwise.

Instruction Format:

0	0	_	REGISTER AX	Ţ	-	SIZE	0 0	٥	-	REGISTER AY	Y A	
Instruction Fields	n Fiel	ds:										
Regis	ter Ax	field	Register Ax field—(always the destination) Specifies an address register in the	ح م	estination)	Specif	, ve	,,	address	register	.⊆	the

6

9

7 12

13

4

15

0

<u>e</u> = מאומו <u>_</u> idiloli) openiies ster Ax iteid—(always trie dest postincrement addressing mode.

Size field—Specifies the size of the operation.

00 — Byte operation

01 — Word operation

10 — Long operation

Register Ay field—(always the source) Specifies an address register in the postincrement addressing mode.

DBcc DBcc

Test Condition, Decrement, and Branch (M68000 Family)

If Condition False Operation:

Then (Dn – 1 \rightarrow Dn; If Dn \neq – 1 Then PC + d_n \rightarrow PC)

Assembler

DBcc Dn, < label > Syntax:

Size = (Word) Attributes:

displacement is a twos complement integer that represents the relative distance in bytes from the current program counter to the destination program counter. Condition code cc specifies one of the following conditional tests (refer to Table 3-19 for more termination; if it is true, no operation is performed. If the termination condition is not true, the low-order 16 bits of the counter data register decrement by one. If the result is -1, execution continues with the next instruction. If the result is not equal to -1, execution continues at the location indicated by the current value of the program counter plus the sign-extended 16-bit displacement. The value in the program counter is the address of the instruction word of the DBcc instruction plus two. The Description: Controls a loop of instructions. The parameters are a condition code, a data register (counter), and a displacement value. The instruction first tests the condition for information on these conditional tests)

Condition	Low or Same	Less Than	Minus	Not Equal	Plus	True	Overflow Clear	Overflow Set
Mnemonic	ST	בו	M	NE	4	⊢	ΛC	SA
1								
Condition	Carry Clear	Carry Set	Equal	False	Greater or Equal	Greater Than	High	Less or Equal

Condition Codes:

Not affected.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-90

Integer Instructions

Test Condition, Decrement, and Branch (M68000 Family) DBcc

DBcc

Instruction Format:

Instruction Fields:

Condition field—The binary code for one of the conditions listed in the table.

Register field—Specifies the data register used as the counter.

Displacement field—Specifies the number of bytes to branch.

high-level languages. For example: DBMI can be stated as "decrement and branch until minus". The terminating condition is similar to the UNTIL loop clauses of

Most assemblers accept DBRA for DBF for use when only a count terminates the loop (no condition is tested).

A program can enter a loop at the beginning or by branching to the trailing DBcc instruction. Entering the loop at the beginning is useful for indexed addressing modes and dynamically execution count. In this case, if a zero count occurs, the DBcc specified bit operations. In this case, the control index count must be one less than the desired number of loop executions. However, when entering a loop by branching directly to the trailing DBcc instruction, the control count should equal the loop instruction does not branch, and the main loop is not executed.

DIVS, DIVSL

DIVS, DIVSL

Signed Divide (M68000 Family)

Destination + Source → Destination

Operation:

Assembler Syntax:

DIVS.W < ea > ,Dn32/16 → 16r − 16q *DIVS.L < ea > ,Dq *DIVS.L < ea > ,Dr:Dq *DIVS.L < ea > ,Dr:Dq *DIVSL.L < ea > ,Dr:Dq

Applies to MC68020, MC68030, MC68040, CPU32 only

Size = (Word, Long)

Description: Divides the signed destination operand by the signed source operand and stores the signed result in the destination. The instruction uses one of four forms. The word form of the instruction divides a long word by a word. The result is a quotient in the lower word (least significant 16 bits) and a remainder in the upper word (most significant 16 bits). The sign of the remainder is the same as the sign of the dividend.

The first long form divides a long word by a long word. The result is a long quotient; the remainder is discarded. The second long form divides a quad word (in any two data registers) by a long word. The result is a long-word quotient and a long-word remainder The third long form divides a long word by a long word. The result is a long-word quotient and a long-word remainder.

Two special conditions may arise during the operation:

- 1. Division by zero causes a trap.
- struction detects an overflow, it sets the overflow condition code, and the oper-2. Overflow may be detected and set before the instruction completes. If the inands are unaffected.

Condition Codes

X—Not affected.

- N Set if the quotient is negative; cleared otherwise; undefined if overflow or divide by zero occurs
- Set if the quotient is zero; cleared otherwise; undefined if overflow or divide by zero occurs.
 - Set if division overflow occurs; undefined if divide by zero occurs; cleared oth-
- C Always cleared.

4-92

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

Integer Instructions

DIVS, DIVSL

DIVS, DIVSL

Instruction Format:

Signed Divide (M68000 Family)

. 3 2 1 EFFECTIVE ADDRESS WORD REGISTER 9

Instruction Fields:

0 12

0 13

0 4

Register field—Specifies any of the eight data registers. This field always specifies the destination operand.

Effective Address field—Specifies the source operand. Only data alterable addressing modes can be used as listed in the following tables

Register	000	100	100			010	011
Mode	111	111	111			111	111
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
	_		_	_			
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Idressing Mode Mode	Du	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([pd,An,Xn],od)	([bd,An],Xn,od)

1 111 11

011 011 11

Overflow occurs if the quotient is larger than a 16-bit signed

^{*}Can be used with CPU32.

DIVS, DIVSL

Signed Divide (M68000 Family)

DIVS, DIVSL

Instruction Format:

LONG

0		œ	ō
-	ESS	REGISTE	EGISTER
7	EADDF		æ
9	FECTIV		0
4	EF	MODE	0
2			0
9	,	-	0
7	c	>	0
80	c	>	0
6	c	>	0
10	,	-	SIZE
=	,	-	1
12		>	Ба
13	c	>	SISTER
4	,		REG
15	c	>	0

Instruction Fields:

Effective Address field—Specifies the source operand. Only data alterable addressing modes can be used as listed in the following tables:

MC68020, MC68030, and MC68040 only

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regis
П	000	reg. number:Dn	W.(xxx)	111	000
An	ı	I	L(xxx).L	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011
(bd,An,Xn)	110	reg. number: An	(bd,PC,Xn)	111	110

MC68020, MC68030, and MC68040 only

([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	
110	110	
([pd,An,Xn],od)	([bd,An],Xn,od)	

011 011

111 11 Register Dq field—Specifies a data register for the destination operand. The low-order 32 bits of the dividend comes from this register, and the 32-bit quotient is loaded into this register.

Size field—Selects a 32- or 64-bit division operation. 0 — 32-bit dividend is in register Dq. 1 — 64-bit dividend is in Dr – Dq.

Integer Instructions

DIVS, DIVSL

Signed Divide (M68000 Family)

DIVS, DIVSL

Register Dr field—After the division, this register contains the 32-bit remainder. If Dr and Dq are the same register, only the quotient is returned. If the size field is 1, this field also specifies the data register that contains the high-order 32 bits of the dividend.

Overflow occurs if the quotient is larger than a 32-bit signed integer.

MOTOROLA

Unsigned Divide (M68000 Family) DIVU, DIVUL

DIVU, DIVUL

Destination + Source → Destination Operation: Assembler Syntax:

DIVU.W < ea > ,Dn32/16 → 16r – 16q *DIVU.L < ea > ,Dq *DIVU.L < ea > ,Dr:Dq *DIVUL.L < ea > ,Dr:Dq *DIVUL.L < ea > ,Dr:Dq *DIVUL.L < ea > ,Dr:Dq 32/32 → 32r – 32q *DIVUL.L < ea > ,Dr:Dq

Applies to MC68020, MC68030, MC68040, CPU32 only.

Size = (Word, Long)

operand and stores the unsigned result in the destination. The instruction uses one of four forms. The word form of the instruction divides a long word by a word. The result is a quotient in the lower word (least significant 16 bits) and a remainder in the upper Divides the unsigned destination operand by the unsigned source word (most significant 16 bits) Description:

The first long form divides a long word by a long word. The result is a long quotient; the remainder is discarded.

The second long form divides a quad word (in any two data registers) by a long word. The result is a long-word quotient and a long-word remainder The third long form divides a long word by a long word. The result is a long-word quotient and a long-word remainder.

Two special conditions may arise during the operation:

- 1. Division by zero causes a trap.
- struction detects an overflow, it sets the overflow condition code, and the oper-2. Overflow may be detected and set before the instruction completes. If the inands are unaffected.

Condition Codes:

— Not affected

- Set if the quotient is negative; cleared otherwise; undefined if overflow or divide
- Set if the quotient is zero; cleared otherwise; undefined if overflow or divide by zero occurs.
- -Set if division overflow occurs; cleared otherwise; undefined if divide by zero
- C Always cleared.

4-96

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

Integer Instructions

DIVU, DIVUL

Unsigned Divide (M68000 Family)

DIVU, DIVUL

Instruction Format:

WORD

	_	_
1 0	SS	EGISTER
2	ADDRE	ĸ
8	EFFECTIVE	
4	Ш	MODE
2		
9	,	-
7	,	
80	c	0
6		۲
10	L	
1		Ľ
12	c	0
13	c	0
14	c	0
15	-	-

Instruction Fields:

Register field—Specifies any of the eight data registers; this field always specifies the destination operand. Effective Address field—Specifies the source operand. Only data addressing modes can be used as listed in the following tables:

MC68020, MC68030, and MC68040 only

Register 000 00 100

	Mode	111	111	111			111	111
	Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
		<u>ا</u>		An	An	An	An	An
•	Register	reg. number:Dn	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
	æ	reg. n		reg. n	reg. n	reg. n	reg. n	reg. n
	Mode	000	I	010	011	100	101	110
	Addressing Mode Mode						<u></u>	(n)
	ressing	Б	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)
	용							

010 11

MC68020, MC68030, and MC68040 only

		-	
([bd,PC],Xn	reg. number:An	110	bd,An],Xn,od)
([bd,PC,Xn]	reg. number:An	110	bd,An,Xn],od)
(bd,PC,Xn	reg. number:An	110	(bd,An,Xn)*

^{**}Can be used with CPU32.

011	110	110
111	111	111
(bd,PC,Xn)*	([pq,PC,Xn],od)	([bd,PC],Xn,od)

NOTE

Overflow occurs if the quotient is larger than a 16-bit signed

DIVU, DIVUL

Unsigned Divide (M68000 Family)

DIVU, DIVUL

Instruction Format:

LONG

0		H.	٦
-	ESS	REGISTE	EGISTER
5	∃ADDR I	_	RE
က	FFECTIVE		0
4	EFF	MODE	0
2			0
9	7	-	0
7	c	>	0
8	c	>	0
6	c	>	0
10	,	-	SIZE
1	,	-	0
12	c	>	ρd
13	c	>	GISTER
4	7	-	RE
15	c	>	0

Instruction Fields:

Effective Address field—Specifies the source operand. Only data addressing modes can be used as listed in the following tables:

MC68020, MC68030, and MC68040 only

Register 000 100 100

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	_
ď	000	reg. number:Dn	W.(xxx)	11	
An	I	1	L (xxx)	111	
(An)	010	reg. number:An	# <data></data>	11	
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	
(bd,An,Xn)*	110	reg. number: An	(bd,PC,Xn)*	111	

010 011 011

MC68020, MC68030, and MC68040 only

([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An
110	110
([bd,An,Xn],od)	([bd,An],Xn,od)

011 011

111 11 Register Dq field—Specifies a data register for the destination operand. The low-order 32 bits of the dividend comes from this register, and the 32-bit quotient is loaded into this register.

Size field—Selects a 32- or 64-bit division operation. 0 — 32-bit dividend is in register Dq. 1 — 64-bit dividend is in Dr – Dq.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-98

MOTOROLA

Integer Instructions

DIVU, DIVUL

Unsigned Divide (M68000 Family)

DIVU, DIVUL

Register Dr field—After the division, this register contains the 32-bit remainder. If Dr and Dq are the same register, only the quotient is returned. If the size field is 1, this field also specifies the data register that contains the high-order 32 bits of the dividend.

Overflow occurs if the quotient is larger than a 32-bit unsigned integer.

EOR

Exclusive-OR Logical (M68000 Family)

Source ⊕ Destination → Destination Operation:

Assembler Syntax:

EOR Dn, < ea >

Size = (Byte, Word, Long) Attributes:

operation may be specified to be byte, word, or long. The source operand must be a data register. The destination operand is specified in the effective address field. **Description:** Performs an exclusive-OR operation on the destination operand using the source operand and stores the result in the destination location. The size of the

Condition Codes:

ပ	0
>	0
Z	*
z	*
×	Π

X — Not affected.
N — Set if the most significant bit of the result is set; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

WORD

	_	
0		œ
-	SS	EGISTE
7	ADDR	<u></u>
က	FECTIVE	
4	EFF	MODE
2		
9	١.	
7		
00		
6	١,	,
10		
Ε		۲
12	,	-
13	,	-
4	-	>
15	,	-

Instruction Fields:

Register field—Specifies any of the eight data registers.

Opmode field

< ea > ⊕ Dn → < ea > Operation **Long** 110 **Byte Word** 100 101 M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-100

MOTOROLA

Integer Instructions

EOR

EOR

Exclusive-OR Logical (M68000 Family)

EOR

Effective Address field—Specifies the destination ope data alterable addressing modes can be used as listed in the following tables:

Register	000	001	I			I	I
Mode	111	111	ı			ı	ı
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	D	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

I	_	_	
I	I	I	
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
_	_	_	

*Can be used with CPU32.

NOTE

Memory-to-data-register operations are not allowed. Most assemblers use EORI when the source is immediate data.

EORI

Exclusive-OR Immediate (M68000 Family)

EORI

Immediate Data ⊕ Destination → Destination Operation:

EORI # < data > , < ea > Assembler Syntax:

Size = (Byte, Word, Long)

Attributes:

Description: Performs an exclusive-OR operation on the destination operand using the immediate data and the destination operand and stores the result in the destination location. The size of the operation may be specified as byte, word, or long. The size of the immediate data matches the operation size.

Condition Codes:

O	0
>	0
Z	*
z	*
×	П

X — Not affected.
N — Set if the most significant bit of the result is set; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

0		œ		
~	SS	REGISTER		
7	/E ADDRESS	œ	_	
က	EFFECTIVE		YTE DATA	
4	EFI	MODE	3-BIT BY	
2				
9	SIZE			⋖
7				NG DAT
80	0			32-BIT LC
6	-			8
10		>	⋖	
£	,	-	ORD DATA	
12	0		6-BIT WORD I	
13	0		16	
4	c	>		
15		>		

EORI Exclusive-OR Immediate (M68000 Family) EORI

Integer Instructions

Size field—Specifies the size of the operation.

Instruction Fields:

00— Byte operation 01— Word operation

10—Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register 00 001

Mode	111	111	I			1	I
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_	_	_					
Register	reg. number:Dn	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	Du	An	(An)	(An) +	(An) —	(d ₁₆ ,An)	(d ₈ ,An,Xn)

I

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)	([bd,An,Xn],od)	([bd,An],Xn,od

1

*Can be used with CPU32.

Immediate field—Data immediately following the instruction.

If size = 00, the data is the low-order byte of the immediate word. If size = 01, the data is the entire immediate word. If size = 10, the data is next two immediate words.

MOTOROLA

4-103

Exclusive-OR Immediate to Condition Code (M68000 Family)

Source ⊕ CCR → CCR Operation:

EORI # < data > ,CCR Assembler Syntax:

Size = (Byte) Attributes: **Description:** Performs an exclusive-OR operation on the condition code register using the immediate operand and stores the result in the condition code register (low-order byte of the status register). All implemented bits of the condition code register are affected.

Condition Codes

— Changed if bit 4 of immediate operand is one; unchanged otherwise.

— Changed if bit 3 of immediate operand is one; unchanged otherwise.

— Changed if bit 2 of immediate operand is one; unchanged otherwise.

Changed if bit 1 of immediate operand is one; unchanged otherwise.
 Changed if bit 0 of immediate operand is one; unchanged otherwise.

Instruction Format:

Supervisor (Privileged) Instructions

EORI to SR

Exclusive-OR Immediate to the Status Register (M68000 Family)

EORI to SR

Then Source ⊕ SR → SR If Supervisor State

Operation:

ELSE TRAP

Assembler Syntax:

EORI # < data > ,SR

Size = (Word)

Attributes:

Description: Performs an exclusive-OR operation on the contents of the status register using the immediate operand and stores the result in the status register. All implemented bits of the status register are affected.

Condition Codes:

X—Changed if bit 4 of immediate operand is one; unchanged otherwise.

N—Changed if bit 3 of immediate operand is one; unchanged otherwise.

Z—Changed if bit 2 of immediate operand is one; unchanged otherwise.

V—Changed if bit 1 of immediate operand is one; unchanged otherwise.

C—Changed if bit 0 of immediate operand is one; unchanged otherwise.

Instruction Format:

0	0	
-	0	
7	-	
က	-	
4	-	
2	-	
9	-	A.
7	0	RD DAT
œ	0	BIT WC
6	-	16
10	0	
7	-	
12	0	
13	0	
4	0	
15	0	

4-104

EXG

Exchange Registers (M68000 Family) EXG

 $Rx \leftarrow \rightarrow Ry$ Operation:

EXG Ax, Ay EXG Dx, Ay **EXG Dx, Dy** Assembler Syntax:

Size = (Long) Attributes: **Description**: Exchanges the contents of two 32-bit registers. The instruction performs three types of exchanges.

- Exchange data registers.
- 2. Exchange address registers.
- 3. Exchange a data register and an address register

Condition Codes:

Not affected.

Instruction Format:

0	Ŋ.
~	GISTER
7	RE
က	
4	
2	DPMODE
9	
7	
80	1
6	RX.
10	GISTER
7	RE
12	0
13	0
4	1
2	

Instruction Fields:

Register Rx field—Specifies either a data register or an address register depending on the mode. If the exchange is between data and address registers, this field always specifies the data register

Opmode field—Specifies the type of exchange.

01000—Data registers

01001—Address registers

10001—Data register and address register

Register Ry field—Specifies either a data register or an address register depending on the mode. If the exchange is between data and address registers, this field always specifies the address register

nteger Instructions

EXT, EXTB

Sign-Extend (M68000 Family)

EXT, EXTB

Destination Sign-Extended → Destination Operation:

EXT.L Dnextend word to long word EXT.W Dnextend byte to word Assembler Syntax:

EXTB.L Dnextend byte to long word (MC68020, MC68030 MC68040, CPU32)

Size = (Word, Long) Attributes:

register to a long word, by replicating the sign bit to the left. If the operation extends a byte to a word, bit 7 of the designated data register is copied to bits 15-8 of that data register. If the operation extends a word to a long word, bit 15 of the designated data register is copied to bits 31 - 16 of the data register. The EXTB form copies bit 7 of the designated register to bits 31 - 8 of the data register. Description: Extends a byte in a data register to a word or a long word, or a word in a data

Condition Codes:

O	0	
>	0	
Z	*	
z	*	
×	ı	

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format

0	œ	
~	EGISTER	
2	œ	
က	0	
4	0	
2	0	
9	ш	
7	DPMODE	
80		
6	0	
10	0	
7	1	
12	0	
13	0	
4	-	
15	0	

Instruction Fields:

Opmode field—Specifies the size of the sign-extension operation.

010—Sign-extend low-order byte of data register to word. 011—Sign-extend low-order word of data register to long.

111—Sign-extend low-order byte of data register to long.

Register field—Specifies the data register is to be sign-extended

MOTOROLA

ILLEGAL

Take Illegal Instruction Trap (M68000 Family) ILLEGAL

*SSP – 2 \rightarrow SSP; Vector Offset \rightarrow (SSP); SSP – 4 \rightarrow SSP; PC \rightarrow (SSP); SSP – 2 \rightarrow SSP; SR \rightarrow (SSP); Operation:

Illegal Instruction Vector Address → PC

*The MC68000 and MC68008 cannot write the vector offset and format code to the system stack.

Assembler

ILLEGAL Syntax:

Unsized Attributes: **Description:** Forces an illegal instruction exception, vector number 4. All other illegal instruction bit patterns are reserved for future extension of the instruction set and should not be used to force an exception.

Condition Codes:

Not affected.

Instruction Format:

Integer Instructions

JMP

Jump (M68000 Family)

JMP

Destination Address → PC Operation:

JMP < ea > Assembler Syntax:

Unsized Attributes: **Description:** Program execution continues at the effective address specified by the instruction. The addressing mode for the effective address must be a control addressing mode.

Condition Codes:

Not affected.

Instruction Format:

0		
		~
1	SS	REGISTE
2	E ADDRE	<u>«</u>
3	FECTIVE	
4	Ш	MODE
2		
9	7	-
7	,	-
8		>
6	,	-
10	,	-
11	,	-
12	c	>
13	c	>
14	,	-
15	c	>

Instruction Field:

Effective Address field—Specifies the address of the next instruction. Only control addressing modes can be used as listed in the following tables:

Register 000 001

Mode	111	111	ı			111	111
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	ı	reg. number:An	1	1	reg. number:An	reg. number:An
Mode	ı	I	010	I	1	101	110
Addressing Mode Mode	Dn	An	(An)	+ (An)	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

010 011

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

011	011	011
	111	111
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)

4-107

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Can be used with CPU32.

JSR

Jump to Subroutine (M68000 Family)

JSR

 $SP-4 \rightarrow Sp; PC \rightarrow (SP); Destination Address \rightarrow PC$ Operation:

Assembler Syntax:

JSR < ea >

Unsized Attributes: **Description:** Pushes the long-word address of the instruction immediately following the JSR instruction onto the system stack. Program execution then continues at the address specified in the instruction.

Condition Codes:

Not affected.

Instruction Format:

0		œ
-	SS	EGISTE
7	E ADDRE	<u></u>
3	FECTIVE	
4	EF	MODE
2		
9	c	>
7	,	-
80	c	>
6	۲	-
10	,	-
7	,	-
12		>
13	c	>
4	,	_
15	c	>

Instruction Field:

Effective Address field—Specifies the address of the next instruction. Only control addressing modes can be used as listed in the following tables:

Register	000	100	I			010	011
Mode	111	111	1			111	111
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	1	reg. number:An	1	ı	reg. number:An	reg. number:An
Mode	ı	ı	010	I	I	101	110
Addressing Mode Mode	Du	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

011 11

1 111 7

110

nteger Instructions

LEA

Load Effective Address (M68000 Family)

LEA

< ea > → An Operation:

LEA < ea > ,An Assembler Syntax:

Size = (Long) Attributes:

Description: Loads the effective address into the specified address register. All 32 bits of the address register are affected by this instruction.

Condition Codes:

Not affected.

Instruction Format:

0		œ	
-	SS	REGISTE	
7	ADDRE	<u>ır</u>	
က	FECTIVE		
4	Ш	MODE	
2			
9	,	_	
7	,	-	
00	,	-	
о	,	ER	
10	REGISTER	REGIST	LGIS II
7			
12		>	
13		>	
4	,	-	
15	٥	>	

Instruction Fields:

Register field—Specifies the address register to be updated with the effective address.

Effective Address field—Specifies the address to be loaded into the address register. Only control addressing modes can be used as listed in the following tables:

Register	000	100	I			010	011
Mode	111	111	ı			111	111
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	l	ı	reg. number:An	I	ı	reg. number:An	reg. number:An
Mode	ı	I	010	ı	ı	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

110 d)		reg. number:An	110	([bd,An],Xn,od) 11 *Can be used with CPU32.
110	_	ig. lidilibdi .ga	2	(no,lin
		reg. number:An	110	(bd,An,Xn)

⁰¹¹ 011 011 11 11 111 ([bd,PC,Xn],od) ([bd,PC],Xn,od) (bd,PC,Xn)*

4-109

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Can be used with CPU32.

Link and Allocate (M68000 Family)

LINK

SP – 4 \rightarrow SP; An \rightarrow (SP); SP \rightarrow An; SP + d_n \rightarrow SP

Operation:

Assembler Syntax:

LINK An, # < displacement >

Size = (Word, Long*) Attributes:

*MC68020, MC68030, MC68040 and CPU32 only.

Description: Pushes the contents of the specified address register onto the stack. Then loads the updated stack pointer into the address register. Finally, adds the displacement value to the stack pointer. For word-size operation, the displacement is the sign-extended word following the operation word. For long size operation, the displacement is the long word following the operation word. The address register occupies one long word on the stack. The user should specify a negative displacement in order to allocate stack area.

Condition Codes:

Not affected.

Instruction Format:

WORD

REGISTER 0 2 WORD DISPLACEMENT œ o 9 7 12 0 13 4 15

Instruction Format:

LONG

0	œ		
-	GISTEF		
2	R		
က	1		
4	0		
2	0		
9	0	EMENT	MENT
7	0	ISPLACE	DISPLACE
80	0	RDER D	RDER D
6	0	HIGH-O	LOW-O
10	0		
7	1		
12	0		
13	0		
4	-		
15	0		
			_

Integer Instructions

LINK

Link and Allocate (M68000 Family)

LINK

Instruction Fields:

Register field—Specifies the address register for the link.

Displacement field—Specifies the twos complement integer to be added to the stack

NOTE

LINK and UNLK can be used to maintain a linked list of local data and parameter areas on the stack for nested subroutine calls.

MOTOROLA

4-111

LSL, LSR

Logical Shift (M68000 Family)

LSL, LSR

Destination Shifted By Count → Destination Operation:

Assembler Syntax:

LSd Dx,Dy LSd # < data > ,Dy LSd < ea >

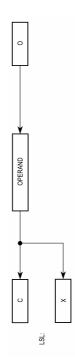
where d is direction, L or R

Size = (Byte, Word, Long) Attributes:

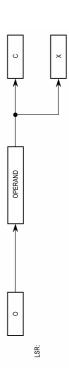
Description: Shifts the bits of the operand in the direction specified (L or R). The carry bit receives the last bit shifted out of the operand. The shift count for the shifting of a register is specified in two different ways:

- Immediate—The shift count (1 8) is specified in the instruction.
- Register—The shift count is the value in the data register specified in the instruction modulo 64.

The size of the operation for register destinations may be specified as byte, word, or long. The contents of memory, < ea >, can be shifted one bit only, and the operand size is restricted to a word. The LSL instruction shifts the operand to the left the number of positions specified as the shift count. Bits shifted out of the high-order bit go to both the carry and the extend bits; zeros are shifted into the low-order bit.



The LSR instruction shifts the operand to the right the number of positions specified as the shift count. Bits shifted out of the low-order bit go to both the carry and the extend bits; zeros are shifted into the high-order bit.



MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

nteger Instructions

LSL, LSR

Logical Shift (M68000 Family)

LSL, LSR

Condition Codes:

X — Set according to the last bit shifted out of the operand; unaffected for a shift count of zero.

N — Set if the result is negative; cleared otherwise.
 Z — Set if the result is zero; cleared otherwise.
 V — Always cleared.
 C — Set according to the last bit shifted out of the operand; cleared for a shift count

Instruction Format:

REGISTER SHIFTS

0	2
-	EGISTE
2	N.
က	1
4	0
2	i/r
9	SIZE
7	1S
œ	ф
6	- Y
10	COUNT/ EGISTEF
£	~
12	0
13	-
4	-
15	-

Instruction Fields:

Count/Register field

If i/r = 0, this field contains the shift count. The values 1 - 7 represent shifts of 1 - 7; value of zero specifies a shift count of eight. If i/r = 1, the data register specified in this field contains the shift count (modulo 64).

dr field—Specifies the direction of the shift. 0 — Shift right

1 — Shift left

Size field—Specifies the size of the operation.

00 — Byte operation

01 — Word operation

10 — Long operation i/r field

If i/r = 0, specifies immediate shift count.

If i/r = 1, specifies register shift count.

Register field—Specifies a data register to be shifted.

4-113

LSL, LSR

Logical Shift (M68000 Family)

LSL, LSR

Instruction Format:

MEMORY SHIFTS

Instruction Fields:

dr field—Specifies the direction of the shift.

0 — Shift right

1 — Shift left

Effective Address field—Specifies the operand to be shifted. Only memory alterable addressing modes can be used as listed in the following tables:

Register

000 00

Moc	111	111	ı			_	I
Addressing Mode Moc	(xxx).W	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	I	I	010	011	100	101	110
Addressing Mode Mode	ď	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([pd,An,Xn],od)	([pq,An],Xn,od)

^{*}Can be used with CPU32.

Integer Instructions

Move Data from Source to Destination (M68000 Family) MOVE

MOVE

Source → Destination Operation:

MOVE < ea > , < ea > Assembler Syntax:

Size = (Byte, Word, Long)

Attributes:

Description: Moves the data at the source to the destination location and sets the condition codes according to the data. The size of the operation may be specified as byte, word, or long. Condition Codes:

* z

X — Not affected.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

	_	
0		œ
-		REGISTE
7	OURCE	_
က	SOL	
4		MODE
2		
9		
7		MODE
80	NATION	
6	DESTI	œ
10		EGISTE
1		œ
12	2213	Ц
13	, ō	กั
4	c	>
15	_	_

Instruction Fields:

I | Size field—Specifies the size of the operand to be moved.

01 — Byte operation 11 — Word operation 10 — Long operation

I 1

1 1

4-115

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOVE

MOVE

Move Data from Source to Destination (M68000 Family)

Destination Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following tables:

Register

100 000

1

1 I

Mode	111	111	1			I	I
Addressing Mode Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_							
Register	reg. number:Dn	-	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	ď	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110 reg.	110 reg.	110 reg.
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

 $\perp \mid \perp$

I

*Can be used with CPU32.

Integer Instructions

MOVE

Move Data from Source to Destination (M68000 Family)

MOVE

Source Effective Address field—Specifies the source operand. All addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regist
Dn	000	reg. number:Dn	W.(xxx)	111	000
An	001	reg. number:An	(xxx).L	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)**	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

011 111 011

110

7 111

register direct is not allowed.	
address	22
*For byte size operation,	**Con house bear and across

Can be used with CPU32.

NOTE

Most assemblers use MOVEA when the destination is an address register. MOVEQ can be used to move an immediate 8-bit value to a data register.

MOTOROLA

MOVEA

Move Address (M68000 Family)

MOVEA

Source → Destination Operation:

MOVEA < ea > ,An Assembler Syntax:

Size = (Word, Long) Attributes:

Description: Moves the contents of the source to the destination address register. The size of the operation is specified as word or long. Word-size source operands are signextended to 32-bit quantities.

Condition Codes:

Not affected.

Instruction Format:

	_	
0		œ
-		REGISTER
2	URCE	
က	son	
4		MODE
2		
9	,	-
7	c	>
80	c	>
6	NO	œ
10	STINATI	EGISTE
1	Ö	ď
12	171	Ц
13	Ğ	36
4	c	>
2		

Instruction Fields:

Size field—Specifies the size of the operand to be moved.

11 — Word operation; the source operand is sign-extended to a long operand and all 32 bits are loaded into the address register.

10 — Long operation.

Destination Register field—Specifies the destination address register.

Integer Instructions

MOVEA

Move Address (M68000 Family)

MOVEA

Effective Address field—Specifies the location of the source operand. All addressing modes can be used as listed in the following tables:

Addressing Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	001	010	011	100	101	110
Addressing Mode Mode	uQ	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

ster	Addressing Mode Mode	Mode	Register
ber:Dn	W.(xxx)	111	000
ber:An	(xxx).L	111	001
ber:An	# <data></data>	111	100
ber:An			
ber:An			
ber:An	(d ₁₆ ,PC)	11	010
ber:An	(d ₈ .PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

*Can be used with CPU32.

(bd,PC,Xn)*	11	011	
([bd,PC,Xn],od)	111	011	
([bd,PC],Xn,od)	111	011	

4-119

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

MOVE to CCR

Move to Condition Code Register (M68000 Family)

Source → CCR Operation:

MOVE < ea > ,CCR Assembler Syntax:

Size = (Word) Attributes: **Description:** Moves the low-order byte of the source operand to the condition code register. The upper byte of the source operand is ignored; the upper byte of the status register is not altered.

Condition Codes:

O	*	
>	*	
Z	*	
z	*	
×		

X — Set to the value of bit 4 of the source operand.
N — Set to the value of bit 3 of the source operand.
Z — Set to the value of bit 2 of the source operand.
V — Set to the value of bit 1 of the source operand.
C — Set to the value of bit 0 of the source operand.

Instruction Format:

0		œ
-	SS	EGISTE
2	ADDRE	œ
ဗ	-ECTIVE	
4	H	MODE
2		
9	,	-
7	,	-
80	c	>
6	c	>
10	7	-
1	c	>
12	c	>
13	c	>
4	-	-
15	-	>

Integer Instructions

MOVE to CCR

Move to Condition Code Register (M68000 Family)

Effective Address field—Specifies the location of the source operand. Only data addressing modes can be used as listed in the following tables: Instruction Field:

Addre						
_						
Register	reg. number:Dn	1	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)

Register	Addressing Mode Mode	Mode	Register
reg. number:Dn	W.(xxx)	11	000
ı	L(xxx).L	111	100
reg. number:An	# <data></data>	111	100
reg. number:An			
reg. number:An			
reg. number:An	(d ₁₆ ,PC)	111	010
reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

110

(d₈,An,Xn)

(bd,An,Xn)*	110	reg. number:An
([bd,An,Xn],od	110	reg. number:An
([bd,An],Xn,od)	110	reg. number:An
*Can be used with CPU32.	·U32.	

110 011

([bd,PC],Xn,od) ([bd,PC,Xn],od)

011

11 1 11

(bd,PC,Xn)*

NOTE
_

MOVE to CCR is a word operation. ANDI, ORI, and EORI to CCR are byte operations.

4-123

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOVE from SR

MOVE from SR

Move from the Status Register (MC68000, MC68008)

SR → Destination Operation:

MOVE SR, < ea > Assembler Syntax: Size = (Word) Attributes: Description: Moves the data in the status register to the destination location. The

destination is word length. Unimplemented bits are read as zeros.

Condition Codes:

Not affected.

Instruction Format:

0		~
_	ESS	EGISTE
2	ADDRE	2
က	FCTIVE	
4	EFF	MODE
2		
9	,	-
7	,	-
œ	c	>
6	c	>
10	0	>
=	0	>
12	c	>
13	c	>
4	,	-
15		>

Instruction Fields:

Effective Address field—Specifies the destination location. Only data alterable addressing modes can be used as listed in the following table:

Register	000	001	1			1	Ι
Mode	111	111	ı			-	_
Addressing Mode Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	-	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	1	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

NOTE

Use the MOVE from CCR instruction to access only the condition codes. Memory destination is read before it is written to.

Supervisor (Privileged) Instructions

MOVE to SR

MOVE to SR

Move to the Status Register (M68000 Family)

Then Source → SR If Supervisor State

Operation:

Else TRAP

MOVE < ea > ,SR Assembler Syntax:

Size = (Word) Attributes: **Description:** Moves the data in the source operand to the status register. The source operand is a word, and all implemented bits of the status register are affected.

Condition Codes:

Set according to the source operand.

Instruction Format:

0		H
-	SS	EGISTE
7	ADDRE	<u>~</u>
က	FFECTIVE	
4	H	MODE
2		
9	,	-
7	,	-
œ		>
6	,	-
10	,	-
£	c	>
12	c	>
13	c	>
4	,	-
15		>

MOTOROLA

Supervisor (Privileged) Instructions

MOVE to SR

Move to the Status Register (M68000 Family)

MOVE to SR

Instruction Field:

Effective Address field—Specifies the location of the source operand. Only data addressing modes can be used as listed in the following tables:

Addressing Mode	Mode	Register	Ă
Dn	000	reg. number:Dn	
An	I	1	
(An)	010	reg. number:An	
(An) +	011	reg. number:An	
—(An)	100	reg. number:An	
(d ₁₆ ,An)	101	reg. number:An	
(d ₈ ,An,Xn)	110	reg. number:An	

Addressing Mode	Mode	Register
W.(xxx)	111	000
(xxx).L	111	001
# < data >	111	100
(d ₁₆ ,PC)	111	010
(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	(bo, [nX,nB])	(bo, nX,[nA,bd])

*Available for the CPU32.

011	011	011
111	111	111
(bd,PC,Xn)*	([bd,PC,Xn] ,od)	([bd,PC],Xn ,od)
_	_	_

Supervisor (Privileged) Instructions

MOVE USP

Move User Stack Pointer (M68000 Family)

MOVE USP

If Supervisor State Then USP \rightarrow An or An \rightarrow USP Else TRAP

Operation:

MOVE USP,An MOVE An,USP Assembler Syntax:

Size = (Long) Attributes: **Description:** Moves the contents of the user stack pointer to or from the specified address register.

Condition Codes:

Not affected.

Instruction Format:

_		
J	E.	
-	REGISTE	
2	Ľ	
က	ъ	
4	0	
2	-	
9	-	
7	0	
80	0	
6	-	
10	-	
7	-	
12	0	
13	0	
4	-	
15	0	

Instruction Fields:

dr field—Specifies the direction of transfer.

0—Transfer the address register to the user stack pointer.

1—Transfer the user stack pointer to the address register.

Register field—Specifies the address register for the operation.

6-20

MOVEM

Move Multiple Registers (M68000 Family)

MOVEM

Registers → Destination; Source → Registers

Operation:

MOVEM < list > , < ea > Assembler

MOVEM < ea > , < list > Syntax:

Size = (Word, Long) Attributes:

locations starting at the location specified by the effective address. A register is selected if the bit in the mask field corresponding to that register is set. The instruction size determines whether 16 or 32 bits of each register are transferred. In the case of a Description: Moves the contents of selected registers to or from consecutive memory word transfer to either address or data registers, each word is sign-extended to 32 bits, and the resulting long word is loaded into the associated register. Selecting the addressing mode also selects the mode of operation of the MOVEM instruction, and only the control modes, the predecrement mode, and the postincrement mode are valid. If the effective address is specified by one of the control modes, the registers are transferred starting at the specified address, and the address is incremented by the operand length (2 or 4) following each transfer. The order of the registers is from D0 to D7, then from A0 to A7. If the effective address is specified by the predecrement mode, only a register-to-memory operation is allowed. The registers are stored starting at the specified address minus the operand length (2 or 4), and the address is decremented by the operand length following each transfer. The order of storing is from A7 to A0, then from D7 to D0. When the instruction has completed, the decremented address register contains the address of the last operand stored. For the MC68020, MC68030, MC68040, and CPU32, if the addressing register is also moved to memory, the value written is the initial register value decremented by the size of the operation. The MC68000 and MC68010 write the initial register value (not decremented)

the address is incremented by the operand length (2 or 4) following each transfer. The order of loading is the same as that of control mode addressing. When the instruction has completed, the incremented address register contains the address of the last operand loaded plus the operand length. If the addressing register is also loaded from f the effective address is specified by the postincrement mode, only a memory-to-regmemory, the memory value is ignored and the register is written with the postincreister operation is allowed. The registers are loaded starting at the specified address; mented effective address

Integer Instructions

MOVEM

Move Multiple Registers (M68000 Family)

MOVEM

Condition Codes:

Not affected.

Instruction Format:

	_		_
0		œ	
-	SS	EGISTE	
2	ADDRE	₩	
3	FECTIVE		
4	H	MODE	
2			
9	SIZE		SK
7	-		LIST MA
80	c	>	3ISTER
6	c	>	RE
10	Ť	a	
7	1		
12	0		
13	0		
4	1		
15	0		

Instruction Fields:

dr field—Specifies the direction of the transfer

- 0 Register to memory.
 - 1 Memory to register

Size field—Specifies the size of the registers being transferred.

- 0 Word transfer
 - 1 Long transfer

to-memory transfers, only control alterable addressing modes or the predecrement addressing mode can be used as listed in the following tables: Effective Address field—Specifies the memory address for the operation. For register-

gister 00

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Re
Dn	I	-	W.(xxx)	111)
An	Ι	I	L(xxx)	111	
(An)	010	reg. number:An	# <data></data>	ı	
(An) +	ı	I			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	I	
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	ı	

MC68020, MC68030, and MC68040 only

reg. number.An	reg. number:An	reg. number:An	
2	110	110	
(pa,An,An)	[bd,An,Xn],od)	(ba,An],Xn,od)	

(bd,PC,Xn)* — — — (lbd,PC,Xn],od) — — — — — — — — — — — — — — — — — — —			
([bd,PC,Xn],od) — ([bd,PC,Xn],od) — ([bd,PC],Xn,od) —	-	-	_
(bd,PC,Xn)* ([bd,PC,Xn],od) ([bd,PC],Xn,od)	I	I	_
	(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)

4-128

^{*}Can be used with CPU32.

MOVEM

Move Multiple Registers (M68000 Family)

MOVEM

For memory-to-register transfers, only control addressing modes or the postincrement addressing mode can be used as listed in the following tables:

Mode	111	111	I			111	111
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	1	-	reg. number:An	reg. number:An	1	reg. number:An	reg. number:An
Mode	ı	1	010	011	1	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

1

010

011

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

011

011

011

7 11 7

*Can be used with CPU32.

Register List Mask field—Specifies the registers to be transferred. The low-order bit corresponds to the first register to be transferred; the high-order bit corresponds to the last register to be transferred. Thus, for both control modes and postincrement mode addresses, the mask correspondence is:

0	D0	
-	D1	-
2	D2	
က	D3	
4	72	•
2	DS	
9	9G	-
7	D7	
œ	A0	
6	A1	
10	A2	-
=	A3	•
12	A4	
13	A5	
4	9V	
15	A7	L

For the predecrement mode addresses, the mask correspondence is reversed:

0	A7
-	9V
2	A5
ო	A4
4	A3
2	A2
9	H4
7	A0
80	2 0
6	9 0
10	9Q
7	P4
12	EQ
13	D2
4	D1
15	0G

Integer Instructions

Move Peripheral Data (M68000 Family) MOVEP

MOVEP

Source → Destination Operation:

MOVEP Dx,(d16,Ay) MOVEP (d16,Ay),Dx Assembler Syntax:

Size = (Word, Long) Attributes:

Register 000 001

peripherals on a 16-bit data bus, such as the MC68000 bus. Although supported by the MC68020, MC68030, and MC68040, this instruction is not useful for those processors **Description:** Moves data between a data register and alternate bytes within the address space starting at the location specified and incrementing by two. The high-order byte of the data register is transferred first, and the low-order byte is transferred last. The addressing mode. This instruction was originally designed for interfacing 8-bit memory address is specified in the address register indirect plus 16-bit displacement with an external 32-bit bus.

Example: Long transfer to/from an even address.

Byte Organization in Register

0 2	LOW ORDER	
15 8	MID LOWER	Organization in -Bit Memory
23 16	MID UPPER	Byte Organization 16-Bit Memory
31 24	HIGH ORDER	

0				
7				
œ	DER	PER	VER	DER
15	HIGH ORDER	MID UPPER	MID LOWER	LOW ORDER

(Low Address at Top)

MOVEP

Move Peripheral Data (M68000 Family)

MOVEP

Byte Organization in 32-Bit Memory

0			
7			
ω	PER	RDER	
15	MID UPPER	LOW ORDER	
16			
23			
24	ER	ER	
-	HIGH ORDER	MID LOWER	

ō

0			
7			
80	HIGH ORDER	MID LOWER	
15			
16			
23			
24		MID UPPER	LOW ORDER
31			

Example:Word transfer to/from (odd address).

Byte Organization in Register

	_
0	LOW ORDER
7	
∞	HIGH ORDER
15	HIG
16	
23	
24	
31	

Byte Organization in 16-Bit Memory (Low Address at Top)

0	HIGH ORDER	W ORDER
7	HIG	ГО
80		
15		
	_	

Byte Organization in 32-Bit Memory

0	HIGH ORDER			0	LOW ORDER
7	ÐH			7	0
80				œ	
15				15	
16 15		(DER	ō	16 15	RDER
23		LOW ORDER		23	HIGH ORDER
24 23				24	
31				31	

MOVEP

Move Peripheral Data (M68000 Family)

Integer Instructions MOVEP

Condition Codes:

Not affected.

Instruction Format:

9876543210	ISTER OPMODE 0 0 1 ADDRESS REGISTER	HATTATION GOIG HIG OF
,	0	
9		Ŀ
7	PMOD	LATION
∞		FIG
တ	STER	0,
9	A REGIS	
7	DAT	
12	0	
13	0	
4	0	
15	0	1

Instruction Fields:

Data Register field—Specifies the data register for the instruction.

Opmode field—Specifies the direction and size of the operation.

100—Transfer word from memory to register. 101—Transfer long from memory to register. 110— Transfer word from register to memory. 111— Transfer long from register to memory.

Address Register field—Specifies the address register which is used in the address register indirect plus displacement addressing mode.

Displacement field—Specifies the displacement used in the operand address.

MOVEQ

Move Quick (M68000 Family)

MOVEQ

MULS

Immediate Data → Destination Operation:

MOVEQ # < data > ,Dn Assembler

Syntax:

Size = (Long)

Attributes:

Description: Moves a byte of immediate data to a 32-bit data register. The data in an 8-bit field within the operation word is sign- extended to a long operand in the data register

Condition Codes

as it is transferred.

0 * * > Ν z

X — Not affected. N — Set if the result is negative; cleared otherwise. Z — Set if the result is zero; cleared otherwise. V — Always cleared.

C — Always cleared.

Instruction Format:

Instruction Fields:

Register field—Specifies the data register to be loaded.

Data field—Eight bits of data, which are sign-extended to a long operand.

Integer Instructions MULS Signed Multiply (M68000 Family)

Source x Destination → Destination Operation:

*MULS.L < ea > ,DI $32 \times 32 \rightarrow 32$ *MULS.L < ea > ,Dh - DI $32 \times 32 \rightarrow 64$ MULS.W < ea > ,Dn16 x 16 \rightarrow 32 Assembler Syntax:

*Applies to MC68020, MC68030, MC68040, CPU32

Size = (Word, Long) Attributes:

Description: Multiplies two signed operands yielding a signed result. This instruction has a word operand form and a long operand form. In the word form, the multiplier and multiplicand are both word operands, and the result is a long-word operand. A register operand is the low-order word; the upper word of the register is ignored. All 32 bits of the product are saved in the destination data register. In the long form, the multiplier and multiplicand are both long- word operands, and the result is either a long word or a quad word. The long-word result is the low-order 32 bits of the quad- word result; the high-order 32 bits of the product are discarded.

Condition Codes:



X — Not affected.

N — Set if the result is negative; cleared otherwise.

Z — Set if the result is zero; cleared otherwise.
 V — Set if overflow; cleared otherwise.
 C — Always cleared.

operands to yield a 32-bit result. Overflow occurs if the high-order 32 bits of the quad-word product are not the sign extension Overflow (V = 1) can occur only when multiplying 32-bit of the low- order 32 bits

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MULS

Signed Multiply (M68000 Family)

Instruction Format:

WORD

Instruction Fields:

Register field—Specifies a data register as the destination.

Effective Address field—Specifies the source operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	100			010	011
Mode	111	111	111			11	111
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Mode	000	I	010	011	100	101	110
-							

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	(lpd,An,Xn],od)	([pq,An],Xn,od)

011 011

111

011

*Can be used with CPU32.

MULS

Signed Multiply (M68000 Family)

MULS

Integer Instructions

Instruction Format:

LONG

0		H	2 Dh
-	RESS	REGISTE	FGISTER
2	/E ADDF	_	2
က	EFFECTI		_
4	Ш	MODE	_
2			_
9	c	>	c
7	c	>	c
00	c	>	0
6		>	0
10	7	-	SIZE
7	-	-	-
12		>	i d
13		>	REGISTER
4	,	-	2
15		>	c

Instruction Fields:

Effective Address field—Specifies the source operand. Only data addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Registe
Du	000	reg. number:Dn	W.(xxx)	111	000
An	1	I	L(xxx).L	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	110

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
_	_	_	
reg. number:An	reg. number:An	reg. number:An	
. num	nuu.	. num	
reg	reg	reg	
110	110	110	
	(F	(F	
(bd,An,Xn)*	Xn],oc	,Xn,oc	
(bd,Aı	[bd,An,Xn],od)	(bd,An],Xn,od)	
	=	1	

011

11 011

1 111

Can be used with CPU32.

Register DI field—Specifies a data register for the destination operand. The 32-bit multiplicand comes from this register, and the low-order 32 bits of the product are loaded into this register.

Size field—Selects a 32- or 64-bit product.

0 — 32-bit product to be returned to register DI.

1 — 64-bit product to be returned to Dh – DI.

Register Dh field—If size is one, specifies the data register into which the high-order 32 bits of the product are loaded. If Dh = Dl and size is one, the results of the operation are undefined. Otherwise, this field is unused.

4-136

Unsigned Multiply (M68000 Family)

Source x Destination → Destination

Operation:

Assembler

MULU.W < ea > ,Dn16 \times 16 \rightarrow 32 *MULU.L < ea > ,DI 32 \times 32 \rightarrow 32 *MULU.L < ea > ,Dh - DI 32 \times 32 \rightarrow 64

Syntax:

Applies to MC68020, MC68030, MC68040, CPU32 only

Size = (Word, Long) Attributes:

Description: Multiplies two unsigned operands yielding an unsigned result. This instruction has a word operand form and a long operand form.

In the word form, the multiplier and multiplicand are both word operands, and the result is a long-word operand. A register operand is the low-order word; the upper word of the register is ignored. All 32 bits of the product are saved in the destination data register.

In the long form, the multiplier and multiplicand are both long-word operands, and the result is either a long word or a quad word. The long-word result is the low-order 32 bits of the quad- word result; the high-order 32 bits of the product are discarded.

Condition Codes:

> z *

- X Not affected.
 N Set if the result is negative; cleared otherwise.
 Z Set if the result is zero; cleared otherwise.
 V Set if overflow; cleared otherwise.
 C Always cleared.

Overflow (V = 1) can occur only when multiplying 32-bit operands to yield a 32-bit result. Overflow occurs if any of the high-order 32 bits of the quad-word product are not equal to zero.

Unsigned Multiply (M68000 Family)

Integer Instructions

Instruction Format:

WORD

0		E
-	ESS	REGISTER
2	E ADDR	_
က	FFECTIV	
4	Н	MODE
2		
9	7	-
7	,	-
80	c	>
6	ç	Ľ,
10	TTOIOL	
7		_
12		>
13	-	>
4	,	-
15	,	_

Instruction Fields:

Register field—Specifies a data register as the destination.

Effective Address field—Specifies the source operand. Only data addressing modes can be used as listed in the following tables:

lode Registe	111 000	111 001	111 100			111 010	777
Addressing Mode Mode	W.(xxx)	L(xxx).	# <data></data>			(d ₁₆ ,PC)	(5)
ter	er:Dn		er:An	er:An	er:An	er:An	
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	
Mode	000	I	010	011	100	101	770
Addressing Mode Mode	Ы	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(5/ 5/ 7/

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

011 11 011

11

1

*Can be used with CPU32.

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

Unsigned Multiply (M68000 Family)

MULU

Instruction Format:

LONG

REGISTER Dh REGISTER EFFECTIVE ADDRESS 0 9 0 0 0 6 SIZE 10 _ 7 0 12 REGISTER DI 13 0 _ 4 0 15

Instruction Fields:

Effective Address field—Specifies the source operand. Only data addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regist
D	000	reg. number:Dn	W.(xxx)	111	000
An	I	I	J.(xxx)	111	001
(An)	010	reg. number:An	# <data></data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

11	111	111
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

011 011

011

*Can be used with CPU32

Register DI field—Specifies a data register for the destination operand. The 32-bit multiplicand comes from this register, and the low-order 32 bits of the product are loaded into this register.

Size field—Selects a 32- or 64-bit product.

0 — 32-bit product to be returned to register DI.

1 — 64-bit product to be returned to Dh – Dl.

Register Dh field—If size is one, specifies the data register into which the high-order 32 bits of the product are loaded. If Dh = Dl and size is one, the results of the operation are undefined. Otherwise, this field is unused.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

Integer Instructions

NBCD

Negate Decimal with Extend (M68000 Family) NBCD

 $0-Destination_{10}-X \rightarrow Destination$ Operation:

NBCD < ea > Assembler Syntax:

Size = (Byte) Attributes:

result is saved in the destination location. This instruction produces the tens complement of the destination if the extend bit is zero or the nines complement if the **Description:** Subtracts the destination operand and the extend bit from zero. The operation is performed using binary-coded decimal arithmetic. The packed binary-coded decimal extend bit is one. This is a byte operation only.

Condition Codes:

ပ	*	
>	n	
Z	*	
z	n	
×	*	

— Set the same as the carry bit

N — Undefined.
Z — Cleared if the result is nonzero; unchanged otherwise.
V — Undefined.
C — Set if a decimal borrow occurs; cleared otherwise.

Normally the Z condition code bit is set via programming before the start of the operation. This allows successful tests for zero results upon completion of multiple-precision operations.

NBCD

Negate Decimal with Extend (M68000 Family)

NBCD

Instruction Format:

Instruction Fields:

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	-
Mode	111	111	ı			1	-
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
	Du		r:An	er:An	ır:An	er:An	ər:An
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Registe	000 reg. number		010 reg. numbe	011 reg. numb	100 reg. numbe	101 reg. numbe	110 reg. numbe

MC68020, MC68030, and MC68040 only

I	I	I
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([pq,An,Xn],od)	([bd,An],Xn,od)

I

NEG

Negate (M68000 Family)

NEG

Integer Instructions

0 – Destination → Destination Operation:

NEG < ea > Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

Description: Subtracts the destination operand from zero and stores the result in the destination location. The size of the operation is specified as byte, word, or long.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×	*	

X — Set the same as the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.
C — Cleared if the result is zero; set otherwise.

Instruction Format:

0		STER
	ADDRESS	REGI
က	FECTIVE	
4	EFI	MODE
2		
9	1213	<u> </u>
7		ñ
80	c	>
6	ď	>
10	4	-
£	C	>
12		-
13		-
4	,	-
15	-	>

^{*}Can be used with CPU32.

NEG

Negate (M68000 Family)

NEG

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	I
Mode	11	111	ı			1	I
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn		reg. number:An	reg. number:An	reg. number:An	reg. number: An	reg. number:An
Re	reg. nu	ľ	reg. nu	reg. nu	reg. nu	reg. nu	reg. nur
Addressing Mode Mode Re	000 reg. nu	1	010 reg. nu	011 reg. nu	100 reg. nu	101 reg. nu	110 reg. nur

MC68020, MC68030, and MC68040 only

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)	([bd,An,Xn],od)	([bd,An],Xn,od)	

([bd,PC,Xn],od) (bd,PC,Xn)*

*Can be used with CPU32.

NEGX

Negate with Extend (M68000 Family)

Integer Instructions

NEGX

0 – Destination – $X \rightarrow$ Destination Operation:

Assembler

NEGX < ea > Syntax: Size = (Byte, Word, Long) Attributes:

Description: Subtracts the destination operand and the extend bit from zero. Stores the result in the destination location. The size of the operation is specified as byte, word, or long.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×	*	

X — Set the same as the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Cleared if the result is nonzero; unchanged otherwise.
V — Set if an overflow occurs; cleared otherwise.
C — Set if a borrow occurs; cleared otherwise.

Normally the Z condition code bit is set via programming before the start of the operation. This allows successful tests for zero results upon completion of multiple-precision operations.

4-144

NEGX

Negate with Extend (M68000 Family)

NEGX

Instruction Format:

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			I	I
Mode	11	111	ı			1	1
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
ster	nper:Dn		ıber:An	ıber:An	ıber:An	ıber:An	ıber:An
Register	reg. number:Dn		reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Addressing Mode Mode	υ	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,An,Xn)* 110	reg. number:An	(bd,PC,Xn)*
[bd,An,Xn],od) 110	reg. number:An	([pd,PC,Xn],od)
[bd,An],Xn,od) 110	reg. number:An	([bd,PC],Xn,od)

| 1

1

NOP

No Operation (M68000 Family)

NOP

Integer Instructions

None Operation:

Assembler Syntax:

NOP

Unsized Attributes: **Description:** Performs no operation. The processor state, other than the program counter, is unaffected. Execution continues with the instruction following the NOP instruction. The NOP instruction does not begin execution until all pending bus cycles have completed. This synchronizes the pipeline and prevents instruction overlap.

Condition Codes:

Not affected.

Instruction Format:

4-146

^{*}Can be used with CPU32.

NOT

Logical Complement (M68000 Family)

NOT

~ Destination → Destination Operation:

NOT < ea > Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

Description:Calculates the ones complement of the destination operand and stores the result in the destination location. The size of the operation is specified as byte, word, or long.

Condition Codes:

ပ	0	
>	0	
Ζ	*	
z	*	
×	_	

- X Not affected.
 N Set if the result is negative; cleared otherwise.
 Z Set if the result is zero; cleared otherwise.
 V Always cleared.
 C Always cleared.

Instruction Format:

REGISTER EFFECTIVE ADDRESS 9 SIZE 0 œ 6 10 _ 7 0 12 0 13 0 4 _ 0 15

NOT

Logical Complement (M68000 Family)

NOT

Integer Instructions

Instruction Fields:

Size field—Specifies the size of the operation.

00— Byte operation 01— Word operation 10— Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Register 00 001

Mode	111	11	1			I	I
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
	_						
Register	reg. number:Dn	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	I	010	011	100	101	110
Addressing Mode Mode	Б	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

ı

MC68020, MC68030, and MC68040 only

|

(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([pq,An,Xn],od)	([pd,An],Xn,od)	

*Can be used with CPU32.

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

Inclusive-OR Logical (M68000 Family) **OR**

Source V Destination → Destination Operation:

Assembler

OR < ea > ,Dn OR Dn, < ea > Syntax: Size = (Byte, Word, Long) Attributes:

Description: Performs an inclusive-OR operation on the source operand and the destination operand and stores the result in the destination location. The size of the operation is specified as byte, word, or long. The contents of an address register may not be used as an operand.

Condition Codes:

	_	
ပ	0	
>	0	
Z	*	
z	*	
×	П	

X — Not affected.
N — Set if the most significant bit of the result is set; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

0		œ
-	RESS	REGISTER
2	EADDR	
က	FFECTIVI	
4	EFF	MODE
2		
9		ш
7		
80		
6	٥	ć
10		
7		_
12		>
13	c	>
4		>
15	Į,	-

Instruction Fields:

Register field—Specifies any of the eight data registers.

Opmode field

Operation	$<$ ea $>$ V Dn \rightarrow Dn	Dn V < ea > → < ea >
Long	010	110
Word	001	101
Byte	000	100

OR

OR

Inclusive-OR Logical (M68000 Family)

Integer Instructions

OR R

Effective Address field—If the location specified is a source operand, only data addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	_	Addressing Mode Mode	Mode	Register
uQ	000	reg. number:Dn		W.(xxx)	111	000
An	ı	1		L(xxx)	111	100
(An)	010	reg. number:An		# <data></data>	111	100
(An) +	011	reg. number:An				
– (An)	100	reg. number:An				
(d ₁₆ ,An)	101	reg. number:An		(d ₁₆ ,PC)	111	010
(d ₈ ,An,Xn)	110	reg. number:An		(d ₈ ,PC,Xn)	111	011

MC68020, MC68030, and MC68040 only

111	111	111
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

011 011 011

*Can be used with CPU32.

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

OR

Inclusive-OR Logical (M68000 Family)

O R

If the location specified is a destination operand, only memory alterable addressing modes can be used as listed in the following tables:

Register	000	001	I			I	1
Mode	111	111	ı			ı	ı
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ .PC,Xn)
ster			iber:An	nber:An	nber:An	ıber:An	nber:An
Register	1	ļ	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Mode Regis	1	_	010 reg. num	011 reg. nun	100 reg. num	101 reg. num	110 reg. nun

MC68020, MC68030, and MC68040 only

I	1	1
I	I	I
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
Ā	An	An
reg. number:An	reg. number:An	reg. number:An
110 reg. number:	110 reg. number	110 reg. number:

*Can be used with CPU32.

NOTE

If the destination is a data register, it must be specified using the destination Dn mode, not the destination < ea > mode.

Most assemblers use ORI when the source is immediate data.

Integer Instructions

OR

OR

Inclusive-OR (M68000 Family)

Immediate Data V Destination → Destination Operation:

Assembler Syntax:

ORI # < data > , < ea >

Size = (Byte, Word, Long) Attributes:

Description: Performs an inclusive-OR operation on the immediate data and the destination operand and stores the result in the destination location. The size of the operation is specified as byte, word, or long. The size of the immediate data matches

the operation size.

Condition Codes:

ပ	0
>	0
Z	*
z	*
×	I

 X — Not affected. N — Set if the most significant bit of the result is set; cleared otherwise. Z — Set if the result is zero; cleared otherwise. V — Always cleared. C — Always cleared. C

Instruction Format:

				_
0		~		
_	SS	REGISTER		
7	ADDRESS	Œ.	_	
	EFFECTIVE		FE DAT	
4	EFF	MODE	8-BIT BYTE DATA	
ç				
9	17.10	4		Ä
_	ā	ō		32-BIT LONG DATA
œ	c	>		2-ВП СС
ກ		>		3
2	0		ΤA	
Ξ	c	>	ORD DAT	
12	c	>	16-BIT WC	
73	c	>	14	
4	c	>		
15	c	-		

OR N

Inclusive-OR (M68000 Family)

OR

Instruction Fields:

Size field—Specifies the size of the operation.

00— Byte operation 01— Word operation

10—Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Register
uQ	000	reg. number:Dn	W.(xxx)	111	000
An	ı	ı	L(xxx).L	111	001
(An)	010	reg. number:An	# <data></data>	I	I
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	1	1
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	ı	1

MC68020, MC68030, and MC68040 only

I	I	-
I	ı	I
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

^{*}Can be used with CPU32.

Immediate field—Data immediately following the instruction. If size = 00, the data is the low-order byte of the immediate word. If size = 01, the data is the entire immediate word.

If size = 10, the data is the next two immediate words.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

4-154

MOTOROLA

Integer Instructions

Inclusive-OR Immediate to Condition Codes

ORI to CCR

(M68000 Family)

Source V CCR → CCR

ORI # < data > ,CCR Assembler

Operation:

Syntax:

Size = (Byte) Attributes: **Description:** Performs an inclusive-OR operation on the immediate operand and the condition codes and stores the result in the condition code register (low-order byte of the status register). All implemented bits of the condition code register are affected.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×	*	

X — Set if bit 4 of immediate operand is one; unchanged otherwise.
N — Set if bit 3 of immediate operand is one; unchanged otherwise.
Z — Set if bit 2 of immediate operand is one; unchanged otherwise.
V — Set if bit 1 of immediate operand is one; unchanged otherwise.
C — Set if bit 0 of immediate operand is one; unchanged otherwise.

Instruction Format:

0	0	
-	0	
7	1	
က	1	TE DATA
4	1	-BIT BY
2	1	
9	0	
_	0	
∞	0	0
6	0	0
9	0	0
=	0	0
12	0	0
13	0	0
4	0	0
12	0	0

PEA

ORI to SR Inclusive-OR Immediate to the Status Register (M68000 Family)

If Supervisor State Operation:

Then Source V SR → SR

Else TRAP

ORI # < data > ,SR Assembler Syntax:

Size = (Word) Attributes: **Description:** Performs an inclusive-OR operation of the immediate operand and the status register's contents and stores the result in the status register. All implemented bits of the status register are affected.

Condition Codes:

ပ	*
>	*
Z	*
z	*
×	*

X—Set if bit 4 of immediate operand is one; unchanged otherwise.

N—Set if bit 3 of immediate operand is one; unchanged otherwise.

Z—Set if bit 2 of immediate operand is one; unchanged otherwise.

V—Set if bit 1 of immediate operand is one; unchanged otherwise.

C—Set if bit 0 of immediate operand is one; unchanged otherwise.

Instruction Format:

0	0	
~	0	
7	-	
က	-	
4	-	
2	-	
9	-	¥
7	0	ORD DA
œ	0	—BIT W
6	0	16
10	0	
7	0	
12	0	
13	0	
14	0	
15	0	

Push Effective Address (M68000 Family) PEA

 $SP-4 \rightarrow SP$; < ea > \rightarrow (SP) Operation:

Assembler

PEA < ea > Syntax: Size = (Long) Attributes:

Description: Computes the effective address and pushes it onto the stack. The effective

address is a long address.

Condition Codes:

Not affected.

Instruction Format:

0		~		
-	SS	EGISTEF		
2	ADDRE	2		
က	FFECTIVE			
4	Ш	MODE		
2				
9	,	-		
7	0			
∞	c			
0	c	0		
10	c	>		
7	,	-		
12	c			
13	c			
4	,	-		
15				

Instruction Field:

Effective Address field—Specifies the address to be pushed onto the stack. Only control addressing modes can be used as listed in the following tables:

Register	000	001	I			010	011
Mode	111	111	ı			111	111
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	ı	reg. number:An	ı	ı	reg. number:An	reg. number:An
Mode	I	I	010	I	I	101	110
Addressing Mode Mode	D	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

111	111	111
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

011

110 011

MOTOROLA

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Can be used with CPU32.

Supervisor (Privileged) Instructions

RESET

RESET

Reset External Devices (M68000 Family)

Operation:

If Supervisor State
Then Assert RESET (RSTO, MC68040 Only) Line

Else TRAP

Assembler

RESET Syntax: Unsized Attributes: **Description:** Asserts the RSTO signal for 512 (124 for MC68000, MC68EC000, MC68HC000, MC68HC001, MC68008, MC68010, and MC68302) clock periods, resetting all external devices. The processor state, other than the program counter, is unaffected, and execution continues with the next instruction.

Condition Codes:

Not affected.

Instruction Format:

nteger Instructions

Rotate (Without Extend) (M68000 Family) ROL, ROR

ROL, ROR

Destination Rotated By < count > → Destination Operation:

Assembler

ROd Dx,Dy ROd < ea > where d is direction, L or R ROd # < data > ,Dy ROd < ea > where d is direction, L or R Syntax:

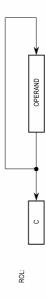
Size = (Byte, Word, Long) Attributes:

Description: Rotates the bits of the operand in the direction specified (L or R). The extend bit is not included in the rotation. The rotate count for the rotation of a register is specified in either of two ways:

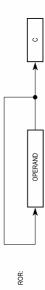
Immediate—The rotate count (1 – 8) is specified in the instruction.

Register—The rotate count is the value in the data register specified in the instruction, modulo 64.

The size of the operation for register destinations is specified as byte, word, or long. The contents of memory, (ROd < ea >), can be rotated one bit only, and operand size is restricted to a word. The ROL instruction rotates the bits of the operand to the left; the rotate count determines the number of bit positions rotated. Bits rotated out of the high-order bit go to the carry bit and also back into the low-order bit.



The ROR instruction rotates the bits of the operand to the right; the rotate count determines the number of bit positions rotated. Bits rotated out of the low-order bit go to the carry bit and also back into the high-order bit.



MOTOROLA

ROL, ROR

Rotate (Without Extend) (M68000 Family)

ROL, ROR

Condition Codes:

	_
ပ	*
>	0
Z	*
z	*
×	Ι

X — Not affected.
N — Set if the most significant bit of the result is set; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Always cleared.
C — Set according to the last bit rotated out of the operand; cleared when the rotate

Instruction Format:

REGISTER ROTATE

0	٧
-	EGISTER
7	R
က	1
4	1
2	i/r
9	SIZE
7	ZIS
80	dr
6	~
10	COUNT/ REGISTER
Ξ	Œ
12	0
13	-
4	-
15	-

Instruction Fields:

Count/Register field:

If *l/r* = 0, this field contains the rotate count. The values 1 – 7 represent counts of 1 – 7, and zero specifies a count of eight.

If i/r = 1, this field specifies a data register that contains the rotate count (modulo 64)

dr field—Specifies the direction of the rotate.

0 — Rotate right

1 — Rotate left

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation

10 — Long operation

i/r field—Specifies the rotate count location. If i/r = 0, immediate rotate count.

If i/r = 1, register rotate count.

Register field—Specifies a data register to be rotated.

Integer Instructions

ROL, ROR

Rotate (Without Extend) (M68000 Family)

ROL, ROR

Instruction Format:

	0		0
	-	EFFECTIVE ADDRESS	DECICTED
	7		_
	က		
	4		
' ROTATE	2		
	9	-	
	7	-	
JEMORY ROT	80	ъ	
≝	6	1 1	
	10		
	£		
	12	0	
	13	-	
	4	,	-
	15	,	-

Instruction Fields:

dr field—Specifies the direction of the rotate.

0 — Rotate right

1 — Rotate left

Effective Address field—Specifies the operand to be rotated. Only memory alterable addressing modes can be used as listed in the following tables:

Register	000	1001					I
Mode	=======================================	#	ı			I	I
Addressing Mode Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
_		_					
Register	I	-	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	ı	ı	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

1	-	-	
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	reg. number:An	
110	110	110	
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)	

|

4-161

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

^{*}Can be used with CPU32.

ROXL, ROXR

ROXL, ROXR Rotate with Extend (M68000 Family)

Destination Rotated With X By Count → Destination Operation:

Assembler Syntax:

ROXd # < data > ,Dy ROXd < ea >

where d is direction, L or R

Size = (Byte, Word, Long) Attributes:

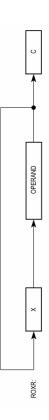
Description: Rotates the bits of the operand in the direction specified (L or R). The extend bit is included in the rotation. The rotate count for the rotation of a register is specified in either of two ways:

- Immediate—The rotate count (1 8) is specified in the instruction.
- 2. Register—The rotate count is the value in the data register specified in the instruction, modulo 64.

rotate count determines the number of bit positions rotated. Bits rotated out of the high-order bit go to the carry bit and the extend bit; the previous value of the extend bit rotates into the low-order bit. The size of the operation for register destinations is specified as byte, word, or long. The contents of memory, < ea >, can be rotated one bit only, and operand size is restricted to a word. The ROXL instruction rotates the bits of the operand to the left; the



The ROXR instruction rotates the bits of the operand to the right; the rotate count determines the number of bit positions rotated. Bits rotated out of the low-order bit go to the carry bit and the extend bit; the previous value of the extend bit rotates into the high-



M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

4-163

ROXL, ROXR

Rotate with Extend (M68000 Family)

ROXL, ROXR

Condition Codes:

*

-Set to the value of the last bit rotated out of the operand; unaffected when the rotate count is zero.

N — Set if the most signilicant with the most signilicant with the result is zero; cleared otherwise.

V — Always cleared.

C — Set according to the last bit rotated out of the operand; when the rotate count is zero, set to the value of the extend bit.

Instruction Format:

	0	~
	-	EGISTER
REGISTER ROTATE	7	2
	က	0
	4	1
	2	i/r
	9	Щ
	7	SIZE
	80	₽
REG	6	~
	10	COUNT/ REGISTER
	7	~
	12	0
	13	-
	4	-
	15	1

Instruction Fields:

Count/Register field:

If i/r = 0, this field contains the rotate count. The values 1 – 7 represent counts of 1

If i/r = 1, this field specifies a data register that contains the rotate count (modulo 64) -7, and zero specifies a count of eight.

dr field—Specifies the direction of the rotate.

0 — Rotate right 1 — Rotate left

Rotate with Extend (M68000 Family) ROXL, ROXR

ROXL, ROXR

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

i/r field—Specifies the rotate count location. If i/r = 0, immediate rotate count.

If i/r = 1, register rotate count.

Register field—Specifies a data register to be rotated.

Instruction Format:

	0		œ
/ ROTATE	-	ECTIVE ADDRESS	EGISTER
	7		œ
	က		
	4	EFFE	MODE
	2		
	9	-	
	7	-	
MOR	œ	ъ	
≝	6	0	
	10	-	
	7	0	
	12	0	
	13	-	
	4	,	-
	15	,	-

Instruction Fields:

dr field—Specifies the direction of the rotate.

0 — Rotate right

1 — Rotate left

Effective Address field—Specifies the operand to be rotated. Only memory alterable addressing modes can be used as listed in the following tables:

Register	000	100	ı			I	1
Mode	111	111	I			I	I
Addressing Mode Mode	W.(xxx)	J.(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	ı	I	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Mode Register	1	1	010 reg. number:An	011 reg. number:An	100 reg. number:An	101 reg. number:An	110 reg. number:An

MC68020, MC68030, and MC68040 only

I	I	I
I	ı	I
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
	_	<u>-</u>
reg. number:Ar	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	[bd,An,Xn],od)	[bd,An],Xn,od)

*Can be used with CPU32.

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

Supervisor (Privileged) Instructions

RTE

Operation:

Return from Exception (M68000 Family)

RTE

If Supervisor State Then (SP) \rightarrow SR; SP + 2 \rightarrow SP; (SP) \rightarrow PC; SP + 4 \rightarrow SP; Restore State and Deallocate Stack According to (SP)

Else TRAP

Assembler

RTE Syntax:

Unsized Attributes: **Description:** Loads the processor state information stored in the exception stack frame located at the top of the stack into the processor. The instruction examines the stack format field in the format/offset word to determine how much information must be restored.

Condition Codes:

Set according to the condition code bits in the status register value restored from the

Instruction Format:

0	1	
_	1	
7	0	
က	0	
4	1	
2	1	
9	1	
7	0	
80	0	
6	1	
10	1	
7	1	
12	0	
13	0	
4	1	
15	0	

Format/Offset Word (in Stack Frame):

MC68010, MC68020, MC68030, MC68040, CPU32

		ı
0		
-		
7		
က		
4	OFFSET	
2	VECTOR O	
9		
7		
80		
6		
10	0	
7	0	
12		
13	MAT	
4	FOR	
15		

Format Field of Format/Offset Word:

Contains the format code, which implies the stack frame size (including the format/ offset word). For further information, refer to Appendix B Exception Processing Reference.

6-84

Return and Restore Condition Codes (M68000 Family) RTR

RTR

(SP) \rightarrow CCR; SP + 2 \rightarrow SP; (SP) \rightarrow PC; SP + 4 \rightarrow SP Operation:

Assembler Syntax:

RTR

Unsized Attributes: **Description:** Pulls the condition code and program counter values from the stack. The previous condition code and program counter values are lost. The supervisor portion of the status register is unaffected.

Condition Codes:

Set to the condition codes from the stack.

Instruction Format:

Integer Instructions

RTS

Return from Subroutine (M68000 Family) **RTS**

 $(SP) \rightarrow PC$; $SP + 4 \rightarrow SP$ Operation:

RTS Assembler Syntax:

Unsized Attributes: Description: Pulls the program counter value from the stack. The previous program counter

value is lost.

Condition Codes:

Not affected.

Instruction Format:

0	1
-	0
2	-
က	0
4	1
2	-
9	-
7	0
8	0
6	-
10	-
1	-
12	0
13	0
4	-
15	0

4-168

Subtract Decimal with Extend (M68000 Family) SBCD

SBCD

Destination10 – Source10 – X → Destination Operation:

SBCD Dx,Dy SBCD – (Ax), – (Ay) Assembler Syntax:

Size = (Byte) Attributes:

Description: Subtracts the source operand and the extend bit from the destination operand and stores the result in the destination location. The subtraction is performed using binary-coded decimal arithmetic; the operands are packed binary-coded decimal

numbers. The instruction has two modes:

1. Data register to data register—the data registers specified in the instruction contain the operands.

2. Memory to memory—the address registers specified in the instruction access the operands from memory using the predecrement addressing mode

This operation is a byte operation only.

Condition Codes

ပ	*	
>	Π	
Z	*	
z	n	
×	*	

- Set the same as the carry bit.
- Cleared if the result is nonzero; unchanged otherwise.
- Undefined.
 Set if a borrow (decimal) is generated; cleared otherwise. X — Set the same
 N — Undefined.
 Z — Cleared if the
 V — Undefined.
 C — Set if a borro

Normally the Z condition code bit is set via programming before the start of an operation. This allows successful tests for zero results upon completion of multiple-precision operations.

Integer Instructions

Subtract Decimal with Extend (M68000 Family) SBCD

SBCD

Instruction Format:

Register Dy/Ay field—Specifies the destination register.

Instruction Fields:

If R/M = 0, specifies a data register. If R/M = 1, specifies an address register for the predecrement addressing mode.

R/M field—Specifies the operand addressing mode.

0 — The operation is data register to data register.

1 — The operation is memory to memory.

Register Dx/Ax field—Specifies the source register.

If R/M = 0, specifies a data register.

If R/M = 1, specifies an address register for the predecrement addressing mode.

4-170

Set According to Condition (M68000 Family) Scc

Scc

Set According to Condition (M68000 Family)

REGISTER

MODE

- _

CONDITION 10 9

0 13

0 15

Instruction Fields:

7

12

4

Instruction Format:

Scc

Scc

EFFECTIVE ADDRESS

7

Integer Instructions

If Condition True Then 1s → Destination Operation:

Else 0s → Destination

Assembler

Scc < ea > Syntax: Size = (Byte) Attributes:

specified by the effective address to TRUE (all ones). Otherwise, sets that byte to FALSE (all zeros). Condition code cc specifies one of the following conditional tests (refer to Table 3-19 for more information on these conditional tests): Description: Tests the specified condition code; if the condition is true, sets the byte

Mnemonic	Condition	Mnemonic	Condition
CC(HI)	Carry Clear	ST	Low or Same
CS(LO)	Carry Set	בו	Less Than
EQ	Equal	M	Minus
ш	False	NE	Not Equal
GE	Greater or Equal	J.	Plus
GT	Greater Than	⊥	True
Ξ	High	ΛC	Overflow Clear
LE	Less or Equal	ΝS	Overflow Set

Condition Codes:

Not affected.

Register	Addressing Mode Mode	Mode	Register	
reg. number:Dn	W.(xxx)	111	000	
I	L(xxx).L	111	100	_
reg. number:An	# <data></data>	1	ı	
reg. number:An				
reg. number:An				
reg. number:An	(d ₁₆ ,PC)	_	_	
reg. number:An	(d ₈ ,PC,Xn)	I	1	

000

5 ٩u

Register

Addressing Mode Mode

following tables:

010 011 100

(An)

(An) +

|

101

(d₁₆,An) – (An)

110

(d₈,An,Xn)

Effective Address field—Specifies the location in which the TRUE/FALSE byte is to be stored. Only data alterable addressing modes can be used as listed in the

Condition field—The binary code for one of the conditions listed in the table.

MC68020, MC68030, and MC68040 only

	1	
([bd,PC,Xn],od)	([bd,PC],Xn,od)	
reg. number:An	reg. number:An	
110	110	
([bd,An,Xn],od)	([bd,An],Xn,od)	
	110 reg. number:An ([bd,PC,Xn],od)	110 reg. number.An ([bd,PC,Xn],od)

I

NOTE

address can be used to change the Scc result from TRUE or FALSE to the equivalent arithmetic value (TRUE = 1, FALSE = 0). In the MC68000 and MC68008, a memory destination is read before it is written. A subsequent NEG.B instruction with the same effective

MOTOROLA

^{*}Can be used with CPU32.

Supervisor (Privileged) Instructions

STOP

Load Status Register and Stop (M68000 Family) STOP

Then Immediate Data → SR; STOP If Supervisor State Operation:

Else TRAP

Assembler

STOP # < data > Syntax:

Unsized Attributes:

instruction tracing is enabled (T0 = 1, T1 = 0) when the STOP instruction begins supervisor portions), advances the program counter to point to the next instruction, and stops the fetching and executing of instructions. A trace, interrupt, or reset exception causes the processor to resume instruction execution. A trace exception occurs if execution. If an interrupt request is asserted with a priority higher than the priority level set by the new status register value, an interrupt exception occurs; otherwise, the Description: Moves the immediate operand into the status register (both user and interrupt request is ignored. External reset always initiates reset exception processing.

Condition Codes:

Set according to the immediate operand.

Instruction Format:

0	0	
-	-	
7	0	
က	0	
4	-	
2	-	
9	-	۵
7	0	TE DAT
œ	0	MMFD
6	-	
9	-	
£	-	
12	0	
13	0	
4	-	
15	0	

Instruction Fields:

Immediate field—Specifies the data to be loaded into the status register.

nteger Instructions

Subtract (M68000 Family)

Destination – Source → Destination

Operation:

SUB < ea > ,Dn Assembler Syntax:

SUB Dn < ea >

Size = (Byte, Word, Long) Attributes:

Description: Subtracts the source operand from the destination operand and stores the result in the destination. The size of the operation is specified as byte, word, or long. The mode of the instruction indicates which operand is the source, which is the destination, and which is the operand size.

Condition Codes:

ပ > Ν z ×

X — Set to the value of the carry bit.

N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow is generated; cleared otherwise.
C — Set if a borrow is generated; cleared otherwise.

Instruction Format:

0		œ
_	SS	EGISTEF
7	ADDRES	쮼
က	FECTIVE	
4	出	MODE
2		
9		
7	100	DINION L
ω		
6		
10	111000	
F		Ľ
12	,	-
13		>
4		>
15	,	-

MOTOROLA

4-174

Subtract (M68000 Family)

SUB

Instruction Fields:

Register field—Specifies any of the eight data registers.

Opmode field

< ea > - Dn → < ea > Dn – < ea > → Dn Operation Long 110 010 Word 101 90 **Byte** 000 100 Effective Address field—Determines the addressing mode. If the location specified is a source operand, all addressing modes can be used as listed in the following tables:

Register	000	100	100			010	011
Mode	111	111	111			111	111
Addressing Mode Mode	W.(xxx)	L(xxx)	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
	ڃ	۲	Ą	٦	An	۸n	٦
Register	reg. number:Dn	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Addressing Mode Register	000 reg. number:□	001 reg. number:/	010 reg. number.	011 reg. number.	100 reg. number.	101 reg. number.	110 reg. number.

MC68020, MC68030, and MC68040 only

011	111	([bd,PC],Xn,od)	reg. number:An	110	([bd,An],Xn,od)
011	111	([bd,PC,Xn],od)	reg. number:An	110	([bd,An,Xn],od)
011	111	(bd,PC,Xn)**	reg. number:An	110	(bd,An,Xn)**

*For byte-sized operation, address register direct is not allowed. **Can be used with CPU32.

Integer Instructions

SUB

Subtract (M68000 Family)

SUB

If the location specified is a destination operand, only memory alterable addressing modes can be used as listed in the following tables:

Register	000	001	ı			I	I
Mode	111	111	ı			I	I
Addressing Mode Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	I	ĺ	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	ı	ı	010	011	100	101	110
Addressing Mode Mode	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

(bd,An,Xn)*	110	reg. number:An	<u>a</u>
([bd,An,Xn],od)	110	reg. number:An	pq])
([pq,An],Xn,od)	110	reg. number:An	pq])

I	1	I	
I	I	I	
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)	
Ş	5	ş	

*Can be used with CPU32.

NOTE

If the destination is a data register, it must be specified as a destination Dn address, not as a destination < ea > address.

Most assemblers use SUBA when the destination is an address register and SUBI or SUBQ when the source is immediate data.

MOTOROLA

SUBA

Subtract Address (M68000 Family)

SUBA

Destination – Source → Destination

Assembler Syntax:

Operation:

SUBA < ea > ,An

Size = (Word, Long) Attributes:

Description: Subtracts the source operand from the destination address register and stores the result in the address register. The size of the operation is specified as word or long. Word-sized source operands are sign-extended to 32-bit quantities prior to the subtraction.

Condition Codes:

Not affected.

Instruction Format:

0		œ
_	SS	REGISTE
7	E ADDRE	<u></u>
က	EFFECTIV	
4	H	MODE
2		
9	١,	1
7	ייייייייייייייייייייייייייייייייייייייי	D IN C
00		
6	٥	۷.
9	TOICE	
7	٥	۷
12	,	-
13	-	>
14		>
15	,	-

Instruction Fields:

Register field—Specifies the destination, any of the eight address registers.

Opmode field—Specifies the size of the operation.

011—Word operation. The source operand is sign-extended to a long operand and the operation is performed on the address register using all 32 bits.

111—Long operation.

Integer Instructions

SUBA

Subtract Address (M68000 Family)

SUBA

Effective Address field—Specifies the source operand. All addressing modes can be used as listed in the following tables:

Dogistor	register	000	001	100			010	011
Mode	MOCE	111	111	111			111	111
Address in Made Made	Addlessing Mode	W.(xxx)	(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Г								
Dogiotor	leiseu	reg. number:Dn	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	MODE	000	100	010	011	100	101	110
Mode Mode	anowi Ringe in	Dn	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

011 011

111 11

011

1

4-177

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

MOTOROLA

^{*}Can be used with CPU32.

Subtract Immediate (M68000 Family)

Destination – Immediate Data → Destination

Operation:

SUBI # < data > , < ea > Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

Description: Subtracts the immediate data from the destination operand and stores the result in the destination location. The size of the operation is specified as byte, word,

or long. The size of the immediate data matches the operation size.

Condition Codes:

ပ	*	
>	*	
Z	*	
z	*	
×	*	

X — Set to the value of the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.

C — Set if a borrow occurs; cleared otherwise.

Instruction Format:

0		٣		
-	SS	REGISTER		
2	ADDRESS	2		
က	EFFECTIVE,		E DATA	
4	EFF	MODE	8-BIT BYT	
2			8	
9	Į,	4		A
7	2	ō		LONG DAT
œ		>		32-BIT LO
6	c	>		ĸ
9	,	-	4	
=		>	JRD DAT	
12		>	-BIT WC	
13	-	>	16	
4		>		
15	,	>		

Integer Instructions

SUBI

Subtract Immediate (M68000 Family)

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation

10 — Long operation

Effective Address field—Specifies the destination operand. Only data alterable addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Mode	Regist
Dn	000	reg. number:Dn	W.(xxx)		111	000
An	I	I	L(xxx).L		111	001
(An)	010	reg. number:An	# <data></data>		I	1
(An) +	011	reg. number:An				
– (An)	100	reg. number:An				
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)		ı	I
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	<u>-</u>	ı	I

MC68020, MC68030, and MC68040 only

	ı	1
l	ı	ı
(ba, P.C, An)	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number.An	reg. number:An	reg. number:An
2	110	110
(DG,AN,AN)	([bd,An,Xn],od)	([bd,An],Xn,od)

^{*}Can be used with CPU32.

If size = 00, the data is the low-order byte of the immediate word. mmediate field—Data immediately following the instruction.

If size = 01, the data is the entire immediate word. If size = 10, the data is the next two immediate words.

4-179

M68000 FAMILY PROGRAMMER'S REFERENCE MANUAL

MOTOROLA

SUBQ

Subtract Quick (M68000 Family)

SUBQ

Destination – Immediate Data → Destination Operation:

Assembler Syntax:

SUBQ # < data > , < ea >

Size = (Byte, Word, Long) Attributes:

be used with address registers, and the condition codes are not affected. When **Description:** Subtracts the immediate data (1 - 8) from the destination operand. The size of the operation is specified as byte, word, or long. Only word and long operations can subtracting from address registers, the entire destination address register is used, despite the operation size.

Condition Codes:

ပ	*
>	*
Z	*
z	*
×	*

X — Set to the value of the carry bit.
N — Set if the result is negative; cleared otherwise.
Z — Set if the result is zero; cleared otherwise.
V — Set if an overflow occurs; cleared otherwise.
C — Set if a borrow occurs; cleared otherwise.

Instruction Format:

0		œ
-	SS	REGISTER
2	E ADDRE	<u></u>
в	FECTIVE	
4	Н	MODE
2		
9	2212	<u> </u>
7	ō	ō
80	,	-
6		
10	Y L	<u> </u>
1		
12	,	-
13	c	>
4	-	-
15	c	>

Integer Instructions

SUBQ

Subtract Quick (M68000 Family)

SUBQ

Instruction Fields:

Data field—Three bits of immediate data; 1 – 7 represent immediate values of 1 – 7, and zero represents eight.

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation

10 — Long operation

Effective Address field—Specifies the destination location. Only alterable addressing modes can be used as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Regist
Dn	000	reg. number:Dn	W.(xxx)	11	000
An*	001	reg. number:An	J.(xxx)	111	001
(An)	010	reg. number:An	# <data></data>	ı	I
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)	I	I
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)	ı	I

MC68020, MC68030, and MC68040 only

I	-	I
I	_	_
(bd,PC,Xn)**	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)**	([bd,An,Xn],od)	([bd,An],Xn,od)

*Word and long only.
**Can be used with CPU32.

MOTOROLA

SUBX

Subtract with Extend (M68000 Family)

SUBX

Destination – Source – X → Destination Operation:

SUBX Dx,Dy SUBX – (Ax), – (Ay) Assembler Syntax: Size = (Byte, Word, Long) Attributes:

Description: Subtracts the source operand and the extend bit from the destination operand and stores the result in the destination

location. The instruction has two modes:

- Data register to data register—the data registers specified in the instruction contain the operands.
- 2. Memory to memory—the address registers specified in the instruction access the operands from memory using the predecrement addressing mode

The size of the operand is specified as byte, word, or long.

Condition Codes:

*	
*	
*	
*	
*	
	*

- X Set to the value of the carry bit.
 N Set if the result is negative; cleared otherwise.
 Z Cleared if the result is nonzero; unchanged otherwise.
 V Set if an overflow occurs; cleared otherwise.
 C Set if a borrow occurs; cleared otherwise.

Normally the Z condition code bit is set via programming before the start of an operation. This allows successful tests for zero results upon completion of multiple-precision operations.

nteger Instructions

SUBX

Subtract with Extend (M68000 Family)

SUBX

Instruction

(Moodoo railing)		
	on Format:	

13

nstruction Fields:

6

10

Register Dy/Ay field—Specifies the destination register.

If R/M = 0, specifies a data register. If R/M = 1, specifies an address register for the predecrement addressing mode.

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

R/M field—Specifies the operand addressing mode. 0 — The operation is data register to data register.

1 — The operation is memory to memory.

Register Dx/Ax field—Specifies the source register:

If R/M = 0, specifies a data register.

If R/M = 1, specifies an address register for the predecrement addressing mode.

MOTOROLA

SWAP

Swap Register Halves (M68000 Family)

SWAP

Register $31 - 16 \longleftrightarrow$ Register 15 - 0

SWAP Dn Assembler Syntax:

Operation:

Size = (Word) Attributes: **Description:** Exchange the 16-bit words (halves) of a data register.

Condition Codes:

z ×

X — Not affected. Not affected. Not affected. Not affected. Not affected. Set if the most significant bit of the 32-bit result is set; cleared otherwise. Z — Set if the 32-bit result is zero; cleared otherwise. Not — Always cleared. Con Always cleared.

Instruction Format:

REGISTER

Instruction Field:

Register field—Specifies the data register to swap.

Integer Instructions

TAS

Test and Set an Operand (M68000 Family)

TAS

Destination Tested → Condition Codes; 1 → Bit 7 of Destination Operation:

Assembler

TAS < ea > Syntax: Size = (Byte) Attributes:

appropriately. TAS also sets the high-order bit of the operand. The operation uses a locked or read-modify-write transfer sequence. This instruction supports use of a flag Description: Tests and sets the byte operand addressed by the effective address field. The instruction tests the current value of the operand and sets the N and Z condition bits

or semaphore to coordinate several processors.

Condition Codes:

* z

 X — Not affected. N — Set if the most significant bit of the operand is currently set; cleared otherwise. Z — Set if the operand was zero; cleared otherwise. V — Always cleared. C — Always cleared.

Instruction Format:

0		R
-	ESS	REGISTE
2	EADDR	_
3	FECTIV	
4	Ш	MODE
2		
9	-	
7	-	
80	0	
6	,	-
10	0	
1	-	
12	0	
13	0	
14	,	-
15	٥	>

MOTOROLA

4-186

TAS

Test and Set an Operand (M68000 Family)

TAS

Instruction Fields:

Effective Address field—Specifies the location of the tested operand. Only data alterable addressing modes can be used as listed in the following tables:

Register	000	100	I			1	I
Mode	11	111	ı			ı	-
Addressing Mode Mode	W.(xxx)	L(xxx).L	# <data></data>			(d ₁₆ ,PC)	(d ₈ ,PC,Xn)
Register	reg. number:Dn	ı	reg. number:An	reg. number:An	reg. number:An	reg. number:An	reg. number:An
Mode	000	ı	010	011	100	101	110
Idressing Mode Mode	Ъ	An	(An)	(An) +	– (An)	(d ₁₆ ,An)	(d ₈ ,An,Xn)

MC68020, MC68030, and MC68040 only

ı	I	I
I		I
(bd,PC,Xn)*	([bd,PC,Xn],od)	([bd,PC],Xn,od)
reg. number:An	reg. number:An	reg. number:An
110	110	110
(bd,An,Xn)*	([bd,An,Xn],od)	([bd,An],Xn,od)

*Can be used with CPU32.

Integer Instructions

TRAP

Trap (M68000 Family)

TRAP

Operation:

1 → S-Bit of SR *SSP – 2 → SSP; Format/Offset → (SSP); SSP – 4 → SSP; PC → (SSP); SSP – 2 → SSP; SR → (SSP); Vector Address → PC

*The MC68000 and MC68008 do not write vector offset or format code to the system stack.

Assembler Syntax:

TRAP # < vector >

Unsized Attributes: **Description:** Causes a TRAP # < vector > exception. The instruction adds the immediate operand (vector) of the instruction to 32 to obtain the vector number. The range of vector values is 0 – 15, which provides 16 vectors.

Condition Codes:

Not affected.

Instruction Format:

0	
-	TOR
7	VEC
က	
4	0
2	0
9	-
7	0
80	0
6	-
10	-
£	-
12	0
13	0
4	-
15	0

Instruction Fields:

Vector field—Specifies the trap vector to be taken.

MOTOROLA

4-188

TRAPV

Trap on Overflow (M68000 Family)

TRAPV

If V Then TRAP Operation:

Assembler Syntax:

TRAPV

Unsized Attributes:

number 7. If the overflow condition is not set, the processor performs no operation and execution continues with the next instruction. Description: If the overflow condition is set, causes a TRAPV exception with a vector

Condition Codes:

Not affected.

Instruction Format:

Integer Instructions

TST

Test an Operand (M68000 Family)

TST

Destination Tested → Condition Codes Operation:

TST < ea > Assembler Syntax:

Size = (Byte, Word, Long) Attributes:

Description: Compares the operand with zero and sets the condition codes according to

the results of the test. The size of the operation is specified as byte, word, or long.

Condition Codes:

ပ	0
>	0
Z	*
z	*
×	I

X — Not affected.
N — Set if the operand is negative; cleared otherwise.
Z — Set if the operand is zero; cleared otherwise.
V — Always cleared.
C — Always cleared.

Instruction Format:

0		H.
_	SS	EGISTE
2	EADDRE	<u>~</u>
က	FECTIVE	
4	EF	MODE
2		
9	121	ш
7	č	7
00	c	>
6	,	-
10	c	>
=		-
12	c	>
13	o	>
4		-
15		>

MOTOROLA

TST

Test an Operand (M68000 Family)

TST

Instruction Fields:

Size field—Specifies the size of the operation.

00 — Byte operation 01 — Word operation 10 — Long operation

Effective Address field—Specifies the addressing mode for the destination operand as listed in the following tables:

Addressing Mode Mode	Mode	Register	Addressing Mode Mode	Mode	Register
Du	000	reg. number:Dn	(xxx).W	111	000
An*	001	reg. number:An	L(xxx).L	111	100
(An)	010	reg. number:An	# <data>*</data>	111	100
(An) +	011	reg. number:An			
– (An)	100	reg. number:An			
(d ₁₆ ,An)	101	reg. number:An	(d ₁₆ ,PC)**	111	010
(d ₈ ,An,Xn)	110	reg. number:An	(d ₈ ,PC,Xn)**	111	011

MC68020, MC68030, and MC68040 only

bd,An,Xn],od)	110	reg. number.An	([bd,PC,Xn]*** ([bd,PC,Xn],od)	+	110
, 20, 12, [17,	2	ieg. Ildiliber.All	(pd,r Cj,r,cd)	=	-

MC68020, MC68030, MC68040, and CPU32. Address register direct allowed only for word

Integer Instructions

UNLK

Unlink (M68000 Family)

UNLK

An \rightarrow SP; (SP) \rightarrow An; SP + 4 \rightarrow SP Operation:

UNLK An Assembler Syntax:

Unsized Attributes: **Description:** Loads the stack pointer from the specified address register, then loads the address register with the long word pulled from the top of the stack.

Condition Codes:

Not affected.

Instruction Format:

REGISTER

Instruction Field:

Register field—Specifies the address register for the instruction.

MOTOROLA

4-193

and long.
**PC relative addressing modes do not apply to MC680000, MC6800008, or MC68010.
***Can be used with CPU32.

Addressing Capabilities

2.4 BRIEF EXTENSION WORD FORMAT COMPATIBILITY

that allows the CPU32, MC68020, MC68030, and MC68040 to distinguish the basic M68000 family architecture's new address extensions. Figure 2-3 illustrates these brief extension SCALE is the same encoding for both extension words. Software that uses this encoding is compatible direction, with downward support only for nonscaled addressing. If the MC68000 gnored and would not access the desired memory address. The earlier microprocessors do Programs can be easily transported from one member of the M68000 family to another in an upward-compatible fashion. The user object code of each early member of the family, microprocessor without change. Brief extension word formats are encoded with information word formats. The encoding for SCALE used by the CPU32, MC68020, MC68030, and MC68040 is a compatible extension of the M68000 family architecture. A value of zero for compatible with all processors in the M68000 family. Both brief extension word formats do not contain the other values of SCALE. Software can be easily migrated in an upwardwere to execute an instruction that encoded a scaling factor, the scaling factor would be not recognize the brief extension word formats implemented by newer processors. Although they can detect illegal instructions, they do not decode invalid encodings of the brief which is upward compatible with newer members, can be executed on the newer extension word formats as exceptions.

0		
-		
7	EGER	
က	ENT INT	
4	ACEM	
2	DISPL	
9		
7		
00	0	
6	0	
10	0	
Ξ	T///	
12	R	
13	EGISTE	
4	æ	
15	D/A	

(a) MC68000, MC68008, and MC68010

	_
0	
_	
2	GER
က	NT INTE
4	LACEME
2	DISPI
9	
7	
œ	0
6	CALE
10	SC/
7	M/L
12	~
13	EGISTE
14	ď
15	D/A

(b) CPU32, MC68020, MC68030, and MC68040

Figure 2-3. M68000 Family Brief Extension Word Formats

Addressing Capabilities

Table 2-1. Instruction Word Format Field Definitions

2	
	Instruction
Mode	Addressing Mode
Register	General Register Number
	Extensions
D/A	Index Register Type 0 = Dn 1 = An
M/L	Word/Long-Word Index Size 0 = Sign-Extended Word 1 = Long Word
Scale	Scale Factor 00 = 1 00 = 2 10 = 4 11 = 8
BS	Base Register Suppress 0 = Base Register Added 1 = Base Register Suppressed
<u>S</u>	Index Suppress 0 = Evaluate and Add Index Operand 1 = Suppress Index Operand
BD SIZE	Base Displacement Size 00 = Reserved 01 = Null Displacement 10 = Word Displacement 11 = Long Displacement
I/IS	Index/Indirect Selection Indexect and Indexing Operand Determined in Conjunction with Bit 6, Index Suppress

For effective addresses that use a full extension word format, the index suppress (IS) bit and the index/indirect selection (I/IS) field determine the type of indexing and indirect action. Table 2-2 lists the index and indirect operations corresponding to all combinations of IS and I/IS values

MOTOROLA

MOTOROLA

Table B-1. Exception Vector Assignments for the M68000 Family

Vector Number(s)	Vector Offset (Hex)	Assignment
0	000	Reset Initial Interrupt Stack Pointer
1	004	Reset Initial Program Counter
2	800	Access Fault
3	00C	Address Error
4	010	Illegal Instruction
5	014	Integer Divide by Zero
6	018	CHK, CHK2 Instruction
7	01C	FTRAPcc, TRAPcc, TRAPV Instructions
8	020	Privilege Violation
9	024	Trace
10	028	Line 1010 Emulator (Unimplemented A- Line Opcode)
11	02C	Line 1111 Emulator (Unimplemented F-Line Opcode)
12	030	(Unassigned, Reserved)
13	034	Coprocessor Protocol Violation
14	038	Format Error
15	03C	Uninitialized Interrupt
16–23	040-05C	(Unassigned, Reserved)
24	060	Spurious Interrupt
25	064	Level 1 Interrupt Autovector
26	068	Level 2 Interrupt Autovector
27	06C	Level 3 Interrupt Autovector
28	070	Level 4 Interrupt Autovector
29	074	Level 5 Interrupt Autovector
30	078	Level 6 Interrupt Autovector
31	07C	Level 7 Interrupt Autovector
32–47	080-0BC	TRAP #0 D 15 Instruction Vectors
48	0C0	FP Branch or Set on Unordered Condition
49	0C4	FP Inexact Result
50	0C8	FP Divide by Zero
51	0CC	FP Underflow
52	0D0	FP Operand Error
53	0D4	FP Overflow
54	0D8	FP Signaling NAN
55	0DC	FP Unimplemented Data Type (Defined for MC68040)
56	0E0	MMU Configuration Error
57	0E4	MMU Illegal Operation Error
58	0E8	MMU Access Level Violation Error
59–63	0ECD0FC	(Unassigned, Reserved)
64–255	100D3FC	User Defined Vectors (192)



How to reach us: USA/EUROPE/Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 303–675–2140 or 1–800–441–2447

Customer Focus Center: 1-800-521-6274

Mfax™: RMFAX0@email.sps.mot.com - TOUCHTONE 602-244-6609 - US & Canada ONLY 1-800-774-1848 Motorola Fax Back System

- http://sps.motorola.com/mfax/

HOME PAGE: http://motorola.com/sps/

Mfax is a trademark of Motorola, Inc.

JAPAN: Nippon Motorola Ltd.; SPD, Strategic Planning Office, 141, 4–32–1, Nishi–Gotanda, Shagawa-ku, Tokyo, Japan. 03–5487–8488

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298