Notation Used in Instruction Set Summary

CPU Register Notation	Machine Coding
Accumulator A — A or a	dd — 8-bit direct address \$0000 to \$00FF (high byte assumed to be \$00).
Accumulator B — B or b	ee — High-order byte of a 16-bit constant offset for indexed addressing.
Accumulator D — D or d	eb — Exchange/Transfer post-byte.
Index Register X — X or x	ff — Low-order 8 bits of a 9-bit signed constant offset for indexed addr.,
Index Register Y — Y or y	or low-order byte of a 16-bit constant offset for indexed address.
Stack Pointer — SP, sp, or s	hh — High-order byte of a 16-bit extended address.
Program Counter — PC, pc, or p	ii — 8-bit immediate data value.
Condition Code Register — CCR or c	jj — High-order byte of a 16-bit immediate data value.
Condition Code Negister — CCN of C	kk — Low-order byte of a 16-bit immediate data value.
Forter than of the time Forest in Course From Orbital	lb — Loop primitive (DBNE) post-byte.
Explanation of Italic Expressions in Source Form Column	, , , , , , , , , , , , , , , , , , , ,
abc — A or B or CCR	· · · · · · · · · · · · · · · · · · ·
abcdxys — A or B or CCR or D or X or Y or SP	mm — 8-bit immediate mask value for bit manipulation instructions.
abdxys — A or B or D or X or Y or SP	Set bits indicate bits to be affected.
msk8 — 8-bit mask (some assemblers require #msk8)	rr — Signed relative offset \$80 (–128) to \$7F (+127). Offset relative to
opr8i — 8-bit immediate value	the byte following the relative offset byte.
opr16i — 16-bit immediate value	xb — Indexed addressing post-byte.
opr8a — 8-bit address used with direct address mode	
opr16a — 16-bit address value	
oprx0_xysp — Indexed addressing postbyte code:	
oprx3,-xys Predecrement X or Y or SP by 1 8	
oprx3,+xys Preincrement X or Y or SP by 1 8	
oprx3,xys- Postdecrement X or Y or SP by 1 8	
oprx3,xys+ Postincrement X or Y or SP by 1 8	Operators
oprx5,xysp 5-bit constant offset from X or Y or SP or PC	+ — Addition
abd,xysp Acc. A or B or D offset from X or Y or SP or PC	 Subtraction
oprx3 — Any positive integer 1 8 for pre/post increment/decrement	& — Logical AND
oprx5 — Any integer in the range –16 +15	+ — Logical OR (inclusive)
oprx9 — Any integer in the range –256 +255	⊕ Logical exclusive OR
oprx16 — Any integer in the range =30,768 65,535	· · · · · · · · · · · · · · · · · · ·
re/9 — Label of branch destination within –512 to +511 locations	× — Multiplication
xys — X or Y or SP	÷ — Division
•	 M — Negation. One's complement (invert each bit of M)
xysp — X or Y or SP or PC	: — Concatenate
Address Mode Notation	⇒ — Transfer
INH — Inherent; no operands in object code	⇔ — Exchange
IMM — Immediate; operand in object code	Condition Codes Columns
DIR — Direct; operand is the lower byte of an addr. from \$0000 to \$00FF	 Status bit not affected by operation.
EXT — Operand is a 16-bit address	 O — Status bit cleared by operation.
REL — Two's complement relative offset; for branch instructions	 Status bit set by operation.
IDX — Indexed (no extension bytes); includes:	Δ — Status bit affected by operation.
5-bit constant offset from X, Y, SP, or PC	 Stat. bit may be cleared or remain set,
Pre/post increment/decrement by 1 8	but is not set by operation.
Accumulator A, B, or D offset	 ↑ Status bit may be set or remain cleared,
IDX1 — 9-bit signed offset from X, Y, SP, or PC; 1 extension byte	but is not cleared by operation.
IDX2 — 16-bit signed offset from X, Y, SP, or PC; 2 extension bytes	? — Status bit may be changed by operation,
[IDX2] — Indexed-indirect; 16-bit offset from X, Y, SP, or PC	but the final state is not defined
[D, IDX] — Indexed-indirect; accumulator D offset from X, Y, SP, or PC	but the initial state to not defined

Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec. T(clk)	SXHI	NZVC	Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec. T (clk)		NZV
ABA	(A) + (B) ⇒ A	INH	18 06	2	Δ -	ΔΔΔΔ			EXT	78 hh II	4		ΔΔΔΔ
	Add Accumulators A and B						ASL oprx0_xysp		IDX	68 xb	3		
ABX	$(B) + (X) \Rightarrow X$	IDX	1A E5	2			ASL oprx9,xysp	C b7 b0	IDX1	68 xb ff	4		
	Translates to LEAX B,X						ASL oprx16,xysp	Arithmetic Shift Left	IDX2	68 xb ee ff	5		
ABY	$(B) + (Y) \Rightarrow Y$	IDY	19 ED	2			ASL [D,xysp]	7 thannous Shint Est	[D,IDX]	68 xb	6		
	Translates to LEAX B,Y						ASL [oprx16,xysp]		[IDX2]	68 xb ee ff	6		
ADCA #opr8i	(A) + (M) + C ⇒ A	IMM	89 ii	1	A -	ΔΔΔΔ	ASLA	Arithmetic Shift Left Accumulator A	INH	48	1		
ADCA opr8a	Add with Carry to A	DIR	99 dd	3	_		ASLB		INH	58	1		
ADCA opr16a	Add with Carry to A	EXT	B9 hh II	3				Arithmetic Shift Left Accumulator B					
							ASLD	← ←	INH	59	1		- Δ Δ Δ Δ
ADCA oprx0_xysp		IDX	A9 xb	3									
ADCA oprx9_xysp		IDX1	A9 xb ff	3				C b7 A b0 b7 B b0					
ADCA oprx16,xysp		IDX2	A9 xb ee ff	4				Arithmetic Shift Left Double					
ADCA [D,xysp]		[D,IDX]	A9 xb	6			ASR opr16a	Attainede Griff Left Boubie	EXT	77 hh II	4		- Δ Δ Δ Δ
ADCA [oprx16,xysp]		[IDX2]	A9 xb ee ff	6			- · · · · · · · · · · · · · · · · · · ·						
ADCB #opr8i	(B) + (M) + C ⇒ B	IMM	C9 ii	1	A -	ΔΔΔΔ	ASR oprx0_xysp		IDX	67 xb	3		
ADCB opr8a	Add with Carry to B	DIR	D9 dd	3	_		ASR oprx9,xysp	b7 b0 C	IDX1	67 xb ff	4		
ADCB opr16a	Add with Carry to B	EXT	F9 hh II	3			ASR oprx16,xysp	Arithmetic Shift Right	IDX2	67 xb ee ff	5		
			-				ASR [D,xysp]	3	[D,IDX]	67 xb	6		
ADCB oprx0_xysp		IDX	E9 xb	3			ASR [oprx16,xysp]		[IDX2]	67 xb ee ff	6		
ADCB oprx9_xysp		IDX1	E9 xb ff	3			ASRA	Arithmetic Shift Right Accumulator A	INH	47	1		
ADCB oprx16,xysp		IDX2	E9 xb ee ff	4			ASRB	<u> </u>	INH	57	1		
ADCB [D,xysp]		[D,IDX]	E9 xb	6				Arithmetic Shift Right Accumulator B			' "		
ADCB [oprx16,xysp]		[IDX2]	E9 xb ee ff	6			BCC rel8	Branch if Carry Clear (if C = 0)	REL	24 rr	3/11)		
ADDA #opr8i	$(A) + (M) \Rightarrow A$	IMM	8B ii	1	Δ -	ΔΔΔΔ	BCLR opr8a, msk8	$(M) \cdot (\overline{mm}) \Rightarrow M$	DIR	4D dd mm	4		Δ Δ 0 -
ADDA opr8a	Add without Carry to A	DIR	9B dd	3	_		BCLR opr16a, msk8	Clear Bit(s) in Memory	EXT	1D hh II mm	4		
ADDA oproa ADDA opr16a	Add Williout Carry to A	EXT	BB hh II	3			BCLR oprx0 xysp, msk8	(4)	IDX	0D xb mm	4		
							BCLR oprx9,xysp, msk8		IDX1	0D xb ff mm	4		
ADDA oprx0_xysp		IDX	AB xb	3			11		IDX1	0D xb ii iiiiii	6		
ADDA oprx9_xysp		IDX1	AB xb ff	3			BCLR oprx16,xysp, msk8	5 1 1 1 0 0 1 1 1 0 1 1					
ADDA oprx16,xysp		IDX2	AB xb ee ff	4			BCS rel8	Branch if Carry Set (if C = 1)	REL	25 rr	3/11)		
ADDA [D,xysp]		[D,IDX]	AB xb	6			BEQ rel8	Branch if Equal (if Z = 1)	REL	27 rr	3/1 ¹⁾		
ADDA [oprx16,xysp]		[IDX2]	AB xb ee ff	6			BGE rel8	Branch if Greater Than or Equal	REL	2C rr	3/11)		-
ADDB #opr8i	(B) + (M) ⇒ B	IMM	CB ii	1	A -	ΔΔΔΔ	1	(if $N \oplus V = 0$) (signed)					
ADDB opr8a	Add without Carry to B	DIR	DB dd	3			BGT rel8	Branch if Greater Than	REL	2E rr	3/11)		
	Add without Carry to B						BG1 reio		KEL	25 11	3/1		
ADDB opr16a		EXT	FB hh II	3				$(if Z + (N \oplus V) = 0) (signed)$					
ADDB oprx0_xysp		IDX	EB xb	3			BHI rel8	Branch if Higher	REL	22 rr	3/11)		
ADDB oprx9_xysp		IDX1	EB xb ff	3				(if $C + Z = 0$) (unsigned)					
ADDB oprx16,xysp		IDX2	EB xb ee ff	4			BHS rel8	Branch if Higher or Same	REL	24 rr	3/1 ¹⁾		
ADDB [D,xysp]		[D,IDX]	EB xb	6				(if C = 0) (unsigned) same as BCC					
ADDB [oprx16,xysp]		[IDX2]	EB xb ee ff	6			BITA #opr8i	(A) • (M)	IMM	85 ii	1		ΔΔ0-
ADDD #opr16i	(A:B) + (M:M+1) ⇒ A:B	IMM	C3 jj kk	2		ΔΔΔΔ		Logical AND A with Memory	DIR	95 dd	3		A A O -
ADDD apr8a	` ' '		D3 dd					•					
	Add 16-Bit to D (A:B)	DIR		3			BITA opr16a	Does not change Accumulator or	EXT	B5 hh II	3		
ADDD opr16a		EXT	F3 hh II	3			BITA oprx0_xysp	Memory	IDX	A5 xb	3		
ADDD oprx0_xysp		IDX	E3 xb	3			BITA oprx9,xysp		IDX1	A5 xb ff	3		
ADDD oprx9_xysp		IDX1	E3 xb ff	3			BITA oprx16,xysp		IDX2	A5 xb ee ff	4		
ADDD oprx16,xysp		IDX2	E3 xb ee ff	4			BITA [D,xysp]		[D,IDX]	A5 xb	6		
ADDD [D,xysp]		[D,IDX]	E3 xb	6			BITA [oprx16,xysp]		[IDX2]	A5 xb ee ff	6	1	1
ADDD [oprx16,xysp]		[IDX2]	E3 xb ee ff	6			BITB #opr8i	(B) • (M)	IMM	C5 ii	1		ΔΔ0-
ANDA #opr8i	(A) • (M)⇒ A	IMM	84 ii	1	L	Δ Δ 0 -	BITB opr8a	Logical AND B with Memory	DIR	D5 dd	3		2 2 0 -
,	. , . ,	DIR	94 dd					•					1
ANDA opr8a	Logical AND A with Memory			3			BITB opr16a	Does not change Accumulator or	EXT	F5 hh II	3	1	1
ANDA opr16a		EXT	B4 hh II	3			BITB oprx0_xysp	Memory	IDX	E5 xb	3	1	1
ANDA oprx0_xysp		IDX	A4 xb	3			BITB oprx9,xysp		IDX1	E5 xb ff	3	1	1
ANDA oprx9,xysp		IDX1	A4 xb ff	3			BITB oprx16,xysp		IDX2	E5 xb ee ff	4	1	1
ANDA oprx16,xysp		IDX2	A4 xb ee ff	4			BITB [D,xysp]				6	1	1
ANDA [D,xysp]			A4 xb	6			BITB [oprx16,xysp]		[IDX2]	E5 xb ee ff	6	1	1
ANDA [oprx16,xysp]		[IDX2	A4 xb ee ff	6			BLE rel8	Branch if Less Than or Equal	REL	2F rr	3/11)		
ANDB #opr8i	(B) • (M)⇒ B	IMM	C4 ii	1		Δ Δ 0 -	1 2 10,0	·	INEL	[-1 11	3/1		
	. , . ,					<u> </u>	D. C. 10	$(if Z + (N \oplus V) = 1) (signed)$	n=:	100	0111	-	+
ANDB opr8a	Logical AND B with Memory	DIR	D4 dd	3			BLO rel8	Branch if Lower	REL	25 rr	3/11)		
ANDB opr16a		EXT	F4 hh II	3				(if C = 1) (unsigned) same as BCS					1
ANDB oprx0_xysp		IDX	E4 xb	3			BLS rel8	Branch if Lower or Same	REL	25 rr	3/11)		-
ANDB oprx9,xysp		IDX1	E4 xb ff	3				(if $C + Z = 1$) (unsigned)		1		1	1
ANDB oprx16,xysp		IDX2	E4 xb ee ff	4			BLT rel8	Branch if Less Than	REL	2D rr	3/11)		
ANDB [D,xysp]		[D,IDX]		6					INEL		3/1		
ANDB [oprx16,xysp]		[IDX2	E4 xb ee ff	6			II	(if $N \oplus V = 1$) (signed)		+		 	+
	(CCD) • (M) • CCD					11 11 11 11	BMI rel8	Branch if Minus (if N = 1)	REL	2B rr			
ANDCC #opr8i	(CCR) • (M)⇒ CCR Logical AND CCR with Memory	IMM	10 ii	1	11 11 11	4 4 4 4	BNE rel8	Branch if Not Equal (if Z = 0)	REL	26 rr	3/1 ¹⁾	<u> </u>	<u> </u>
												_	_

Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec. T (clk)		INZVC	Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec T (clk		INZVC
BPL rel8	Branch if Plus (if N = 0)	REL	2A rr	3/1 ¹⁾			CPD #opr16i	(A:B) – (M:M+1)	IMM	8C jj kk	2	*	- Δ Δ Δ Δ
BRA rel8	Branch Always (if 1 = 1)	REL	20 rr	3			CPD opr8a	Compare D to Memory (16-Bit)	DIR	9C dd	3		
BRCLR opr8a, msk8, rel8	Branch if (M) • (mm) = 0	DIR	4F dd mm rr	4			. CPD opr16a		EXT	BC hh II	3		
BRCLR opr16a, msk8, rel8	(if All Selected Bit(s) Clear)	EXT	1F hh II mm rr	5			CPD oprx0_xysp		IDX	AC xb	3		
BRCLR oprx0_xysp, msk8, rel8		IDX	0F xb mm rr	4			CPD oprx9,xysp		IDX1	AC xb ff	3		
BRCLR oprx9,xysp, msk8, rel8		IDX1	0F xb ff mm rr	5			CPD oprx16,xysp		IDX2	AC xb ee ff	4		
BRCLR oprx16,xysp, msk8, rel8		IDX2	0F xb ee ff mm rr	6			CPD [D,xysp]		[D,IDX]	AC xb	6		
BRSET opr8, msk8, rel8	Branch if $(\overline{M}) \cdot (mm) = 0$	DIR	4E dd mm rr	4			. CPD [oprx16,xysp] CPS #opr16i	(CD) (M-M-A)	[IDX2]	AC xb ee ff	2		- Δ Δ Δ Δ
BRSET opr16a, msk8, rel8	(if All Selected Bit(s) Set)	EXT	1E hh II mm rr	5			CPS #0pr16r	(SP) – (M:M+1) Compare SP to Memory (16-Bit)	IMM DIR	8F jj kk 9F dd	3		
BRSET oprx0_xysp, msk8, rel8		IDX	0E xb mm rr	4 5			CPS opr16a	Compare SP to Memory (16-Bit)	EXT	BF hh II	3		
BRSET oprx9,xysp, msk8, rel8 BRSET oprx16,xysp, msk8, rel8	,	IDX1 IDX2	0E xb ff mm rr 0E xb ee ff mm rr	6			CPS oprx0_xysp		IDX	AF xb	3		
BSET opr8, msk8	(M) + (mm)⇒ M	DIR	4C dd mm	4		Δ Δ 0 –	CPS oprx9,xysp		IDX1	AF xb ff	3		
BSET opr16a, msk8	Set Bit(s) in Memory	EXT	1C hh II mm	4			CPS oprx16,xysp		IDX2	AF xb ee ff	4		
BSET oprx0_xysp, msk8	Set Bit(s) in Memory	IDX	0C xb mm	4			CPS [D,xysp]		[D,IDX]	AF xb	6		
BSET oprx9,xysp, msk8		IDX1	0C xb ff mm	4			CPS [oprx16,xysp]		[IDX2]	AF xb ee ff	6		
BSET oprx16,xysp, msk8		IDX1	0C xb ee ff mm	6			CPX #opr16i	(X) – (M:M+1)	IMM	8E jj kk	2		-ΔΔΔΔ
BSR rel8	(SP) – 2⇒ SP;	REL	07 rr	4			CPX opr8a	Compare X to Memory (16-Bit)	DIR	9E dd	3		
Borriero	$(SF) = 2 \Rightarrow SF$, RTNH:RTNL \Rightarrow M(SP):M(SP+1);	INEL	07 11	_			CPX opr16a	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	EXT	BE hh II	3		
	Subroutine address⇒ PC						CPX oprx0_xysp		IDX	AE xb	3		
	Branch to Subroutine						CPX oprx9,xysp		IDX1	AE xb ff	3		
BVC rel8	Branch if Overflow Bit Clr (V=0)	REL	28 rr	3/1 ¹⁾			CPX oprx16,xysp		IDX2	AE xb ee ff	4		
BVS rel8	Branch if Overflow Bit Set (V=1)	REL	29 rr	3/1 ¹⁾			CPX [D,xysp]		[D,IDX]	AE xb	6		
CBA	(A) – (B)	INH	18 17	2		ΔΔΔΔ	CPX [oprx16,xysp]		[IDX2]	AE xb ee f	6		
CBA	Compare 8-Bit Accumulators	IINI	10 17				CPY #opr16i	(Y) – (M:M+1)	IMM	8D jj kk	2		-ΔΔΔΔ
CLC	0⇒ C	IMM	10 FE	1		0	CPY opr8a	Compare X to Memory (16-Bit)	DIR	9D dd	3		
CEC	Translates to ANDCC #\$FE	IIVIIVI	1012	'		0	CPY opr16a		EXT	BD hh II	3		
CLI	0⇒ I	IMM	10 EF	1	1	0	CPY oprx0_xysp		IDX	AD xb	3		
CLI	Translates to ANDCC #\$EF	IIVIIVI	10 L1	'		ŭ	CPY oprx9,xysp		IDX1	AD xb ff	3		
	(enables I-bit interrupts)						CPY oprx16,xysp		IDX2	AD xb ee ff	4		
CLR opr16a	0⇒ M Clear Memory Location	EXT	79 hh II	3		0 1 0 0	CPY [D,xysp]		[D,IDX]	AD xb	6		
CLR oprx0_xysp	04 W Olcar Wellory Location	IDX	69 xb	2		0 1 0 0	CPY [oprx16,xysp]		[IDX2]	AD xb ee ff	6		
CLR oprx9,xysp		IDX1	69 xb ff	3			DAA	Adjust Sum to BCD (Decimal Adj. Acc.A)	inh	18 07	3		-ΔΔ?Δ
CLR oprx16,xysp		IDX2	69 xb ee ff	3			DBEQ abdxys, rel9	(cntr) – 1⇒cntr	REL	04 lb rr	3		
CLR [D,xysp]		[D,IDX]	69 xb	4				if (cntr) = 0, then Branch;	(9-bit)				
CLR [oprx16,xysp]		[IDX2]	69 xb ee ff	4				else Continue to next instruction					
CLRA	0⇒ A Clear Accumulator A	INH	87	1				Decrement Counter and Branch if = 0					
CLRB	0⇒ B Clear Accumulator B	INH	C7	1			DD115 / / / /	(cntr = A, B, D, X, Y or SP)	551	04 lb rr	_		
CLV	0⇒ V Translates to ANDCC #\$FD	IMM	10 FD	1		0-	DBNE abdxys, rel9	(cntr) – 1⇒ cntr	REL	04 10 11	3		
CMPA #opr8i	(A) – (M)	IMM	81 ii	1		ΔΔΔΔ		if (cntr) not = 0, then Branch;	(9-bit)				
CMPA opr8a	Compare Accumulator A with Memory	DIR	91 dd	3				else Continue to next instruction Decrement Counter and Branch if ≠ 0					
CMPA opr16a		EXT	B1 hh II	3				(cntr = A, B, D, X, Y or SP)					
CMPA oprx0_xysp		IDX	A1 xb	3			DEC opr16a	(M) – \$01⇒ M	EXT	73 hh II	4		- Δ Δ Δ -
CMPA oprx9,xysp		IDX1	A1 xb ff	3			DEC opritoa DEC oprx0 xysp	Decrement Memory Location	IDX	63 xb	3		-\D \D \D \D -
CMPA oprx16,xysp		IDX2	A1 xb ee ff	4			DEC oprx0_xysp DEC oprx9.xysp	Decrement Memory Location	IDX1	63 xb ff	4		
CMPA [D,xysp]		[D,IDX]	A1 xb	6			DEC oprx16,xysp		IDX1	63 xb ee ff	5		
CMPA [oprx16,xysp]		[IDX2]	A1 xb ee ff	6			DEC ID xvsnl		[D,IDX]		6		
CMPB #opr8i	(B) – (M)	IMM	C1 ii	1		Δ Δ Δ Δ	DEC [oprx16,xysp]		[IDX2]	63 xb ee ff	6		
CMPB opr8a	Compare Accumulator B with Memory	DIR	D1 dd	3			DECA	(A) – \$01⇒ A Decrement A	INH	43	1		
CMPB opr16a		EXT	F1 hh II	3			DECB	(B) – \$01⇒ B Decrement B	INH	53	1		
CMPB oprx0_xysp		IDX	E1 xb	3			DES	(SP) – \$0001⇒ SP	IDX	1B 9F	2		
CMPB oprx9,xysp		IDX1	E1 xb ff	3				Translates to LEAS –1,SP	.5,		-		
CMPB oprx16,xysp		IDX2	E1 xb ee ff	4			DEX	(X) – \$0001⇒ X	INH	09	1		Δ
CMPB [D,xysp]		[D,IDX]	E1 xb	6				Decrement Index Register X					
CMPB [oprx16,xysp]	<u></u>	[IDX2]	E1 xb ee ff	6	}		DEY	(Y) – \$0001⇒ Y Decrement Ind. Reg. Y	INH	03	1	 	Δ
COM opr16a COM oprx0_xysp	$(\overline{M}) \Rightarrow M$ equivalent to $FF - (M) \Rightarrow M$	EXT	71 hh II	4		Δ Δ 0 1	EORA #opr8i	(A) ⊕ (M)⇒ A	IMM	88 ii	1	 	-ΔΔ0-
II I IIVI ODEVII VIJED	1's Complement Memory Location	IDX	61 xb	3			EORA opr8a	Exclusive-OR A with Memory	DIR	98 dd	3	1	
	•	IDX1	61 xb ff	4	1		EORA opr16a	2.Sidorro Orter Mari Worldy	EXT	B8 hh II	3		1
COM oprx9,xysp		IDVO	Cd ub as ff	_									1
COM oprx9,xysp COM oprx16,xysp		IDX2	61 xb ee ff	5			1						
COM oprx9,xysp COM oprx16,xysp COM [D,xysp]		[D,IDX]	61 xb	6			EORA oprx0_xysp		IDX	A8 xb	3		
COM oprx9,xysp COM oprx16,xysp COM [D,xysp] COM [oprx16,xysp]	$(\overline{\mathbf{A}})\Rightarrow\mathbf{A}$ Complement Acc. A	[D,IDX] [IDX2]	61 xb 61 xb ee ff				EORA oprx0_xysp EORA oprx9,xysp		IDX IDX1				
COM oprx9,xysp COM oprx16,xysp COM [D,xysp]	$(\overline{A})\Rightarrow A$ Complement Acc. A $(\overline{B})\Rightarrow B$ Complement Acc. B	[D,IDX]	61 xb	6			EORA oprx0_xysp		IDX	A8 xb A8 xb ff A8 xb ee ff	3		

Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec. T(clk)		NZVC	Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec.		NZVC
EORB #opr8i	$(B) \oplus (M) \Rightarrow B$	IMM	C8 ii	1		Δ Δ Ο -	LDAB #opr8i	(M)⇒ B	IMM	C6 ii	1		ΔΔ0-
EORB opr8a	Exclusive-OR B with Memory	DIR	D8 dd	3			LDAB opr8a	Load Accumulator B	DIR	D6 dd	3		
EORB opr16a		EXT	F8 hh II	3			LDAB opr16a		EXT	F6 hh II	3		
EORB oprx0_xysp		IDX	E8 xb	3			LDAB oprx0_xysp		IDX	E6 xb	3		
EORB oprx9,xysp		IDX1	E8 xb ff	3			LDAB oprx9,xysp		IDX1	E6 xb ff	3		
EORB oprx16,xysp		IDX2	E8 xb ee ff	4			LDAB oprx16,xysp		IDX2	E6 xb ee ff	4		
EORB [D,xysp]		[D,IDX]	E8 xb	6			LDAB [D,xysp]		[D,IDX]	E6 xb	6		
EORB [oprx16,xysp]		[IDX2]	E8 xb ee ff	6			LDAB [oprx16,xysp]		[IDX2]	E6 xb ee ff	6		
EXG abcdxys,abcdxys	(r1)⇔ (r2) (if r1 and r2 same size) or	INH	B7 eb	1			LDD #opr16i	(M:M+1)⇒ A:B	IMM	CC jj kk	2		ΔΔ0-
	\$00:(r1)⇒ r2 (if r1=8-bit; r2=16-bit) or						LDD opr8a	Load Double Accumulator D (A:B)	DIR	DC dd	3		
	$(r1_{low}) \Leftrightarrow (r2)$ (if r1=16-bit; r2=8-bit)						LDD opr16a	,	EXT	FC hh II	3		
	r1 and r2 may be						LDD oprx0_xysp		IDX	EC xb	3		
	A, B, CCR, D, X, Y, or SP						LDD oprx9,xysp		IDX1	EC xb ff	3		
FDIV	$(D) \div (X) \Rightarrow X$; Remainder $\Rightarrow D$	INH	18 11	12		- Δ Δ Δ	LDD oprx16,xysp		IDX2	EC xb ee ff	4		
I DIV	16 by 16 Bit Fractional Divide	IINIII	10 11	12			LDD [D,xysp]		[D,IDX]	EC xb	6		
IDEO abduus valo	- i - '	REL	04 lb rr	3			LDD [oprx16,xysp]		[IDX2]	EC xb ee ff	6		
IBEQ abdxys, rel9	(cntr) + 1⇒ cntr		04 lb rr	3			LDS #opr16i	(M:M+1)⇒ SP	IMM	CF jj kk	2		ΔΔ0-
	If (cntr) = 0, then Branch;	(9-bit)					LDS opr8a	Load Stack Pointer	DIR	DF dd	3		Δ Δ 0 –
	else Continue to next instruction						LDS opr16a	Load Stack Pointer	EXT	FF hh II	3		
	Increment Counter and Branch if = 0						'						
	(cntr = A, B, D, X, Y, or SP)						LDS oprx0_xysp		IDX	EF xb	3		
IBNE abdxys, rel9	(cntr) + 1⇒ cntr	REL	04 lb rr	3			LDS oprx9,xysp		IDX1	EF xb ff	3		
-	if (cntr) not = 0, then Branch;	(9-bit)					LDS oprx16,xysp		IDX2	EF xb ee ff	4		
	else Continue to next instruction	, ,					LDS [D,xysp]		[D,IDX]		6		
	Increment Counter and Branch if ≠ 0						LDS [oprx16,xysp]		[IDX2]	EF xb ee ff	6		
							LDX #opr16i	(M:M+1)⇒ X	IMM	CE jj kk	2		Δ Δ 0 -
10.11	(cntr = A, B, D, X, Y, or SP)		18 10	10			LDX opr8a	Load Index Register X	DIR	DE dd	3		
IDIV	(D) ÷ (X)⇒ X; Remainder⇒ D	INH	10 10	12		- Δ0 Δ	LDX opr16a		EXT	FE hh II	3		
	16 by 16 Bit Integer Divide (unsigned)						LDX oprx0_xysp		IDX	EE xb	3		
INC opr16a	(M) + \$01⇒ M	EXT	72 hh II	4		Δ Δ Δ –	LDX oprx9,xysp		IDX1	EE xb ff	3		
INC oprx0_xysp	Increment Memory Byte	IDX	62 xb	3			LDX oprx16,xysp		IDX2	EE xb ee ff	4		
INC oprx9,xysp		IDX1	62 xb ff	4			LDX [D,xysp]		[D,IDX]	EE xb	6		
INC oprx16,xysp		IDX2	62 xb ee ff	5			LDX [oprx16,xysp]		[IDX2]	EE xb ee ff	6		
INC [D,xysp]		[D,IDX]	62 xb	6			LDY #opr16i	(M:M+1)⇒ Y	IMM	CD jj kk	2		ΔΔ0-
INC [oprx16,xysp]		[IDX2]	62 xb ee ff	6			LDY opr8a	Load Index Register Y	DIR	DD dd	3		440
INCA	(A) + \$01⇒ A Increment Acc. A	INH	42	1			LDY opr16a	Load Macx Register 1	EXT	FD hh II	3		
INCB	(B) + \$01⇒ B Increment Acc. B	INH	52	1			LDY oprx0_xysp		IDX	ED xb	3		
INS	(SP) + \$0001⇒ SP	INH	1B 81	2			LDY oprx9,xysp		IDX1	ED xb ff	3		
	Translates to LEAS 1,SP			_			LDY oprx16,xysp		IDX1	ED xb ee ff	4		
INX	(X) + \$0001⇒ X	INH	08	1		- Δ	II				1		
	Increment Index Register X		00				LDY [D,xysp]		[D,IDX]	ED xb	6		
INY	(Y) + \$0001 ⇒ Y	INH	02	1			LDY [oprx16,xysp]		[IDX2]	ED xb ee ff			
IIN T	. , .	IINI	02			- Δ ₁	LEAS oprx0_xysp	Effective Address⇒ SP	IDX	1B xb	2		
1145	Increment Index Register Y	EV.	0011 !!				LEAS oprx9,xysp	Load Effective Address into SP	IDX1	1B xb ff	2		
JMP opr16a	Routine address⇒ PC	EXT	06 hh II	3			LEAS oprx16,xysp		IDX2	1B xb ee ff	2		
JMP oprx0_xysp		IDX	05 xb	3			LEAX oprx0_xysp	Effective Address⇒ X	IDX	1A xb	2		
JMP oprx9,xysp	Jump	IDX1	05 xb ff	3			LEAX oprx9,xysp	Load Effective Address into X	IDX1	1A xb ff	2		
JMP oprx16,xysp		IDX2	05 xb ee ff	4			LEAX oprx16,xysp		IDX2	1A xb ee ff	2		
JMP [D,xysp]		[D,IDX]	05 xb	6			LEAY oprx0_xysp	Effective Address⇒ Y	IDX	19 xb	2		
JMP [oprx16,xysp]		[IDX2]	05 xb ee ff	6			LEAY oprx9,xysp	Load Effective Address into Y	IDX1	19 xb ff	2		
JSR opr8a	(SP) – 2⇒ SP;	DIR	17 dd	4			LEAY oprx16.xysp		IDX2	19 xb ee ff	2		
JSR opr16a	RTNH:RTNL⇒ M(SP):M(SP+1);	EXT	16 hh II	4			LSL opr16a	4—	EXT	78 hh II	4		ΔΔΔΔ
JSR oprx0_xysp	Subroutine address⇒ PC	IDX	15 xb	4			LSL oprx0 xysp		IDX	68 xb	3		
JSR oprx9,xysp		IDX1	15 xb ff	4			LSL oprx9,xysp	C b7 b0	IDX1	68 xb ff	4		
JSR oprx16,xysp	Jump to Subroutine	IDX2	15 xb ee ff	5			LSL oprx16,xysp		IDX1	68 xb ee ff	5		
JSR [D,xysp]		[D,IDX]		6				Logical Shift Left		68 xb	6		
JSR [oprx16,xysp]		[IDX2]	15 xb ee ff	6	1		LSL [D,xysp] LSL [oprx16,xysp]	same function as ASL	[D,IDX] [IDX2]	68 xb ee ff	6		
LDAA #opr8i	(M)⇒ A	IMM	86 ii	1	L	Δ Δ 0 –		Logical Shift Accumulator A to Left			0		
LDAA #opr8a	(M)⇒ A Load Accumulator A	DIR	96 dd	3		12 20 -	LSLA	Logical Shift Accumulator A to Left	INH	48	1		
	Loau Accumulator A			3			LSLB	•	INH	58	1	1	ļ
LDAA opr16a		EXT	B6 hh II	-			LSLD		INH	59	1		ΔΔΔΔ
LDAA oprx0_xysp		IDX	A6 xb	3							1		
LDAA oprx9,xysp		IDX1	A6 xb ff	3				C b7 A b0 b7 B b0			1	1	
LDAA oprx16,xysp		IDX2	A6 xb ee ff	4				Logical Shift Left D Accumulator			1		
LDAA [D,xysp]			A6 xb	6				Same function as ASLD			1		
LDAA [oprx16,xysp]	i	[IDX2]	A6 xb ee ff	6	1	1	ļ		1	1	1	1	

Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec.		NZV	Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec. T(clk)	SXHI	NZVC
LSR opr16a LSR oprx0_xysp		EXT IDX	74 hh II 64 xb	4 3		- 0 Δ Δ Δ	PSHX	$(SP) - 2 \Rightarrow SP; (XH:XL) \Rightarrow M_{(SP)}:M_{(SP+1)}$ Push Index Register X onto Stack	INH	34	2		
LSR oprx9,xysp	b7 b0 C	IDX1	64 xb ff	4			PSHY	$(SP) - 2 \Rightarrow SP; (Y_H:Y_L) \Rightarrow M_{(SP)}:M_{(SP+1)}$	INH	35	2		
LSR oprx16,xysp LSR [D,xysp]	Logical Shift Right	IDX2 [D,IDX]	64 xb ee ff 64 xb	5 6			PULA	Push Index Register Y onto Stack $(M_{(SP)}) \Rightarrow A; (SP) + 1 \Rightarrow SP$	INH	32	3		
LSR [oprx16,xysp] LSRA	Logical Shift Accumulator A to Right	[IDX2] INH	64 xb ee ff 44	6				Pull Accumulator A from Stack					
LSRB	Logical Shift Accumulator B to Right	INH	54	1			PULB	(M _(SP))⇒ B; (SP) + 1⇒ SP Pull Accumulator B from Stack	INH	33	3		
LSRD		INH	49	1		- 0 Δ Δ Δ	PULC	(M _(SP))⇒ CCR; (SP) + 1⇒ SP Pull CCR from Stack	INH	38	3	ΔΨΔΔ	ΔΔΔΔ
	b7 A b0 b7 B b0 C Logical Shift Right D Accumulator						PULD	$(M_{(SP)}:M_{(SP+1)})\Rightarrow A:B; (SP) + 2\Rightarrow SP$ Pull D from Stack	INH	3A	3		
MOVB #opr8, opr16a ²⁾ MOVB #opr8i, oprx0_xysp ²⁾	$(M_1) \Rightarrow M_2$ Memory to Memory Byte-Move (8-Bit)		18 0B ii hh II 18 08 xb ii	4			PULX	$(M_{(SP)}:M_{(SP+1)})\Rightarrow X_H:X_L; (SP) + 2\Rightarrow SP$ Pull Index Register X from Stack	INH	30	3		
MOVB opr16a, opr16a ²⁾ MOVB opr16a, oprx0_xysp ²⁾			18 0C hh II hh II 18 09 xb hh II	6 5			PULY	(M _(SP) :M _(SP+1))⇒ Y _H :Y _L ; (SP) + 2⇒ SP Pull Index Register Y from Stack	INH	31	3		
MOVB oprx0_xysp, opr16a2)		IDX-EXT	18 0D xb hh ll	5			ROL opr16a	Tull index register 1 from Stack	EXT	75 hh II	4		ΔΔΔΔ
MOVB oprx0_xysp,oprx0_xysp ²⁾		IDX-IDX	18 0A xb xb	5			ROL oprx0_xysp		IDX	65 xb	3		
MOVW #oprx16, opr16a ²⁾	$(M:M+1_1) \Rightarrow M:M+1_2$	IMM-EXT	18 03 jj kk hh ll	5			ROL oprx9,xysp	' ' ' ' ' ' ' ' '	IDX1	65 xb ff	4		
MOVW #opr16i, oprx0_xysp ²⁾	Memory to Memory Word-Move (16-Bit)		18 00 xb jj kk	4			ROL oprx16,xysp	C b7 b0	IDX1	65 xb ee ff	5		
MOVW opr16a, opr16a ²⁾		EXT-EXT	18 04 hh II hh II	6			ROL [D,xysp]	Rotate Memory Left through Carry	[D,IDX]	65 xb	6		
MOVW opr16a, oprx0_xysp2)		EXT-IDX	18 01 xb hh ll	5			ROL [oprx16,xysp]		[D,IDX]	65 xb ee ff	6		
MOVW oprx0_xysp, opr16a2)			18 05 xb hh II	5			ROLA	Rotate A Left through Carry	INH	45	1		
MOVW oprx0_xysp,oprx0_xysp2)	IDX-IDX	18 02 xb xb	5			ROLB	Rotate B Left through Carry	INH	55	1		
MUL	$(A) \times (B) \Rightarrow A:B - 8 \times 8 $ Unsigned	INH	12	1		/		Troute B East through early	EXT	76 hh II	4		ΔΔΔΔ
	Multiply						ROR oprx0_xysp		IDX	66 xb	3		
NEG opr16a	$0 - (M) \Rightarrow M \text{ equivalent to } (\overline{M}) + 1 \Rightarrow M$	EXT	70 hh II	4		ΔΔΔΔ	ROR oprx9,xysp		IDX IDX1	66 xb ff	4		
NEG oprx0_xysp	. , , , , , , , , , , , , , , , , , , ,	IDX	60 xb	3			ROR oprx16,xysp	b7 b0 C	IDX1	66 xb ee ff	5		
NEG oprx9,xysp	Two's Complement Negate	IDX1	60 xb ff	4			ROR <i>Oprato,xysp</i> ROR [D, <i>xysp</i>]	Rotate Memory Right through Carry	[D,IDX]	66 xb	6		
NEG oprx16,xysp		IDX2	60 xb ee ff	5			ROR [D,xysp] ROR [oprx16,xysp]		[D,IDX] [IDX2]	66 xb ee ff	6		
NEG [D,xysp]		[D,IDX]	60 xb	6			RORA	Rotate A Right through Carry	INH	46	1		
NEG [oprx16,xysp]		[IDX2]	60 xb ee ff	6				Rotate B Right through Carry		56	1		
NEGA	$0 - (A) \Rightarrow A $ equivalent to $(\overline{A}) + 1 \Rightarrow A$	INH	40	1			RORB RTI	$(M_{(SP)}) \Rightarrow CCR; (SP) + 1 \Rightarrow SP$	INH	0B	8/	ΔΨΔΔ	ΔΔΔΔ
	Negate Accumulator A							$(M_{(SP)}:M_{(SP+1)})\Rightarrow B:A; (SP) + 2 \Rightarrow SP$			10		
NEGB	$0 - (B) \Rightarrow B$ equivalent to $(\overline{B}) + 1 \Rightarrow B$	INH	50	1				$(M_{(SP)}:M_{(SP+1)}) \Rightarrow X_H:X_L; (SP) + 4 \Rightarrow SP$			with		
	Negate Accumulator B							$(M_{(SP)}:M_{(SP+1)}) \Rightarrow PC_H:PC_L; (SP) - 2 \Rightarrow SP$			inter-		
NOP	No Operation	INH	A7	1			_]	$(M_{(SP)}:M_{(SP+1)}) \Rightarrow Y_H:Y_L; (SP) + 4 \Rightarrow SP$			rupt		
ORAA #opr8i	(A) + (M)⇒ A	IMM	8A ii	1		-ΔΔ0-		Return from Interrupt			pen- ding		
ORAA opr8a	Logical OR A with Memory	DIR	9A dd	3			RTS	$(M_{(SP)}:M_{(SP+1)})\Rightarrow PC_H:PC_L;$	INH	3D	5		
ORAA opr16a	,	EXT	BA hh II	3			KIS	$(SP) \cdot V(SP+1) \rightarrow FGH.FGL,$ $(SP) + 2 \Rightarrow SP$	IINI	30	3		
ORAA oprx0_xysp		IDX	AA xb	3				Return from Subroutine					
ORAA oprx9,xysp		IDX1	AA xb ff	3			SBA	(A) – (B)⇒ A Subtract B from A	INH	18 16	2		ΔΔΔΔ
ORAA oprx16,xysp		IDX2	AA xb ee ff	4			SBCA #opr8i				_		
ORAA [D,xysp]		[D,IDX]	AA xb	6			SBCA #opr8i SBCA opr8a	$(A) - (M) - C \Rightarrow A$	IMM	82 ii	1		ΔΔΔΔ
ORAA [oprx16,xysp]		[IDX2	AA xb ee ff	6			SBCA opraa SBCA opr16a	Subtract with Borrow from A	DIR	92 dd	3		
ORAB #opr8i	(B) + (M)⇒ B	IMM	CA ii	1		ΔΔ0-	SBCA opritoa SBCA oprix0 xysp		EXT	B2 hh II	3		
ORAB opr8a	Logical OR B with Memory	DIR	DA dd	3			SBCA oprx0_xysp SBCA oprx9,xysp		IDX IDX1	A2 xb A2 xb ff	3		
ORAB opr16a	,	EXT	FA hh II	3			SBCA oprx9,xysp SBCA oprx16,xysp		IDX1	A2 xb ee ff	4		
ORAB oprx0_xysp		IDX	EA xb	3			SBCA (D,xysp)						
ORAB oprx9,xysp		IDX1	EA xb ff	3					[D,IDX]	A2 xb ee ff	6		
ORAB oprx16,xysp		IDX2	EA xb ee ff	4			SBCA [oprx16,xysp]	(D) (M) O D			6		
ORAB [D,xysp]		[D,IDX]		6			SBCB #opr8i	$(B) - (M) - C \Rightarrow B$	IMM	C2 ii	1		ΔΔΔΔ
ORAB [oprx16,xysp]			EA xb ee ff	6			SBCB opr8a	Subtract with Borrow from B	DIR	D2 dd	3		
ORCC #opr8i	(CCR) + M⇒ CCR	IMM	14 ii	1	1 - 1 1	1 1 1 1 1	SBCB OPIXU_XYSP		EXT IDX	F2 hh II E2 xb	3		
DOLLA	Logical OR CCR with Memory	18	20	_	1	1	SBCB oprx9,xysp		IDX1	E2 xb ff	3		
PSHA	$(SP) - 1 \Rightarrow SP; (A) \Rightarrow M_{(SP)}$	INH	36	2			SBCB oprx16,xysp		IDX2	E2 xb ee ff	4		
	Push Accumulator A onto Stack	<u> </u>		ļ	 		SBCB [D,xysp]		[D,IDX]	E2 xb	6		
PSHB	(SP) – 1⇒ SP; (B)⇒ M _(SP) Push Accumulator B onto Stack	INH	37	2			SBCB [oprx16,xysp]	1 · C	[IDX2]	E2 xb ee ff	6		1
PSHC	(SP) – 1⇒ SP; (CCR)⇒ M _(SP) Push CCR onto Stack	INH	39	2			SEC SEI	1⇒ C Translates to ORCC #\$01 1⇒ I; (inhibit I interrupts)	IMM	14 01 14 10	1	<u></u> 1	<u> </u>
1		 	3B	2	+	+		Translates to ORCC #\$10					1 -
PSHD	$(SP) - 2 \Rightarrow SP; (A:B) \Rightarrow M_{(SP)}:M_{(SP+1)}$	INH					SEV	1⇒ V Translates to ORCC #\$02	IMM	14 02	1		

Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec.		NZV	Source Form	Operation	Addr. Mode	Machine Code (hex)	Exec.		INZVO
STAA opr8a	(A)⇒ M	DIR	5A dd	2		ΔΔ0-	- SUBB #opr8i	(B) – (M)⇒ B	IMM	C0 ii	1		- Δ Δ Δ Δ
STAA opr16a	Store Accumulator A to Memory	EXT	7A hh II	3			SUBB opr8a	Subtract Memory from Accumulator B	DIR	D0 dd	3		
STAA oprx0_xysp	,	IDX	6A xb	2			SUBB opr16a		EXT	F0 hh II	3		
STAA oprx9,xysp		IDX1	6A xb ff	3			SUBB oprx0 xysp		IDX	E0 xb	3		
STAA oprx16.xysp		IDX2	6A xb ee ff	3			SUBB oprx9,xysp		IDX1	E0 xb ff	3		
STAA (D,xysp)		[D,IDX]	6A xb	4			SUBB oprx16,xysp		IDX1	E0 xb ee ff	4		
				4					l l				
STAA [oprx16,xysp]		[IDX2]	6A xb ee ff				SUBB [D,xysp]		[D,IDX]	E0 xb	6		
STAB opr8a	(B)⇒ M	DIR	5B dd	2		Δ Δ 0 -			[IDX2]	E0 xb ee ff	6		
STAB opr16a	Store Accumulator B to Memory	EXT	7B hh II	3			SUBD #opr16i	(D) – (M:M+1)⇒ D	IMM	83 jj kk	2		- Δ Δ Δ Δ
STAB oprx0_xysp		IDX	6B xb	2			SUBD opr8a	Subtract Memory from D (A:B)	DIR	93 dd	3		
STAB oprx9,xysp		IDX1	6B xb ff	3			SUBD opr16a		EXT	B3 hh II	3		
STAB oprx16,xysp		IDX2	6B xb ee ff	3			SUBD oprx0_xysp		IDX	A3 xb	3		
STAB [D,xysp]		[D,IDX]	6B xb	4			SUBD oprx9,xysp		IDX1	A3 xb ff	3		
STAB [oprx16,xysp]		[IDX2]	6B xb ee ff	4			SUBD oprx16,xysp		IDX2	A3 xb ee ff	4		
STD opr8a	(A)⇒ M, (B)⇒ M+1	DIR	5C dd	2		ΔΔ0-	SUBD [D,xysp]		[D,IDX]	A3 xb	6		
•	. , ,			3		Δ Δ 0 -	SUBD [oprx16,xysp]		[IDX2]	A3 xb ee ff	6		
STD opr16a	Store Double Accumulator	EXT	7C hh II					(25) 2 25					
STD oprx0_xysp		IDX	6C xb	2			SWI	(SP) – 2⇒ SP;	INH	3F	9		1
STD oprx9,xysp		IDX1	6C xb ff	3				$RTN_H:RTN_L \Rightarrow M_{(SP)}:M_{(SP+1)};$					
STD oprx16,xysp		IDX2	6C xb ee ff	3				$(SP) - 2 \Rightarrow SP; (Y_H:Y_L) \Rightarrow M_{(SP)}:M_{(SP+1)};$					
STD [D,xysp]		[D,IDX]	6C xb	4				$(SP) - 2 \Rightarrow SP; (X_H:X_L) \Rightarrow M_{(SP)}:M_{(SP+1)};$					
STD [oprx16,xysp]		[IDX2]	6C xb ee ff	4				$(SP) - 2 \Rightarrow SP; (B:A) \Rightarrow M_{(SP)}:M_{(SP+1)};$					
STOP	(SP) – 2⇒ SP:	INH	18 3E	8			.]	$(SP) - 1 \Rightarrow SP; (CCR) \Rightarrow M_{(SP)}$					
o. o.	$RTN_H:RTN_L \Rightarrow M_{(SP)}:M_{(SP+1)};$			(ente-				1⇒ I; (SWI Vector)⇒ PC					
				,				, ,					
	$(SP) - 2 \Rightarrow SP; (Y_H:Y_L) \Rightarrow M_{(SP)}:M_{(SP+1)};$			ring)			TAB	Software Interrupt		10.05	_		
	$(SP) - 2 \Rightarrow SP; (X_H:X_L) \Rightarrow M_{(SP)}:M_{(SP+1)};$			6 (exi-			TAB	(A)⇒ B	INH	18 0E	2		- Δ Δ 0 -
	$(SP) - 2 \Rightarrow SP; (B:A) \Rightarrow M_{(SP)}:M_{(SP+1)};$			ting)				Transfer A to B					
	$(SP) - 1 \Rightarrow SP; (CCR) \Rightarrow M_{(SP)};$			2			TAP	(A)⇒ CCR	INH	B7 02	1	$\Delta \Downarrow \Delta$	ΔΔΔΔΔ
	STOP All Clocks							Translates to TFR A, CCR					
	Registers stacked to allow quicker			(con- tinue)			ТВА	(B)⇒ A	INH	18 0F	2		- Δ Δ 0 -
	recovery by interrupt.			unue)			1.571	Transfer B to A			_		
	If S control bit = 1, the STOP instruction			2 (if			TDEO abduus malo		DEL	0416	2		
	is disabled and acts like a 2-cycle NOP.			disab.))		TBEQ abdxys,rel9	If (cntr) = 0, then Branch;	REL	04 lb rr	3		
CTC0-		DIR	EE 44	-		4 4 0	-	else Continue to next instruction	(9-bit)				
STS opr8a	$(SP_H:SP_L) \Rightarrow M:M+1$		5F dd	2		Δ Δ 0 -	1	Test Counter and Branch if Zero					
STS opr16a	Store Stack Pointer	EXT	7F hh II	3				(cntr = A, B, D, X,Y, or SP)					
STS oprx0_xysp		IDX	6F xb	2			TDNE - to to 10		DEI	0.4.11	_		
STS oprx9,xysp		IDX1	6F xb ff	3			TBNE abdxys,rel9	If (cntr) not = 0, then Branch;	REL	04 lb rr	3		
STS oprx16,xysp		IDX2	6F xb ee ff	3				else Continue to next instruction	(9-bit)				
STS [D,xysp]		[D,IDX]	6F xb	4				Test Counter and Branch if Not Zero					
STS [oprx16,xysp]		[IDX2]	6F xb ee ff	4									
STX opr8a	$(X_H:X_L) \Rightarrow M:M+1$	DIR	5E dd	2		ΔΔ0-	TED about a short as	(cntr = A, B, D, X,Y, or SP)	INIT	D7 - I	_		
STX opr16a	Store Index Register X	EXT	7E hh II	3			TFR abcdxys,abcdxys	(r1)⇒ r2 or	INH	B7 eb	1		
STX oprx0_xysp	Store index register X	IDX	6E xb	2									
				3				Transfer Register to Register					
STX oprx9,xysp		IDX1	6E xb ff					r1,r2 may be A, B, CCR, D, X, Y, or SP					
STX oprx16,xysp		IDX2	6E xb ee ff	3			TPA	(CCR)⇒ A	INH	B7 20	1		
STX [D,xysp]		[D,IDX]	6E xb	4				Translates to TFR CCR ,A		-			
STX [oprx16,xysp]		[IDX2]	6E xb ee ff	4			TST opr16a	(M) – 0	EXT	F7 hh II	3		- Δ Δ Ο Ο
STY opr8a	$(Y_H:Y_L) \Rightarrow M:M+1$	DIR	5D dd	2		ΔΔ0-	- · · · · · · · · · · · · · · · · · · ·	` '	l l				
STY opr16a	Store Index Register Y	EXT	7D hh II	3			TST oprx0_xysp	Test Memory for Zero or Minus	IDX	E7 xb	3	1	
STY oprx0_xysp		IDX	6D xb	2			TST oprx9,xysp	(A) – 0 Test A for Zero or Minus	IDX1	E7 xb ff	3	1	
STY oprx9,xysp		IDX1	6D xb ff	3			TST oprx16,xysp	(B) – 0 Test B for Zero or Minus	IDX2	E7 xb ee ff	4	1	
			6D xb ee ff	3			TST [D,xysp]		[D,IDX]		6	1	
STY oprx16,xysp							TST [oprx16,xysp]		[IDX2]	E7 xb ee ff	6	1	
STY [D,xysp]			6D xb	4			TSTA		INH	97	1	1	
STY [oprx16,xysp]		[IDX2]	6D xb ee ff	4			ТЅТВ		INH	D7	1		
SUBA #opr8i	$(A) - (M) \Rightarrow A$	IMM	80 ii	1		ΔΔΔΔ	TSX	(SP)⇒ X	INH	B7 75	1		
SUBA opr8a	Subtract Memory from Accumulator A	DIR	90 dd	3				Translates to TFR SP,X	1/1/1	5, 10	1 '	1	
SUBA opr16a	· ·	EXT	B0 hh II	3			TOY		16	D7 70	+ -	+	+
SUBA oprx0_xysp		IDX	A0 xb	3			TSY	(SP)⇒ Y	INH	B7 76	1		
SUBA oprx9,xysp		IDX1	A0 xb ff	3				Translates to TFR SP,Y			<u> </u>	<u> </u>	1
		IDX1	A0 xb iii	4			TXS	(X)⇒ SP Translates to TFR X,SP	INH	B7 57	1		<u>-1</u>
	1						TYS	(Y)⇒ SP Translates to TFR Y,SP	INH	B7 67	1		
SUBA oprx16,xysp SUBA [D,xysp] SUBA [oprx16,xysp]		[D,IDX]	A0 xb ee ff	6 6			XGDX	(D)⇔ (X) Translates to EXG D, X	INH	B7 C5	1		