	LIDA ADD SUB STA LIDA ADD SUB STA	ALPHA INCR ONE BETA GAMMA INCR ONE DELTA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
ONTE	·	1	ONTE INODE CONTOURNE
ONE	WORD	1	ONE-WORD CONSTANT ONE-WORD VARIABLES
ALPHÁ	RESW	1	ONE-WORD VARIABLES
BETA	RESW	1	
GAMMA	RESW	1	
DELTA	RESW	1	
INCR	RESW	1	
			(a)
	LDS	INCR	LOAD VALUE OF INCR INTO REGISTER S
	LDS LDA	INCR ALPHA	LOAD VALUE OF INCR INTO REGISTER S LOAD ALPHA INTO REGISTER A
	LDA	ALPHA	LOAD ALPHA INTO REGISTER A
	LDA ADDR	ALPHA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR
	LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A
	LDA ADDR SUB STA LDA ADDR	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR
	LDA ADDR SUB STA LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1
	LDA ADDR SUB STA LDA ADDR	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR
	LDA ADDR SUB STA LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1
	LDA ADDR SUB STA LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1
	LDA ADDR SUB STA LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
ALPH <b>A</b>	LDA ADDR SUB STA LDA ADDR SUB	ALPHA S,A #1 BETA GAMMA S,A	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
BETA	LDA ADDR SUB STA LDA ADDR SUB STA	ALPHA S,A #1 BETA GAMMA S,A #1 DELTA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
BETA	LDA ADDR SUB STA LDA ADDR SUB STA	ALPHA S,A #1 BETA GAMMA S,A #1 DELTA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
BETA	LDA ADDR SUB STA LDA ADDR SUB STA	ALPHA S,A #1 BETA GAMMA S,A #1 DELTA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA
BETA GAMMA	LDA ADDR SUB STA LDA ADDR SUB STA	ALPHA S,A #1 BETA GAMMA S,A #1 DELTA	LOAD ALPHA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN BETA LOAD GAMMA INTO REGISTER A ADD THE VALUE OF INCR SUBTRACT 1 STORE IN DELTA

Figure 1.3 Sample arithmetic operations for (a) SIC and (b) SIC/XE.

(b)

loop will continue in this way until all 11 bytes have been copied from STR1 to STR2. Notice that after the TIX instruction is executed, the value in register X is equal to the number of bytes that have already been copied.

	LDX	ZERO	INI	TIALIZE INDEX REGISTER TO 0
MOVECH	LDCH	STR1,X	LOA	D CHARACTER FROM STR1 INTO REG A
	STCH	STR2,X	STO	RE CHARACTER INTO STR2
	TIX	ELEVEN	ADD	1 TO INDEX, COMPARE RESULT TO 11
	$\mathcal{J}\mathrm{LT}$	MOVECH	LOO	P IF INDEX IS LESS THAN 11
STR1	BYTE	C'TEST S	TRING'	11-BYTE STRING CONSTANT
STR2	RESB	11		11-BYTE VARIABLE
				ONE-WORD CONSTANTS
ZERO	WORD	0		
ELEVEN	WORD	11		
			(8	a)
	LDT	#11	INI	TIALIZE REGISTER T TO 11
	LDT LDX	#11 #0		TIALIZE REGISTER T TO 11 TIALIZE INDEX REGISTER TO 0
MOVECH			INI	
MOVECH	LDX	#0	INI LOA	TIALIZE INDEX REGISTER TO 0
MOVECH	LDX	#0 STR1,X	INI LOA STO	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A
MOVECH	LDX LDCH STCH	#0 STR1,X STR2,X	INI LOA STO ADD	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2
MOVECH	LDX LDCH STCH TIXR	#0 STR1,X STR2,X T	INI LOA STO ADD	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11
MOVECH	LDX LDCH STCH TIXR JLT	#0 STR1,X STR2,X T	INI LOA STO ADD	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11
MOVECH	LDX LDCH STCH TIXR JLT	#0 STR1,X STR2,X T	INI LOA STO ADD	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11
MOVECH STR1	LDX LDCH STCH TIXR JLT	#0 STR1,X STR2,X T MOVECH	INI LOA STO ADD LOO	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11
	LDX LDCH STCH TIXR JLT	#0 STR1,X STR2,X T MOVECH	INI LOA STO ADD LOO	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11 P IF INDEX IS LESS THAN 11
STR1	LDX LDCH STCH TIXR JLT	#0 STR1,X STR2,X T MOVECH	INI LOA STO ADD LOO	TIALIZE INDEX REGISTER TO 0 D CHARACTER FROM STR1 INTO REG A RE CHARACTER INTO STR2 1 TO INDEX, COMPARE RESULT TO 11 P IF INDEX IS LESS THAN 11  11-BYTE STRING CONSTANT 11-BYTE VARIABLE

**Figure 1.4** Sample looping and indexing operations for (a) SIC and (b) SIC/XE.

Figure 1.4(b) shows the same loop as it might be written for SIC/XE. The main difference is that the instruction TIXR is used in place of TIX. TIXR works exactly like TIX, except that the value used for comparison is taken from another register (in this case, register T), not from memory. This makes the loop more efficient, because the value does not have to be fetched from memory each time the loop is executed. Immediate addressing is used to initialize register T to the value 11 and to initialize register X to 0.

Figure 1.5 contains another example of looping and indexing operations. The variables ALPHA, BETA, and GAMMA are arrays of 100 words each. In this case, the task of the loop is to add together the corresponding elements of