3.4.5 *Instruction Set Description*

The instruction map is shown in Figure 3-12 and the complete instruction set is summarized in Table 3-17.

	000	040	080	0C0	100	140	180	1C0	200	240	280	2C0	300	340	380	3C0
0xxx																
4xxx																
8xxx																
Cxxx																
1xxx	RRC	RRC.B	SWPB		RRA	RRA.B	SXT		PUSH	PUSH.B	CALL		RETI			
14xx																
18xx																
1Cxx																
20xx	JNE/JNZ															
24xx	JEQ/JZ															
28xx	JNC															
2Cxx	JC															
30xx	JN															
34xx	JGE															
38xx	JL															
3Cxx	JMP															
4xxx	MOV, MOV.B															
5xxx	ADD, ADD.B															
6xxx	ADDC, ADDC.B															
7xxx	SUBC, SUBC.B															
8xxx	SUB, SUB.B															
9xxx	CMP, CMP.B															
Axxx	DADD, DADD.B															
Bxxx	BIT, BIT.B															
Cxxx	BIC, BIC.B															
Dxxx	BIS, BIS.B															
Exxx	XOR, XOR.B															
Fxxx	AND, AND.B															

Figure 3-12. Core Instruction Map

Table 3-17. MSP430 Instruction Set

Mnemonic		Description		V	N	Z	C
`ADC(.B)` [1]	`dst`	Add C to destination	dst + C → dst	*	*	*	*
`ADD(.B)`	`src,dst`	Add source to destination	src + dst → dst	*	*	*	*
`ADDC(.B)`	`src,dst`	Add source and C to destination	src + dst + C → dst	*	*	*	*
`AND(.B)`	`src,dst`	AND source and destination	src .and. dst → dst	0	*	*	*
`BIC(.B)`	`src,dst`	Clear bits in destination	not.src .and. dst → dst	-	-	-	-
`BIS(.B)`	`src,dst`	Set bits in destination	src .or. dst → dst	-	-	-	-
`BIT(.B)`	`src,dst`	Test bits in destination	src .and. dst	0	*	*	*
`BR` [1]	`dst`	Branch to destination	dst → PC	-	-	-	-
`CALL`	`dst`	Call destination	PC+2 → stack, dst → PC	-	-	-	-
`CLR(.B)` [1]	`dst`	Clear destination	0 → dst	-	-	-	-
`CLRC` [1]		Clear C	0 → C	-	-	-	0
`CLRN` [1]		Clear N	0 → N	-	0	-	-
`CLRZ` [1]		Clear Z	0 → Z	-	-	0	-
`CMP(.B)`	`src,dst`	Compare source and destination	dst - src	*	*	*	*
`DADC(.B)` [1]	`dst`	Add C decimally to destination	dst + C → dst (decimally)	*	*	*	*
`DADD(.B)`	`src,dst`	Add source and C decimally to dst	src + dst + C → dst (decimally)	*	*	*	*
`DEC(.B)` [1]	`dst`	Decrement destination	dst - 1 → dst	*	*	*	*

[1] Emulated Instruction

Table 3-17. MSP430 Instruction Set (continued)

Mnemonic		Description		V	N	Z	C
DECD(.B) [1]	dst	Double-decrement destination	dst - 2 → dst	*	*	*	*
DINT [1]		Disable interrupts	0 → GIE	-	-	-	-
EINT [1]		Enable interrupts	1 → GIE	-	-	-	-
INC(.B) [1]	dst	Increment destination	dst +1 → dst	*	*	*	*
INCD(.B) [1]	dst	Double-increment destination	dst+2 → dst	*	*	*	*
INV(.B) [1]	dst	Invert destination	.not.dst → dst	*	*	*	*
JC/JHS	label	Jump if C set/Jump if higher or same		-	-	-	-
JEQ/JZ	label	Jump if equal/Jump if Z set		-	-	-	-
JGE	label	Jump if greater or equal		-	-	-	-
JL	label	Jump if less		-	-	-	-
JMP	label	Jump	PC + 2 × offset → PC	-	-	-	-
JN	label	Jump if N set		-	-	-	-
JNC/JLO	label	Jump if C not set/Jump if lower		-	-	-	-
JNE/JNZ	label	Jump if not equal/Jump if Z not set		-	-	-	-
MOV(.B)	src,dst	Move source to destination	src → dst	-	-	-	-
NOP [2]		No operation		-	-	-	-
POP(.B) [2]	dst	Pop item from stack to destination	@SP → dst, SP+2 → SP	-	-	-	-
PUSH(.B)	src	Push source onto stack	SP - 2 → SP, src → @SP	-	-	-	-
RET [2]		Return from subroutine	@SP → PC, SP + 2 → SP	-	-	-	-
RETI		Return from interrupt		*	*	*	*
RLA(.B) [2]	dst	Rotate left arithmetically		*	*	*	*
RLC(.B) [2]	dst	Rotate left through C		*	*	*	*
RRA(.B)	dst	Rotate right arithmetically		0	*	*	*
RRC(.B)	dst	Rotate right through C		*	*	*	*
SBC(.B) [2]	dst	Subtract not(C) from destination	dst + 0FFFFh + C → dst	*	*	*	*
SETC [2]		Set C	1 → C	-	-	-	1
SETN [2]		Set N	1 → N	-	1	-	-
SETZ [2]		Set Z	1 → Z	-	-	1	-
SUB(.B)	src,dst	Subtract source from destination	dst + .not.src + 1 → dst	*	*	*	*
SUBC(.B)	src,dst	Subtract source and not(C) from dst	dst + .not.src + C → dst	*	*	*	*
SWPB	dst	Swap bytes		-	-	-	-
SXT	dst	Extend sign		0	*	*	*
TST(.B) [2]	dst	Test destination	dst + 0FFFFh + 1	0	*	*	1
XOR(.B)	src,dst	Exclusive OR source and destination	src .xor. dst → dst	*	*	*	*

[2] Emulated Instruction