4.7.3 Program flow control - Jumps

The jump instruction is formatted as follows:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
opcode			C			10 bit PC offset									

Bit		Description
15-13	opcode	
12-10	C	
9-0	PC offset	PCnew = PCold + 2 + PCoffset × 2

Table 4-18 shows the program flow control (jump) instructions that are not emulated.

Table 4-18. Program flow control (jump) instructions.

Mnemonic	Description
Program flow control instructions	
`JEQ/JZ label`	Jump to label if zero flag is set
`JNE/JNZ label`	Jump to label if zero flag is reset
`JC label`	Jump to label if carry flag is set
`JNC label`	Jump to label if carry flag is reset
`JN label`	Jump to label if negative flag is set
`JGE label`	Jump to label if greater than or equal
`JL label`	Jump to label if less than
`JMP label`	Jump to label unconditionally

4.7.4 Emulated instructions

Table 4-19 gives the different emulated instructions. This table also contains the type of operation and the emulated instruction based on the core instructions.

Table 4-19. Emulated instructions.

Mnemonic	Operation	Emulation	Description
Arithmetic instructions			
`ADC(.B or .W) dst`	dst+C→dst	ADDC(.B or .W) #0,dst	Add carry to destination
`DADC(.B or .W) dst`	dst+C→dst (decimally)	DADD(.B or .W) #0,dst	Decimal add carry to destination
`DEC(.B or .W) dst`	dst-1→dst	SUB(.B or .W) #1,dst	Decrement destination
`DECD(.B or .W) dst`	dst-2→dst	SUB(.B or .W) #2,dst	Decrement destination twice
`INC(.B or .W) dst`	dst+1→dst	ADD(.B or .W) #1,dst	Increment destination
`INCD(.B or .W) dst`	dst+2→dst	ADD(.B or .W) #2,dst	Increment destination twice
`SBC(.B or .W) dst`	dst+0FFFFh+C→dst dst+0FFh→dst	SUBC(.B or .W) #0,dst	Subtract source and borrow /.NOT. carry from dest.
Logical and register control instructions			
`INV(.B or .W) dst`	.NOT.dst→dst	XOR(.B or .W) #0(FF)FFh,dst	Invert bits in destination
`RLA(.B or .W) dst`	C←MSB←MSB-1...LSB+1←LSB←0	ADD(.B or .W) dst,dst	Rotate left arithmetically
`RLC(.B or .W) dst`	C←MSB←MSB-1...LSB+1←LSB←C	ADDC(.B or .W) dst,dst	Rotate left through carry
Data instructions			
`CLR(.B or .W) dst`	0→dst	MOV(.B or .W) #0,dst	Clear destination
`CLRC`	0→C	BIC #1,SR	Clear carry flag
`CLRN`	0→N	BIC #4,SR	Clear negative flag
`CLRZ`	0→Z	BIC #2,SR	Clear zero flag
`POP(.B or .W) dst`	@SP→temp SP+2→SP temp→dst	MOV(.B or .W) @SP+,dst	Pop byte/word from stack to destination
`SETC`	1→C	BIS #1,SR	Set carry flag
`SETN`	1→N	BIS #4,SR	Set negative flag
`SETZ`	1→Z	BIS #2,SR	Set zero flag
`TST(.B or .W) dst`	dst + 0FFFFh + 1 dst + 0FFh + 1	CMP(.B or .W) #0,dst	Test destination
Program flow control			
`BR dst`	dst→PC	MOV dst,PC	Branch to destination
`DINT`	0→GIE	BIC #8,SR	Disable (general) interrupts
`EINT`	1→GIE	BIS #8,SR	Enable (general) interrupts
`NOP`	None	MOV #0,R3	No operation
`RET`	@SP→PC SP+2→SP	MOV @SP+,PC	Return from subroutine

Table 4-20. Conditions for status bits, depending on the emulated instruction result.

	Status bits			
Mnemonic	**V**	**N**	**Z**	**C**
Arithmetic instructions				
ADC(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, null result =0, otherwise	=1, dst from 0FFFFh to 0000 =0, otherwise
DADC(.B or .W) dst	-	=1, MSB=1 =0, otherwise	=1, dst=0 =0, otherwise	=1, dst from 99(99) to 00(00) =0, otherwise
DEC(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, dst contained 1 =0, otherwise	=1, dst contained 0 =0, otherwise
DECD(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, dst contained 2 =0, otherwise	=1, dst contained 0 or 1 =0, otherwise
INC(.B or .W) dst	=1, dst contained 07(FF)h =0, otherwise	=1, negative result =0, if positive	=1, dst contained FF(FF)h =0, otherwise	=1, dst contained FF(FF)h =0, otherwise
INCD(.B or .W) dst	=1, dst contained 07(FFE)h =0, otherwise	=1, negative result =0, if positive	=1, dst contained FF(FE)h =0, otherwise	=1, dst contained FF(FF)h or FF(FE)h =0, otherwise
SBC(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, null result, =0, otherwise	=1, if no borrow =0, otherwise
Logical and register control instructions				
INV(.B or .W) dst	=1, negative initial dst =0, otherwise	=1, negative result =0, if positive	=1, dst contained FF(FF)h =0, otherwise	=1, not zero =0, otherwise
RLA(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, null result, =0, otherwise	Loaded from MSB
RLC(.B or .W) dst	=1, Arithmetic overflow =0, otherwise	=1, negative result =0, if positive	=1, null result, =0, otherwise	Loaded from MSB
Data instructions				
CLR(.B or .W) dst	-	-	-	-
CLRC	-	-	-	=0
CLRN	-	=0	-	-
CLRZ	-	-	=0	-
POP(.B or .W) dst	-	-	-	-
SETC	-	-	-	=1
SETN	-	=1	-	-
SETZ	-	-	=1	-
TST(.B or .W) dst	=0	=1, dst negative =0, otherwise	=1, dst contains zero =0, otherwise	=1
Program flow control				
BR dst	-	-	-	-
DINT	-	-	-	-
EINT	-	-	-	-
NOP	-	-	-	-
RET	-	-	-	-