SECTION 1 Coprocessor Instructions

This chapter describes the generic ColdFire coprocessor instructions. These instructions support optional application specific accelerators such as the CAU (see chapter XX).

cpIDbcbusy Branch Conditionally if CoProcessor Busy

Operation: If CoProcessor is Busy
Then PC + d_{16} → PC

Assembler Syntax: cpIDbcbusy <label>

Attributes: Size = Word

Description: The state of the selected CoProcessor is tested and if it is busy executing a multi-cycle instruction, program execution continues at location (PC) + displacement. The program counter contains the address of the instruction word for the cpIDbcbusy instruction plus two. The displacement is a two's-complement integer that represents the relative distance in bytes from the current program counter to the destination program counter.

Condition Codes: Not affected

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	ID	0	1	1	0	0	0	-	-	-
16-BIT SIGNED DISPLACEMENT															

Instruction Fields:

ID—CoProcessor identifier {0,1}.

16-Bit Displacement field—two's complement integer specifying the number of bytes between the branch instruction and the next instruction to be executed if the condition is met.

The low-order 3 bits of the operation word can contain any value [0-7].

NOTE

If execution of this instruction is attempted and the selected CoProcessor not present (or logically disabled), the processor core responds with a non-supported instruction exception.

cpIDld CoProcessor Load

Operation: Source Operands, Command → CoProcessor

<ea>y → CoProcessor Operand 1

Rx → CoProcessor Operand 2

CMD → CoProcessor Command

Assembler Syntax: cpIDld <ea>y,Rx,ET,CMD

Attributes: Size = Byte, Word, Longword

Description: Move the contents of the two source operands along with the coprocessor specific command to the destination CoProcessor.

Condition Codes: Not affected

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	ID	0	SIZE		MODE			REGISTER		
Rx				ET			CMD								

Instruction Fields:

ID—CoProcessor identifier {0,1}.

SIZE—Size specifier for the source operand defined by <ea>y:

00 - byte operation

01 - word operation

10 - longword operation

Effective Address—Specifies the source operand <ea>y; use only the data addressing modes listed in the following table:

Addressing Mode	Mode	Register		Addressing Mode	Mode	Register
Dy	000	reg. number:Dy		(xxx).W	—	—
Ay	001	reg. number:Ay		(xxx).L	—	—
(Ay)	010	reg. number:Ay		#<data>	—	—
(Ay) +	011	reg. number:Ay				
– (Ay)	100	reg. number:Ay				
(d_{16},Ay)	101	reg. number:Ay		(d_{16},PC)	—	—
(d_8,Ay,Xi)	—	—		(d_8,PC,Xi)	—	—

Rx—Register specifier for a second source operand defined by Rx, where $0 is D0,..., $7 is D7, $8 is A0,..., $F is A7. This operand is always longword in size.

ET—Execution time specifier which defines the number of machine cycles required by the CoProcessor before the *next* instruction can be dispatched to the CoProcessor:

if ET = $0 - $6, then “ET+1” cycles are required by the CoProcessor

if ET = $7, then the execution time is 8 cycles or more, and the ColdFire core uses a coprocessor interface signal to determine the next available issue time

CMD—CoProcessor-defined command. This field must be non-zero since the ColdFire core interprets CMD=0 as the CoProcessor NOP instruction.

NOTE

If execution of this instruction is attempted and the selected CoProcessor not present (or logically disabled), the processor core responds with a non-supported instruction exception.

cpIDnop CoProcessor No Operation

Operation: Stall the Core Until the Next-Available Instruction Dispatch Time

Assembler

Syntax: cpIDnop ET

Attributes: Unsized

Description: No operation occurs. The processor state, other than the program counter, is unaffected. The processor core stalls until the next-available CoProcessor dispatch time, and then continues with the instruction following the **cpIDnop**. This operation is **not** sent to the CoProcessor, and may be useful for instruction scheduling.

Condition Codes: Not affected

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	ID	0	SIZE		MODE			REGISTER		
Rx				ET			0	0	0	0	0	0	0	0	0

Instruction Fields:

ID—CoProcessor identifier {0,1}.

SIZE—This field is not used, and may be programmed to {00, 01, 10}.

Effective Address—This field is not used, and may be programmed to any *supported* value shown in the following table:

Addressing Mode	Mode	Register		Addressing Mode	Mode	Register
Dy	000	reg. number:Dy		(xxx).W	—	—
Ay	001	reg. number:Ay		(xxx).L	—	—
(Ay)	010	reg. number:Ay		#<data>	—	—
(Ay) +	011	reg. number:Ay				
– (Ay)	100	reg. number:Ay				
(d_{16},Ay)	101	reg. number:Ay		(d_{16},PC)	—	—
(d_8,Ay,Xi)	—	—		(d_8,PC,Xi)	—	—

Rx—This field is not used, and may be programmed to any value.

ET—Execution time specifier which defines the number of machine cycles required by the CoProcessor before the *next* instruction can be dispatched to the CoProcessor:

if ET = $0 - $7, then “ET+1” cycles are required by the CoProcessor

NOTE

If execution of this instruction is attempted and the selected CoProcessor not present (or logically disabled), the processor core responds with a non-supported instruction exception.

cpIDst CoProcessor Store

Operation: Source Ry Operand, Command → CoProcessor
CoProcessor Result → Destination

Ry → CoProcessor Operand 2

CMD → CoProcessor Command

Result → Destination

Assembler Syntax: cpIDst Ry,<ea>x,ET,CMD

Attributes: Size = Byte, Word, Longword

Description: Move the contents of the register source operand along with the coprocessor specific command to the destination CoProcessor, and store the result from the CoProcessor in the destination effective address.

Condition Codes: Not affected

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	ID	1	SIZE		MODE			REGISTER		
Ry				ET			CMD								

Instruction Fields:

ID—CoProcessor identifier {0,1}.

SIZE—Size specifier for the destination operand defined by <ea>x:

00 - byte operation

01 - word operation

10 - longword operation

If the destination <ea>x specifies a register operand (Dx, Ax), the SIZE specifier must be longword (10). Stores to memory destinations can be any size.

Effective Address field—Specifies the destination operand <ea>x; use only the data addressing modes listed in the following table:

Addressing Mode	Mode	Register		Addressing Mode	Mode	Register
Dx*	000	reg. number:Dx		(xxx).W	—	—
Ax*	001	reg. number:Ax		(xxx).L	—	—
(Ax)	010	reg. number:Ax		#<data>	—	—
(Ax) +	011	reg. number:Ax				
– (Ax)	100	reg. number:Ax				
(d_{16},Ax)	101	reg. number:Ax		(d_{16},PC)	—	—
(d_8,Ax,Xi)	—	—		(d_8,PC,Xi)	—	—

* Size must be longword (10)

Ry—Register specifier for a second source operand defined by Ry, where $0 is D0, ..., $7 is D7, $8 is A0,..., $F is A7. This operand is always longword in size.

ET—Execution time specifier which defines the number of machine cycles required by the CoProcessor before the result is driven back to the ColdFire core:

if ET = $0 - $6, then “ET” cycles (after the command is received) are required by the CoProcessor

if ET = $7, then the execution time is 8 cycles or more, and the ColdFire core uses the **CoProcIDBusyB** signal to determine when the result is valid

CMD—CoProcessor-defined command. This field must be non-zero since the ColdFire core interprets CMD=0 as the CoProcessor NOP instruction.

NOTE

If execution of this instruction is attempted and the selected CoProcessor not present (or logically disabled), the processor core responds with a non-supported instruction exception.