

4 CRIS Execution Times

4.1 Introduction

Instruction execution times for all CRIS instructions and addressing modes are given below in numbers of CPU cycles. Optimal cache performance (i.e. no cache misses) is assumed.

4.2 Instruction execution times

This section gives the execution times for instructions with the four basic addressing modes Quick immediate, Register, Indirect and Autoincrement. Except for the following seven special cases, the execution time is the same for all instructions with the same addressing mode and data size.

General case:

Addressing mode	Data size	Data alignment	Execution time
Quick immediate	6-bit	N/A	1
Register	Any	N/A	1
Indirect, Auto inc.	Byte	Any	2
Indirect, Auto inc.	Word	Address <1:0> != 3	2
Indirect, Auto inc.	Word	Address <1:0> == 3	3
Indirect, Auto inc.	Dword	Address <1:0> == 0	2
Indirect, Auto inc.	Dword	Address <1:0> != 0	3

Table 4-1 General instruction execution times

Special case 1:

Bcc Instruction

Branch offset size	Execution time
Byte	1
Word	2

Table 4-2 Bcc instruction execution times

Special case 2:

MULS and MULU Instructions

The MULS and MULU instructions require two clock cycles.

Special case 3:

MOVEM Instruction

Data size	Data alignment	Execution time
Dword	Address <1:0> == 0	$n + 1$
Dword	Address <1:0> != 0	$2 * n + 1$

Table 4-3 MOVEM instruction execution times

(Where n is the number of registers moved.)

Special case 4:

PC Operand

One idle bus cycle is added to the execution times given above, if PC is used as the destination operand in any of the following instructions:

ABS	ADD	ADDQ	ADDS	ADDU	AND
ANDQ	ASR	ASRQ	BTSTQ	MOVEM	MOVEQ
MOVE (except from a special register)			MOVS	MOVU	OR
ORQ	POP	SUB	SUBQ	SUBS	SUBU
XOR					

One idle bus cycle is also added for the TEST.m PC instruction.

Special case 5:

Break instruction

The BREAK instruction takes two cycles to execute.

Special case 6:

SBFS Instruction

Data alignment	Execution time
Address <1:0> == 0	5
Address <1:0> != 0	9

Table 4-4 SBFS instruction execution times

Special case 7:

RBF Instruction

The RBF execution time includes the time for the restarted cycle.

Data alignment	Type of restarted cycle	Execution time
Address <1:0> == 0	Instruction fetch	6
Address <1:0> == 0	First cycle of data read or write	7
Address <1:0> == 0	Second cycle of data read or write	8
Address <1:0> != 0	Instruction fetch	10
Address <1:0> != 0	First cycle of data read or write	11
Address <1:0> != 0	Second cycle of data read or write	12

Table 4-5 RBF instruction execution times

Special case 8:

SWAP instruction

The SWAP instruction requires one extra clock cycle if all the options (N, W, B AND H) are used in the same instruction.

4.3 Complex addressing modes execution times

The table below gives the extra execution time required to calculate the effective address in complex addressing modes. The effective address calculation time is added to the Indirect/Autoincrement execution time given in *section 4.2 Instruction execution times* to give the total execution time of the instruction.

Data alignment refers to the alignment of data involved in the effective address calculation.

Addressing mode	Data alignment	Execution time
Indexed	N/A	1
Indexed with assigned	N/A	1
Immediate Byte offset	N/A	1
Indirect Byte offset	any	2
Word offset	address <1:0> != 3	2
Word offset	address <1:0> == 3	3
Dword offset	address <1:0> == 0	2
Dword offset	address <1:0> != 0	3
Immediate Byte offset with assign	N/A	1
Indirect Byte offset with assign	any	2
Word offset with assign	address <1:0> != 3	2
Word offset with assign	address <1:0> == 3	3
Dword offset with assign	address <1:0> == 0	2
Dword offset with assign	address <1:0> != 0	3
Double indirect	address <1:0> == 0	2
Double indirect	address <1:0> != 0	3
Double indirect with autoincrement	address <1:0> == 0	2
Double indirect with autoincrement	address <1:0> != 0	3
Absolute	address <1:0> == 0	2
Absolute	address <1:0> != 0	3

Table 4-3 Complex addressing modes execution times

4.4 Interrupt acknowledge execution time

The interrupt acknowledge sequence, including the interrupt acknowledge cycle and the interrupt vector read following it, requires 2 bus cycles. However, if the interrupt vector number is read from the mode register or externally, a number of wait states is added which increases the length of the CPU cycle.