

致尊敬的顾客

关于产品目录等资料中的旧公司名称

NEC电子公司与株式会社瑞萨科技于2010年4月1日进行业务整合（合并），整合后的新公司暨“瑞萨电子公司”继承两家公司的所有业务。因此，本资料中虽还保留有旧公司名称等标识，但是并不妨碍本资料的有效性，敬请谅解。

瑞萨电子公司网址：<http://www.renesas.com>

2010年4月1日
瑞萨电子公司

【发行】瑞萨电子公司（<http://www.renesas.com>）

【业务咨询】<http://www.renesas.com/inquiry>

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
 - “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
 - “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
 - “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

M16C/60、 M16C/20、 M16C/Tiny 系列

瑞萨 16 位单片机

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
The information described here may contain technical inaccuracies or typographical errors.
Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (<http://www.renesas.com>).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.

注意

本文只是参考译文，前页所载英文具有正式效力。

请遵循安全第一进行电路设计

1. 虽然瑞萨科技尽力提高半导体产品的质量和可靠性，但是半导体产品也可能发生故障。半导体的故障可能导致人身伤害、火灾事故以及财产损害。在电路设计时，请充分考虑安全性，采用合适的如冗余设计、利用非易燃材料以及故障或者事故防止等的安全设计方法。

关于利用本资料时的注意事项

1. 本资料是为了让用户根据用途选择合适的瑞萨科技产品的参考资料，不转让属于瑞萨科技或者第三者所有的知识产权和其它权利的许可。
2. 对于因使用本资料所记载的产品数据、图、表、程序、算法以及其它应用电路的例子而引起的损害或者对第三者的权力的侵犯，瑞萨科技不承担责任。
3. 本资料所记载的产品数据、图、表、程序、算法以及其它所有信息均为本资料发行时的信息，由于改进产品或者其它原因，本资料记载的信息可能变动，恕不另行通知。在购买本资料所记载的产品时，请预先向瑞萨科技或者经授权的瑞萨科技产品经销商确认最新信息。
本资料所记载的信息可能存在技术不准确或者印刷错误。因这些错误而引起的损害、责任问题或者其它损失，瑞萨科技不承担责任。
同时也请通过各种方式注意瑞萨科技公布的信息，包括瑞萨科技半导体网站。
(<http://www.renesas.com>)
4. 在使用本资料所记载部分或者全部数据、图、表、程序以及算法等信息时，在最终做出有关信息和产品是否适用的判断前，务必对作为整个系统的所有信息进行评估。由于本资料所记载的信息而引起的损害、责任问题或者其它损失，瑞萨科技不承担责任。
5. 瑞萨科技的半导体产品不是为在可能和人命相关的环境下使用的设备或者系统而设计和制造的产品。在研讨将本资料所记载的产品用于运输、机动车辆、医疗、航空宇宙用、原子能控制、海底中继器的设备或者系统等特殊用途时，请与瑞萨科技或者经授权的瑞萨产品经销商联系。
6. 未经瑞萨科技的书面许可，不得翻印或者复制全部或者部分资料的内容。
7. 如果本资料所记载的某产品或者技术内容受日本出口管理限制，必须在得到日本政府的有关部门许可后才能出口，并且不准进口到批准目的地国家以外的国家。
禁止违反日本和（或者）目的地国家的出口管理法和法规的任何转卖、挪用或者再出口。
8. 如果需要了解本资料所记载的信息或者产品的详细，请与瑞萨科技联系。

本手册的使用方法

本手册是M16C/60、M16C/20、M16C/Tiny系列的软件手册。能用于具有M16C/60系列CPU内核的所有产品。
在使用本手册时，需要具备电子电路、逻辑电路以及微型计算机的基础知识。

本手册由6章构成，按目的参照的章节如下所示：

○M16C/60、M16C/20、M16C/Tiny系列的概要和特点.....	第1章“概要”
○各寻址方式的操作	第2章“寻址方式”
○指令功能 （语法、操作、功能、能选择的src/dest(label)、标志变化、记述例、相关指令）	第3章“功能”
○指令码、周期数	第4章“指令码/周期数”
○中断	第5章“中断”
○周期数的计算方法	第6章“周期数的计算”

另外，在目录后面有各种页一览表，能按目的查找功能和指令码/周期数的记载页。

○从助记符确认记载页	按英文字母分类的页一览表
○从功能确认记载页	按功能分类的页一览表
○从助记符确认寻址和记载页	按寻址分类的页一览表

在本手册的后面有符号表、词汇表以及索引。

M16C族的有关文献

对M16C族，提供了如下的文献：（注1）

文献种类	记载内容
简易图表	硬件概要
数据图表	硬件概要和电特性
硬件手册	硬件规格（管脚配置、存储器映像、外围功能的说明、电特性、时序）
软件手册	指令（汇编语言）运行的详细内容
应用说明	外围功能的应用例子 参考程序 用于M16C族入门的基本功能说明 根据汇编语言和C语言的编程方法
RENESAS TECHNICAL UPDATE	有关产品的规格、文献等的速报

注1. 请使用最新版本。最新版本刊登在瑞萨科技的主页上。

目 录

第 1 章 概要

1.1	M16C/60、M16C/20、M16C/Tiny系列的特点	2
1.1.1	M16C/60、M16C/20、M16C/Tiny 系列的特点	2
1.1.2	速度性能	2
1.2	地址空间	3
1.3	寄存器构成	4
1.3.1	数据寄存器 (R0/R0H/R0L/R1/R1H/R1L/R2/R3)	4
1.3.2	地址寄存器 (A0/A1)	5
1.3.3	帧基址寄存器 (FB)	5
1.3.4	程序计数器 (PC)	5
1.3.5	中断表寄存器 (INTB)	5
1.3.6	用户堆栈指针 (USP) /中断堆栈指针 (ISP)	5
1.3.7	静态基址寄存器 (SB)	5
1.3.8	标志寄存器 (FLG)	5
1.4	标志寄存器 (FLG)	6
1.4.1	位 0: 进位标志 (C 标志)	6
1.4.2	位 1: 调试标志 (D 标志)	6
1.4.3	位 2: 零标志 (Z 标志)	6
1.4.4	位 3: 符号标志 (S 标志)	6
1.4.5	位 4: 寄存器组指定标志 (B 标志)	6
1.4.6	位 5: 溢出标志 (O 标志)	6
1.4.7	位 6: 中断允许标志 (I 标志)	6
1.4.8	位 7: 堆栈指针指定标志 (U 标志)	6
1.4.9	位 8~位 11: 保留位	6
1.4.10	位 12~位 14: 处理器中断优先级 (IPL)	7
1.4.11	位 15: 保留位	7
1.5	寄存器组	8
1.6	复位解除后的内部状态	9
1.7	数据类型	10
1.7.1	整数	10
1.7.2	10 进制	11
1.7.3	位	12
1.7.4	字符串	15
1.8	数据分配	16
1.8.1	寄存器的数据分配	16
1.8.2	存储器内的数据分配	17
1.9	指令格式	18
1.9.1	一般格式 (:G)	18
1.9.2	快速格式 (:Q)	18
1.9.3	短格式 (:S)	18
1.9.4	零格式 (:Z)	18
1.10	向量表	19
1.10.1	固定向量表	19
1.10.2	可变向量表	20

第 2 章	寻址方式	
2.1	寻址方式	22
2.1.1	一般指令寻址	22
2.1.2	特殊指令寻址	22
2.1.3	位指令寻址	22
2.2	本章的阅读方法	23
2.3	一般指令寻址	24
2.4	特殊指令寻址	27
2.5	位指令寻址	30
第 3 章	功能	
3.1	本章的阅读方法	34
3.2	功能	39
第 4 章	指令码/周期数	
4.1	本章的阅读方法	138
4.2	指令码/周期数	140
第 5 章	中断	
5.1	中断概要	248
5.1.1	中断分类	248
5.1.2	软件中断	249
5.1.3	硬件中断	250
5.2	中断控制	251
5.2.1	中断允许标志 (I 标志)	251
5.2.2	中断请求位	251
5.2.3	中断优先级选择位及处理器中断优先级 (IPL)	252
5.2.4	中断控制寄存器的改变	253
5.3	中断响应顺序	254
5.3.1	中断响应时间	255
5.3.2	接受中断请求时的处理器中断优先级 (IPL) 的变化	256
5.3.3	寄存器保存	257
5.4	从中断程序的返回	258
5.5	中断优先权	259
5.6	多重中断	260
5.7	中断注意事项	262
5.7.1	0000016 地址的读出	262
5.7.2	SP 的设定	262
5.7.3	中断控制寄存器的改变	262
第 6 章	同期数的计算	
6.1	指令队列缓冲器	266

按英文字母分类的页一览表

助记符	功能 记载页	指令码/周期数 记载页	助记符	功能 记载页	指令码/周期数 记载页
ABS	39	140	DIVU	68	173
ADC	40	140	DIVX	69	174
ADCF	41	142	DSBB	70	175
ADD	42	142	DSUB	71	177
ADJNZ	44	148	ENTER	72	179
AND	45	149	EXITD	73	180
BAND	47	152	EXTS	74	180
BCLR	48	152	FCLR	75	181
BMCnd	49	154	FSET	76	182
BMEQ/Z	49	154	INC	77	182
BMGE	49	154	INT	78	183
BMGEU/C	49	154	INTO	79	184
BMGT	49	154	JCnd	80	184
BMGTU	49	154	JEQ/Z	80	184
BMLE	49	154	JGE	80	184
BMLEU	49	154	JGEU/C	80	184
BMLT	49	154	JGT	80	184
BMLTU/NC	49	154	JGTU	80	184
BMN	49	154	JLE	80	184
BMNE/NZ	49	154	JLEU	80	184
BMNO	49	154	JLT	80	184
BMO	49	154	JLTU/NC	80	184
BMPZ	49	154	JN	80	184
BNAND	50	155	JNE/NZ	80	184
BNOR	51	156	JNO	80	184
BNOT	52	156	JO	80	184
BNTST	53	157	JPZ	80	184
BNXOR	54	158	JMP	81	185
BOR	55	158	JMPI	82	187
BRK	56	159	JMPS	83	188
BSET	57	159	JSR	84	189
BTST	58	160	JSRI	85	190
BTSTC	59	161	JSRS	86	191
BTSTS	60	162	LDC	87	191
BXOR	61	162	LDCTX	88	192
CMP	62	163	LDE	89	193
DADC	64	167	LDINTB	90	194
DADD	65	169	LDIPL	91	195
DEC	66	171	MOV	92	195
DIV	67	172	MOVA	94	202

按英文字母分类的页一览表

助记符	功能 记载页	指令码/周期数 记载页	助记符	功能 记载页	指令码/周期数 记载页
MOV <i>Dir</i>	95	203	ROT	114	222
MOVHH	95	203	RTS	115	223
MOVHL	95	203	SBB	116	224
MOVLH	95	203	SBJNZ	117	226
MOVLL	95	203	SHA	118	227
MUL	96	205	SHL	119	230
MULU	97	207	SMOVB	120	232
NEG	98	209	SMOVF	121	233
NOP	99	209	SSTR	122	233
NOT	100	210	STC	123	234
OR	101	211	STCTX	124	235
POP	103	213	STE	125	235
POPC	104	215	STNZ	126	237
POPM	105	215	STZ	127	237
PUSH	106	216	STZX	128	238
PUSHA	107	218	SUB	129	238
PUSHC	108	218	TST	131	241
PUSHM	109	219	UND	132	243
REIT	110	219	WAIT	133	243
RMPA	111	220	XCHG	134	244
ROLC	112	220	XOR	135	245
RORC	113	221			

按功能分类的页一览表

功能	助记符	内容	功能 记载页	指令码/周期数 记载页
传送	MOV	传送	92	195
	MOVA	有效地址的传送	94	202
	MOVDir	4 位数据的传送	95	203
	POP	寄存器/存储器的恢复	103	213
	POPM	多个寄存器的恢复	105	215
	PUSH	寄存器/存储器/立即数的保存	106	216
	PUSHA	有效地址的保存	107	218
	PUSHM	多个寄存器的保存	109	219
	LDE	从扩展数据区的传送	89	193
	STE	向扩展数据区的传送	125	235
	STNZ	条件传送	126	237
	STZ	条件传送	127	237
	STZX	条件传送	128	238
	XCHG	交换	134	244
位处理	BAND	位逻辑与	47	152
	BCLR	位清除	48	152
	BM <i>Cnd</i>	条件位传送	49	154
	BNAND	反转位的逻辑与	50	155
	BNOR	反转位的逻辑或	51	156
	BNOT	位反转	52	156
	BNTST	反转位的测试	53	157
	BNXOR	反转位的逻辑异或	54	158
	BOR	位逻辑或	55	158
	BSET	置位	57	159
	BTST	位测试	58	160
	BTSTC	位测试和清除	59	161
	BTSTS	位测试和置位	60	162
	BXOR	位逻辑异或	61	162
移位	ROL	带进位的循环左移	112	220
	ROR	带进位的循环右移	113	221
	ROT	循环移位	114	222
	SHA	算术移位	118	227
	SHL	逻辑移位	119	230
算术	ABS	绝对值	39	140
	ADC	带进位的加法运算	40	140
	ADCF	进位标志的加法运算	41	142
	ADD	无进位的加法运算	42	142
	CMP	比较	62	163
	DADC	带进位的 10 进制加法运算	64	167

按功能分类的页一览表

功能	助记符	内容	功能 记载页	指令码/周期数 记载页
算术	DADD	无进位的 10 进制加法运算	65	169
	DEC	递减	66	171
	DIV	带符号的除法运算	67	172
	DIVU	无符号的除法运算	68	173
	DIVX	带符号的除法运算	69	174
	DSBB	带借位的 10 进制减法运算	70	175
	DSUB	无借位的 10 进制减法运算	71	177
	EXTS	符号扩展	74	180
	INC	递增	77	182
	MUL	带符号的乘法运算	96	205
	MULU	无符号的乘法运算	97	207
	NEG	2 的补码	98	209
	RMPA	乘加运算	111	220
	SBB	带借位的减法运算	116	224
	SUB	无借位的减法运算	129	238
逻辑	AND	逻辑与	45	149
	NOT	全位反转	100	210
	OR	逻辑或	101	211
	TST	测试	131	241
	XOR	逻辑异或	135	245
转移	ADJNZ	加法运算和条件转移	44	148
	SBJNZ	减法运算和条件转移	117	226
	JCnd	条件转移	80	184
	JMP	无条件转移	81	185
	JMPI	间接转移	82	187
	JMPS	专用页转移	83	188
	JSR	子程序调用	84	189
	JSRI	间接子程序调用	85	190
	JSRS	专用页子程序调用	86	191
	RTS	从子程序的返回	115	223
字符串	SMOVB	反向字符串传送	120	232
	SMOVF	正向字符串传送	121	233
	SSTR	字符串保存	122	233
其它	BRK	调试中断	56	159
	ENTER	堆栈帧的建立	72	179
	EXITD	堆栈帧的释放	73	180
	FCLR	标志寄存器的位清除	75	181
	FSET	标志寄存器的置位	76	182
	INT	软件中断	78	183
	INTO	溢出中断	79	184
	LDC	向专用寄存器的传送	87	191

按功能分类的页一览表

功能	助记符	内容	功能 记载页	指令码/周期数 记载页
其它	LDCTX	环境恢复	88	192
	LDINTB	向 INTB 寄存器的传送	90	194
	LDIPL	中断允许级的设定	91	195
	NOP	空操作	99	209
	POPC	专用寄存器的恢复	104	215
	PUSHC	专用寄存器的保存	108	218
	REIT	从中断的返回	110	219
	STC	从专用寄存器的传送	123	234
	STCTX	环境保存	124	235
	UND	未定义指令中断	132	243
	WAIT	等待	133	243

按寻址分类的页一览表（一般指令寻址）

助记符	寻址														功能 记载页	指令码/ 周期数 记载页	
	R0L/R0	R0H/R1	R1L/R2	R1H/R3	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16	#IMM8	#IMM16	#IMM20			#IMM
ABS	○	○	○	○	○	○	○	○	○	○	○					39	140
ADC	○	○	○	○	○	○	○	○	○	○	○	○	○			40	140
ADCF	○	○	○	○	○	○	○	○	○	○	○					41	142
ADD* ¹	○	○	○	○	○	○	○	○	○	○	○	○	○			42	142
ADJNZ* ¹	○	○	○	○	○	○	○	○	○	○	○				○	44	148
AND	○	○	○	○	○	○	○	○	○	○	○	○	○			45	149
CMP	○	○	○	○	○	○	○	○	○	○	○	○	○			62	163
DADC	○	○											○	○		64	167
DADD	○	○											○	○		65	169
DEC	○	○			○			○			○					66	171
DIV	○	○	○	○	○	○	○	○	○	○	○	○	○	○		67	172
DIVU	○	○	○	○	○	○	○	○	○	○	○	○	○	○		68	173
DIVX	○	○	○	○	○	○	○	○	○	○	○	○	○	○		69	174
DSBB	○	○											○	○		70	175
DSUB	○	○											○	○		71	177
ENTER													○			72	179
EXTS	○		○* ²			○	○	○	○	○	○					74	180
INC	○* ³	○* ⁴			○			○			○					77	182
INT															○	78	183
JMPI* ¹	○	○	○	○	○	○	○	○		○	○					82	187
JMPS													○			83	188
JSRI* ¹	○	○	○	○	○	○	○	○		○	○					85	190
JSRS													○			86	191
LDC* ¹	○	○	○	○	○	○	○	○	○	○	○			○		87	191
LDE* ¹	○	○	○	○	○	○	○	○	○	○	○					89	193

*1 有特殊的指令寻址。

*2 只能选择RIL。

*3 只能选择R0L。

*4 只能选择R0H。

按寻址分类的页一览表（一般指令寻址）

助记符	寻址														功能 记载页	指令码/ 周期数/ 记载页
	R0L/R0	R0H/R1	R1L/R2	R1H/R3	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16	#MM8	#MM16	#MM20	#MM	
LDINTB														○		90 194
LDIPL															○	91 195
MOV ^{*1}	○	○	○	○	○	○	○	○	○	○	○	○	○			92 195
MOVA	○	○	○	○	○		○	○	○	○	○					94 202
MOV _{Dir}	○	○	○	○		○	○	○	○	○	○					95 203
MUL	○	○	○	○	○	○	○	○	○	○	○	○	○			96 205
MULU	○	○	○	○	○	○	○	○	○	○	○	○	○			97 207
NEG	○	○	○	○	○	○	○	○	○	○	○					98 209
NOT	○	○	○	○	○	○	○	○	○	○	○					100 210
OR	○	○	○	○	○	○	○	○	○	○	○	○	○			101 211
POP	○	○	○	○	○	○	○	○	○	○	○					103 213
POPM ^{*1}	○	○	○	○	○											105 215
PUSH	○	○	○	○	○	○	○	○	○	○	○					106 216
PUSHA							○	○	○	○	○					107 218
PUSHM ^{*1}	○	○	○	○	○											109 219
ROL	○	○	○	○	○	○	○	○	○	○	○					112 220
ROR	○	○	○	○	○	○	○	○	○	○	○					113 221
ROT	○	○	○	○	○	○	○	○	○	○	○				○	114 222
SBB	○	○	○	○	○	○	○	○	○	○	○	○	○			116 224
SBJNZ ^{*1}	○	○	○	○	○	○	○	○	○	○	○				○	117 226
SHA ^{*1}	○	○	○	○	○	○	○	○	○	○	○				○	118 227
SHL ^{*1}	○	○	○	○	○	○	○	○	○	○	○				○	119 230
STC ^{*1}	○	○	○	○	○	○	○	○	○	○	○					123 234
STCTX ^{*1}											○					124 235
STE ^{*1}	○	○	○	○	○	○	○	○	○	○	○					125 235

*1 有特殊的指令寻址。

按寻址分类的页一览表（一般指令寻址）

助记符	寻址														功能 记载页	指令码/ 周期数 记载页
	R0L/R0	R0H/R1	R1L/R2	R1H/R3	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16	#IMM8	#IMM16	#IMM20	#IMM	
STNZ	○	○						○			○	○			126	237
STZ	○	○						○			○	○			127	237
STZX	○	○						○			○	○			128	238
SUB	○	○	○	○	○	○	○	○	○	○	○	○	○		129	238
TST	○	○	○	○	○	○	○	○	○	○	○	○	○		131	241
XCHG	○	○	○	○	○	○	○	○	○	○	○				134	244
XOR	○	○	○	○	○	○	○	○	○	○	○	○	○		135	245

*1 有特殊的指令寻址。

按寻址分类的页一览表（特殊指令寻址）

助记符	寻址												功能 记载页	指令码/ 周期数 记载页
	dsp:20[A0]	dsp:20[A1]	abs20	R2R0/R3R1	A1A0	[A1A0]	dsp:8[SP]	label	SB/FB	ISP/USP	FLG	INTBL/INTBH	PC	
ADD ^{*1}										○				42 142
ADJNZ ^{*1}								○						44 148
JCnd								○						80 184
JMP			○					○						81 185
JMPI ^{*1}	○	○		○	○									82 187
JSR			○					○						84 189
JSRI ^{*1}	○	○		○	○									85 190
LDC ^{*1}									○	○	○	○		87 191
LDCTX			○											88 192
LDE ^{*1}	○		○			○								89 193
LDINTB												○ ^{*2}		90 194
MOV ^{*1}							○							92 195
POPC									○	○	○	○		104 215
POPM ^{*1}									○					105 215
PUSHC									○	○	○	○		108 218
PUSHM ^{*1}									○					109 219
SBJNZ ^{*1}								○						117 226
SHA ^{*1}				○										118 227
SHL ^{*1}				○										119 230
STC ^{*1}				○	○				○	○	○	○	○	123 234
STCTX ^{*1}			○											124 235
STE ^{*1}	○		○			○								125 235

*1 有一般的指令寻址。

*2 如果使用 LDINTB 指令，就能同时设定 INTBL 和 INTBH。

按寻址分类的页一览表（位指令寻址）

助记符	寻址										功能 记载页	指令码/ 周期数 记载页
	bit,Rn	bit,An	[An]	base:8[An]	bit,base:8[SB/FB]	base:16[An]	bit,base:16[SB]	bit,base: 16	bit,base:11	U//O/B/S/Z/D/C		
BAND	○	○	○	○	○	○	○	○			47	152
BCLR	○	○	○	○	○	○	○	○	○		48	152
BMCnd	○	○	○	○	○	○	○	○		○	49	154
BNAND	○	○	○	○	○	○	○	○			50	155
BNOR	○	○	○	○	○	○	○	○			51	156
BNOT	○	○	○	○	○	○	○	○	○		52	156
BNTST	○	○	○	○	○	○	○	○			53	157
BNXOR	○	○	○	○	○	○	○	○			54	158
BOR	○	○	○	○	○	○	○	○			55	158
BSET	○	○	○	○	○	○	○	○	○		57	159
BTST	○	○	○	○	○	○	○	○	○		58	160
BTSTC	○	○	○	○	○	○	○	○			59	161
BTSTS	○	○	○	○	○	○	○	○			60	162
BXOR	○	○	○	○	○	○	○	○			61	162
FCLR										○	75	181
FSET										○	76	182

第 1 章

概要

- 1.1 M16C/60 、 M16C/20 、
M16C/Tiny系列的特点
- 1.2 地址空间
- 1.3 寄存器构成
- 1.4 标志寄存器（FLG）
- 1.5 寄存器组
- 1.6 复位解除后的内部状态
- 1.7 数据类型
- 1.8 数据分配
- 1.9 指令格式
- 1.10 向量表

1.1 M16C/60、M16C/20、M16C/Tiny 系列的特点

M16C/60、M16C/20、M16C/Tiny系列是以嵌入式设备为对象开发的单芯片微型计算机。因具有适合C语言的指令，并且将频繁使用的指令分配成1字节的操作码，所以，无论使用汇编语言还是使用C语言，都能开发出占用存储容量小的、高效率的程序。另外，具有以1个时钟周期执行的指令，实现了高速运算处理。

具有对应丰富的寻址方式的91种指令的指令集，能进行寄存器-寄存器、寄存器-存储器、存储器-存储器之间的运算，并且能进行以位和4位数据为对象的运算。

对于具有内部乘法器的产品，能进行高速乘法运算。

1.1.1 M16C/60、M16C/20、M16C/Tiny 系列的特点

- 寄存器结构

数据寄存器16位×4个（其中2个能作为8位寄存器使用）

地址寄存器16位×2个

基址寄存器16位×2个

- 有特点的指令集

适合C语言的指令（堆栈帧操作） : ENTER、EXITD等

不区分寄存器和存储器的指令 : MOV、ADD、SUB等

强大的位处理指令 : BNOT、BTST、BSET等

4位传送指令 : MOVLL、MOVHL等

频繁使用的1字节指令 : MOV、ADD、SUB、JMP等

高速的1个周期指令 : MOV、ADD、SUB等

- 1M字节的线性地址空间

对应转移距离的相对跳转指令

- 高速的指令执行时间

最短1个周期指令：在91种指令中，20种指令为1个周期指令

（大约75%的指令为5个周期以下）

1.1.2 速度性能

寄存器之间的传送 0.125μs

寄存器和存储器之间的传送 0.125μs

寄存器之间的加减运算 0.125μs

8位×8位寄存器之间的运算 0.25μs

16位×16位寄存器之间的运算 0.313μs

16位÷8位寄存器之间的运算 1.13μs

32位÷16位寄存器之间的运算 1.56μs

[条件]

- 具有内部乘法器产品

- 时钟频率 16MHz

1.2 地址空间

地址空間如图1.2.1所示。

00000₁₆地址～003FF₁₆地址为SFR（专用功能寄存器）区。根据产品种类，从003FF₁₆地址开始向低地址方向扩展SFR区。

00400₁₆地址以后的区域为存储器区。根据产品种类，从00400₁₆地址开始向高地址方向扩展RAM区，从FFFFF₁₆地址开始向低地址方向扩展ROM区。但是，FFE00₁₆地址～FFFFF₁₆地址为固定向量区。

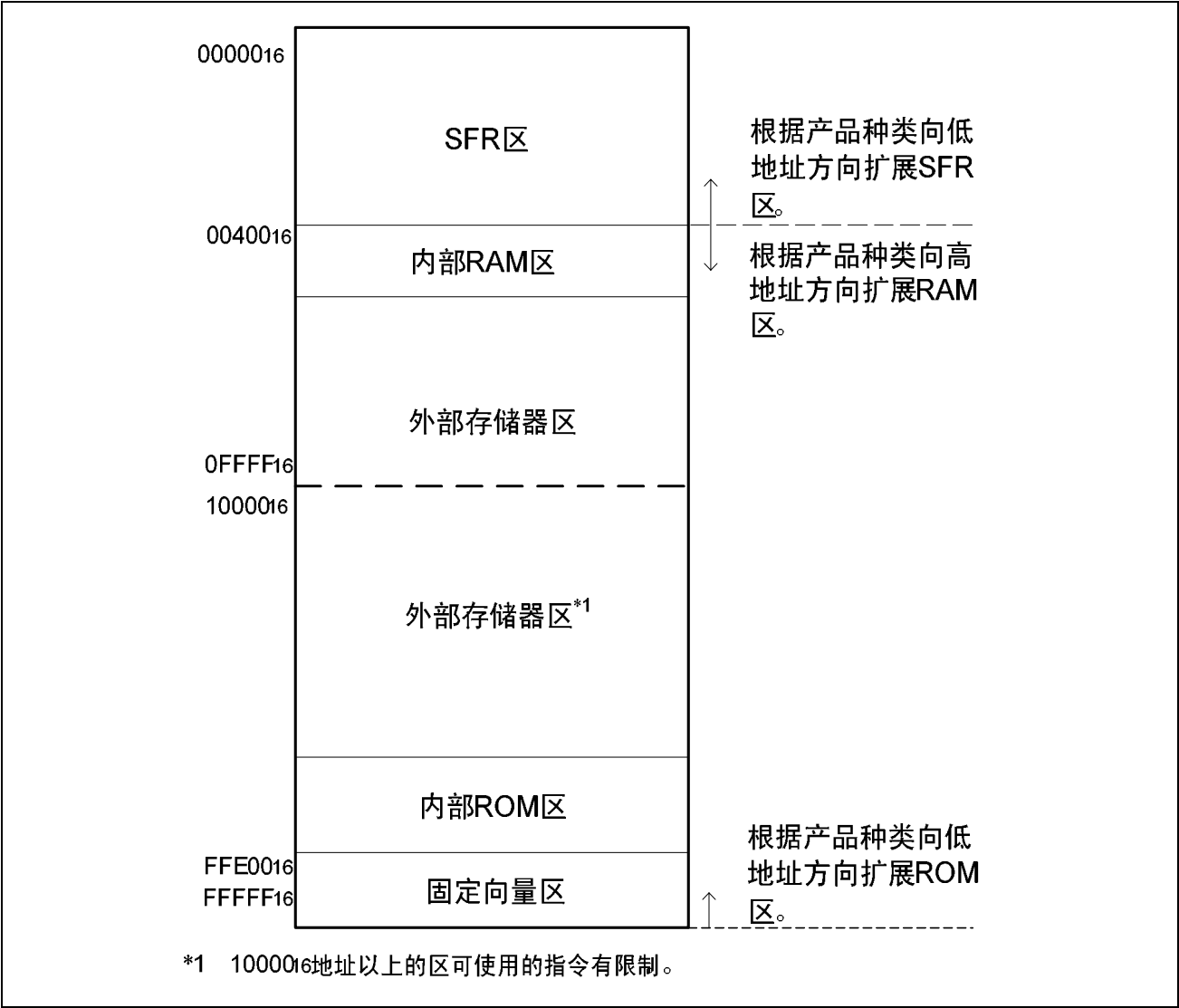


图1.2.1 地址空间

1.3 寄存器构成

中央处理器有图1.3.1所示的13个寄存器。其中R0、R1、R2、R3、A0、A1、FB的7个寄存器有2组，构成2个寄存器组。

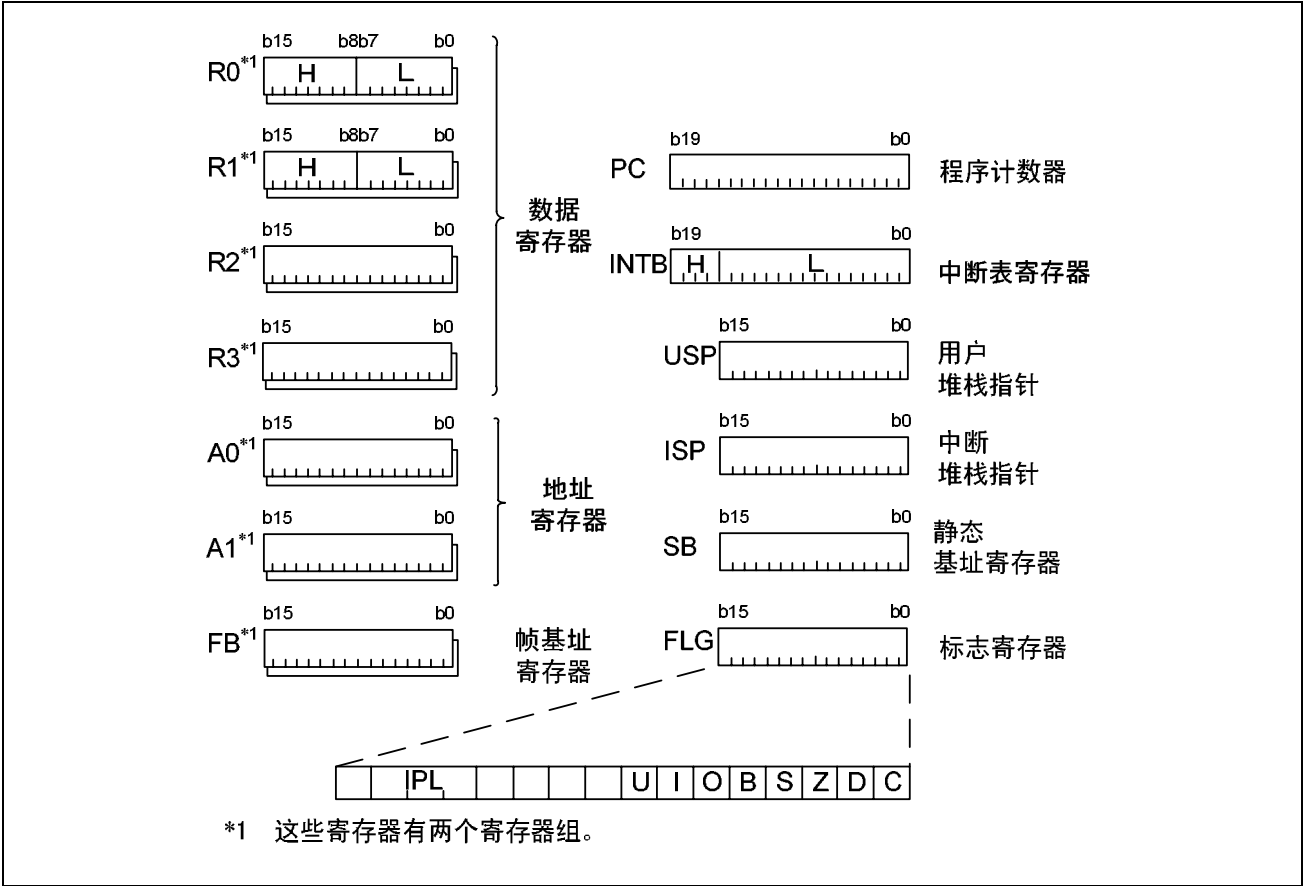


图1.3.1 中央处理器的寄存器构成

1.3.1 数据寄存器（R0/R0H/R0L/R1/R1H/R1L/R2/R3）

数据寄存器（R0/R1/R2/R3）由16位构成，主要用于传送、算术和逻辑运算。

能将R0/R1的高位（R0H/R1H）和低位（R0L/R1L）分别作为8位数据寄存器使用。另外，在一部分的指令中，也能将R2和R0、R3和R1组合成32位数据寄存器（R2R0/R3R1）使用。

1.3.2 地址寄存器 (A0/A1)

地址寄存器 (A0/A1) 由16位构成, 持有和数据寄存器同等的功能。用于地址寄存器间接寻址和地址寄存器相对寻址。

在一部分的指令中, 也能将A1和A0组合成32位地址寄存器 (A1A0) 使用。

1.3.3 帧基址寄存器 (FB)

帧基址寄存器 (FB) 由16位构成, 用于FB相对寻址。

1.3.4 程序计数器 (PC)

程序计数器 (PC) 由20位构成, 指示下次执行的指令的地址。

1.3.5 中断表寄存器 (INTB)

中断表寄存器 (INTB) 由20位构成, 指示中断向量表的起始地址。

1.3.6 用户堆栈指针 (USP) / 中断堆栈指针 (ISP)

堆栈指针有用户堆栈指针 (USP) 和中断堆栈指针 (ISP) 2种, 都由16位构成。

能通过堆栈指针指定标志 (U标志), 切换使用的堆栈指针 (USP/ISP)。

堆栈指针指定标志 (U标志) 为标志寄存器 (FLG) 的位7。

1.3.7 静态基址寄存器 (SB)

静态基址寄存器 (SB) 由16位构成, 用于SB相对寻址。

1.3.8 标志寄存器 (FLG)

标志寄存器 (FLG) 由11位构成, 作为以位为单位的标志使用。各标志的功能请参照 “1.4 标志寄存器 (FLG)”。

1.4 标志寄存器（FLG）

标志寄存器（FLG）的构成如图1.4.1所示。各标志的功能如下所示：

1.4.1 位 0：进位标志（C 标志）

保存由算术逻辑运算器产生的进位、借位和移出位等。

1.4.2 位 1：调试标志（D 标志）

它是允许单步中断的标志。

当此标志为“1”时，在指令执行后产生单步中断。如果接受中断，此标志就变为“0”。

1.4.3 位 2：零标志（Z 标志）

在运算结果为0时置“1”，否则清“0”。

1.4.4 位 3：符号标志（S 标志）

在运算结果为负时置“1”，否则清“0”。

1.4.5 位 4：寄存器组指定标志（B 标志）

选择寄存器组。在此标志为“0”时，指定寄存器组0；在此标志为“1”时，指定寄存器组1。

1.4.6 位 5：溢出标志（O 标志）

在运算结果溢出时置“1”。

1.4.7 位 6：中断允许标志（I 标志）

它是允许屏蔽中断的标志。

在此标志为“0”时，禁止中断；在此标志为“1”时，允许中断。

如果接受中断，此标志就变为“0”。

1.4.8 位 7：堆栈指针指定标志（U 标志）

在此标志为“0”时，指定中断堆栈指针（ISP）；在此标志为“1”时，指定用户堆栈指针（USP）。

在接受了硬件中断或者执行了软件中断号0~31的INT指令时，此标志变为“0”。

1.4.9 位 8～位 11：保留位

1.4.10 位 12～位 14：处理器中断优先级（IPL）

处理器中断优先级（IPL）由3位构成，指定0～7级的8个处理器中断优先级。
如果请求的中断优先级高于处理器中断优先级（IPL），就允许该中断请求。

1.4.11 位 15：保留位

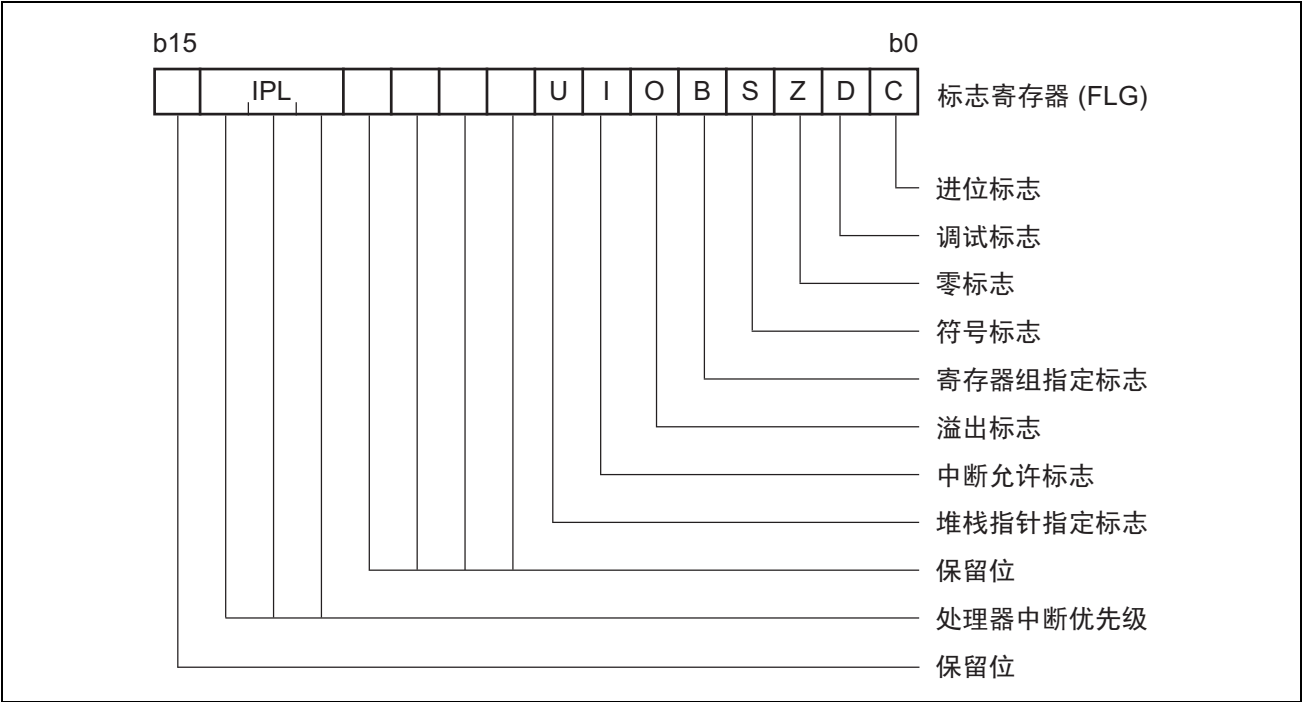


图1.4.1 标志寄存器（FLG）的构成

1.5 寄存器组

由数据寄存器（R0/R1/R2/R3）、地址寄存器（A0/A1）以及帧基址寄存器（FB）构成的寄存器组有2组。寄存器组通过标志寄存器（FLG）的寄存器组指定标志（B标志）切换。
寄存器组的构成如图1.5.1所示。

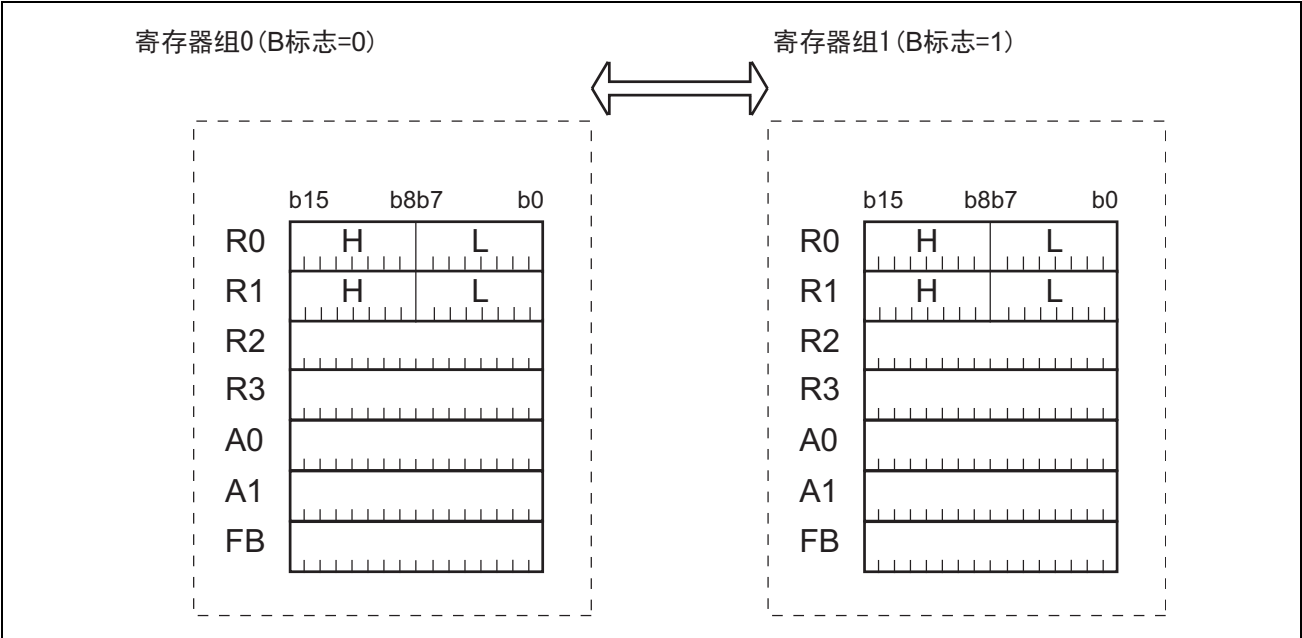


图1.5.1 寄存器组的构成

1.6 复位解除后的内部状态

复位解除后的各寄存器内容如下所示：

- 数据寄存器 (R0/R1/R2/R3) : 0000₁₆
- 地址寄存器 (A0/A1) : 0000₁₆
- 帧基址寄存器 (FB) : 0000₁₆
- 中断表寄存器 (INTB) : 00000₁₆
- 用户堆栈指针 (USP) : 0000₁₆
- 中断堆栈指针 (ISP) : 0000₁₆
- 静态基址寄存器 (SB) : 0000₁₆
- 标志寄存器 (FLG) : 0000₁₆

1.7 数据类型

数据类型为整数、10进制、位、字符串4种。

1.7.1 整数

整数有带符号整数和无符号整数。带符号整数的负值以2的补码表现。

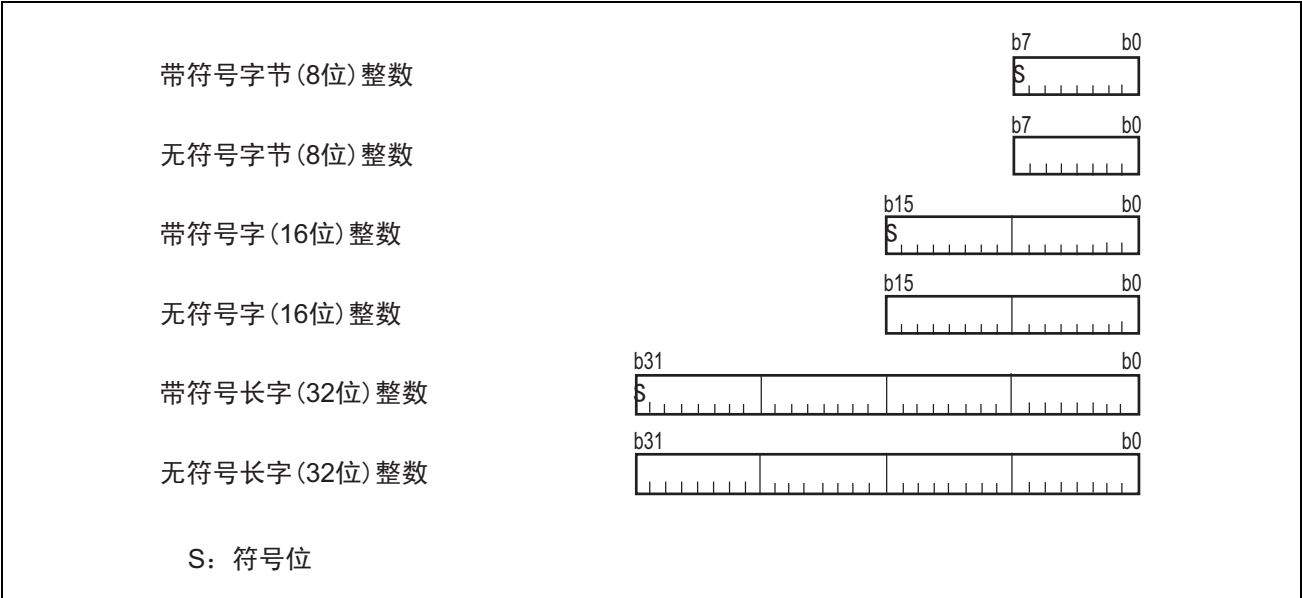


图1.7.1 整数数据

1.7.2 10 进制

10进制的数据类型能用于DADC、DADD、DSBB、DSUB的4种指令。

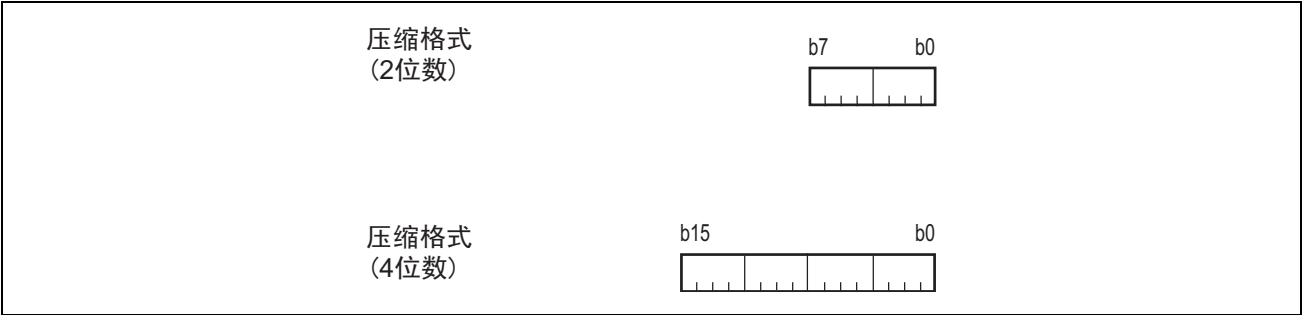


图1.7.2 10进制数据

1.7.3 位

(1) 寄存器的位

寄存器的位指定如图1.7.3所示。

寄存器的位能通过寄存器直接 (bit,Rn/bit,An) 指定。bit,Rn用于数据寄存器 (Rn) 的位指定, bit,An用于地址寄存器 (An) 的位指定。

各寄存器的位从LSB到MSB, 分别有0~15的位号。因此, bit,Rn/bit,An的bit能在0~15的范围中指定。

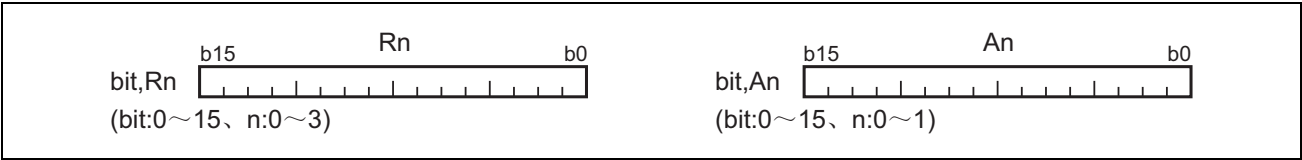


图1.7.3 寄存器的位指定

(2) 存储器的位名称

在指定存储器的位时使用的寻址方式如图1.7.4所示, 在各寻址方式中能指定位的地址如表1.7.1所示。存储器的位必须指定在表1.7.1所示的范围内。

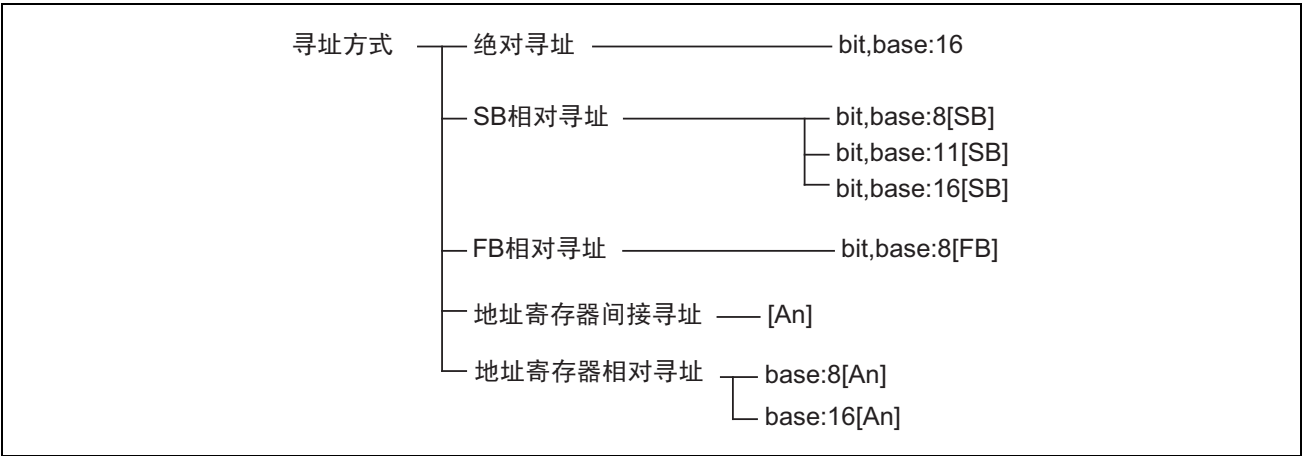


图1.7.4 在指定存储器的位时使用的寻址方式

表 1.7.1 在各寻址方式中能指定位的地址

寻址	指定范围		备 注
	下限 (地址)	上限 (地址)	
bit,base:16	00000 ₁₆	01FFF ₁₆	
bit,base:8[SB]	[SB]	[SB]+0001F ₁₆	存取范围为00000 ₁₆ ~0FFFF ₁₆ 。
bit,base:11[SB]	[SB]	[SB]+000FF ₁₆	存取范围为00000 ₁₆ ~0FFFF ₁₆ 。
bit,base:16[SB]	[SB]	[SB]+01FFF ₁₆	存取范围为00000 ₁₆ ~0FFFF ₁₆ 。
bit,base:8[FB]	[FB]-00010 ₁₆	[FB]+0000F ₁₆	存取范围为00000 ₁₆ ~0FFFF ₁₆ 。
[An]	00000 ₁₆	01FFF ₁₆	
base:8[An]	base:8	base:8+01FFF ₁₆	存取范围为00000 ₁₆ ~020FE ₁₆ 。
base:16[An]	base:16	base:16+01FFF ₁₆	存取范围为00000 ₁₆ ~0FFFF ₁₆ 。

① 通过bit,base的位指定

存储器映像和位映像的关系如图1.7.5所示。

存储器的位能作为连续的位数组处理。位的指定能通过任意的bit和base的组合进行。以设定给base的地址的位0为基准（0），将对象位的位置设定给bit。0000A₁₆地址的位2的指定例子如图1.7.6所示。

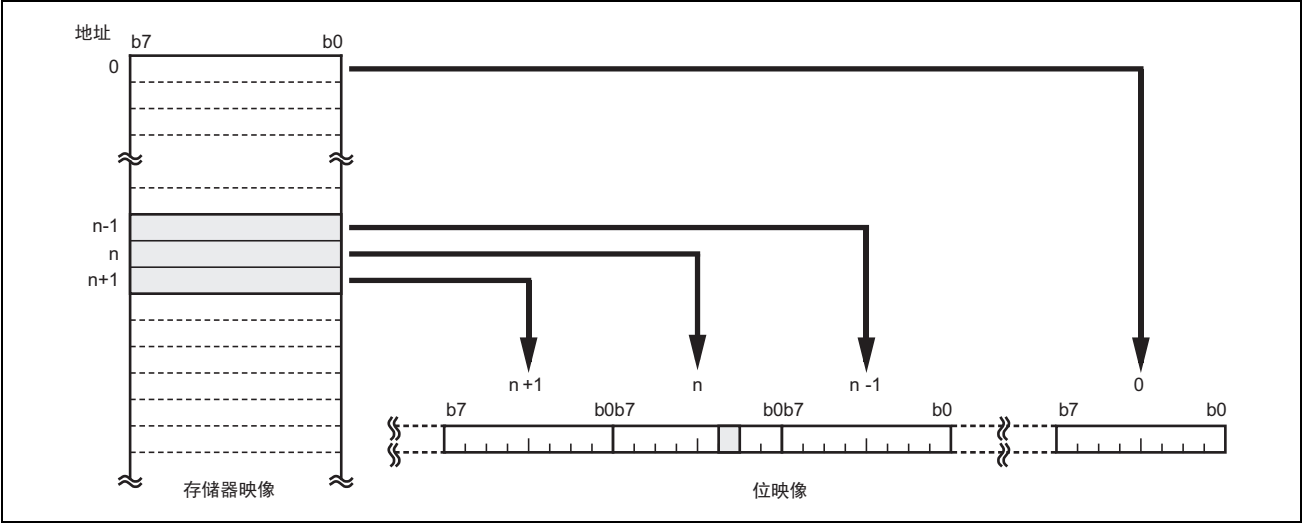


图1.7.5 存储器映像和位映像的关系

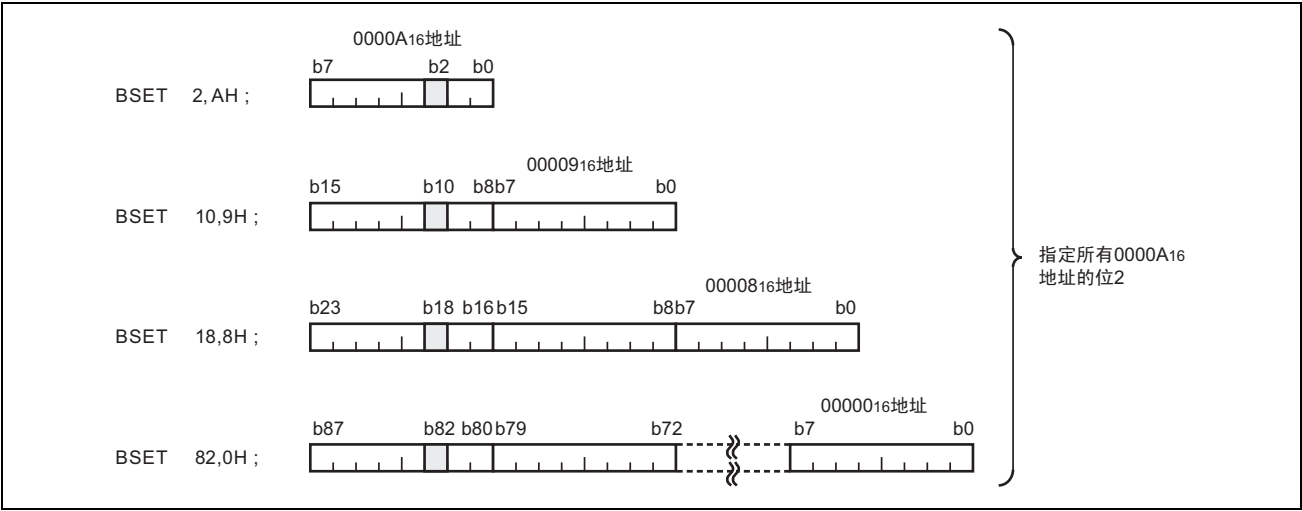


图1.7.6 0000A₁₆地址的位2的指定例子

② 通过SB/FB相对的位指定

在SB/FB相对寻址的情况下，以将设定给静态基址寄存器（SB）/帧基址寄存器（FB）的地址和设定给base的地址相加的地址的位0为基准（0），将对象位的位置设定给bit。

③ 通过地址寄存器间接/地址寄存器相对的位指定

在地址寄存器间接寻址的情况下，以00000₁₆地址的位0为基准（0），将对象位的位置设定给地址寄存器（An）。

在地址寄存器相对寻址的情况下，以设定给base的地址的位0为基准（0），将对象位的位置设定给地址寄存器（An）。

1.7.4 字符串

字符串是将任意长的字节（8位）或者字（16位）的数据连续排列的数据类型。

此数据类型能用于字符串指令的字符串反向传送（SMOVB指令）、字符串正向传送（SMOVF指令）以及指定区域的初始化（SSTR指令）的3种指令。

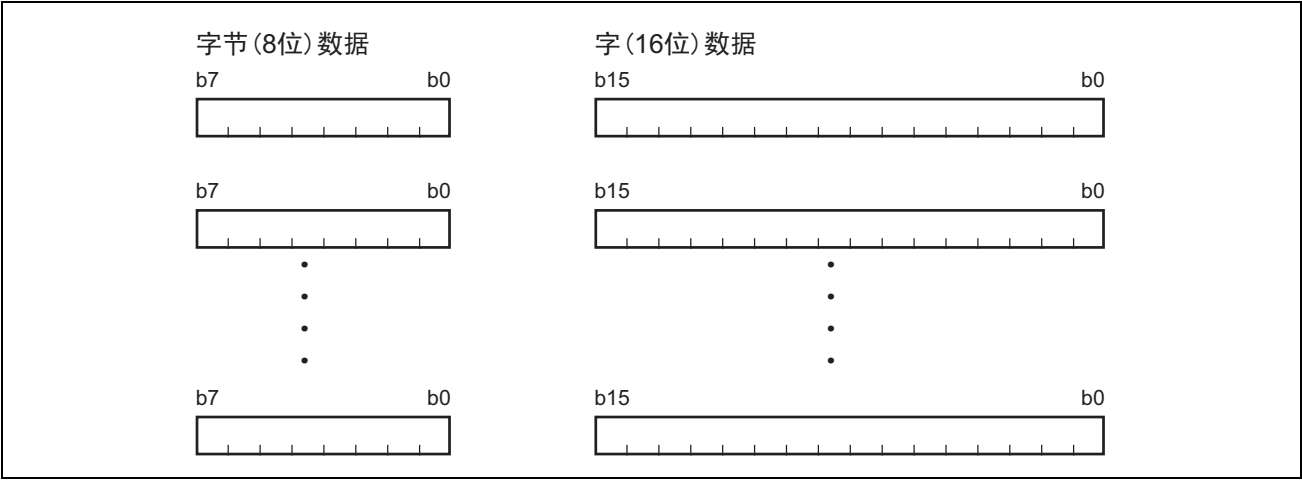


图1.7.7 字符串数据

1.8 数据分配

1.8.1 寄存器的数据分配

寄存器的数据长度和位号的关系如图1.8.1所示。

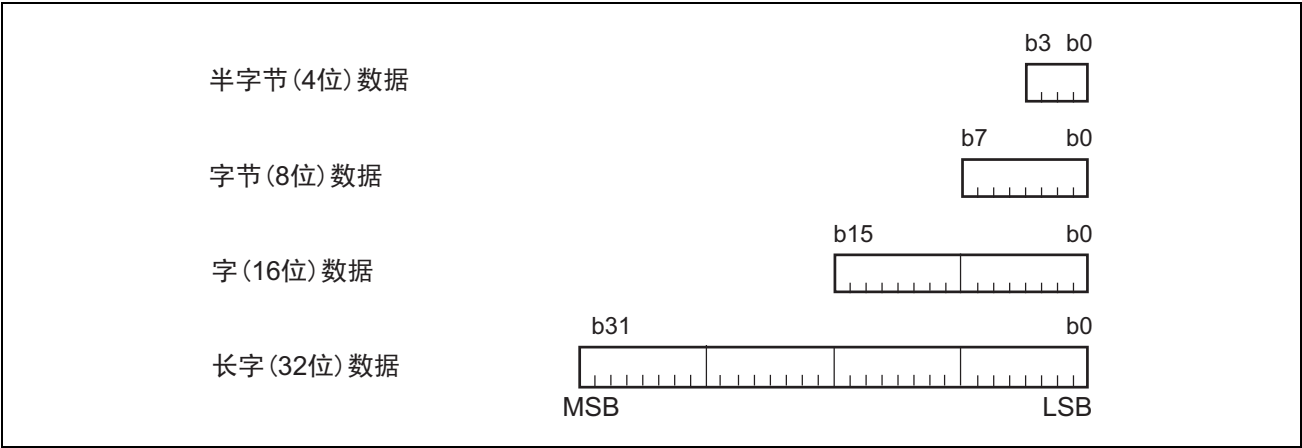


图1.8.1 寄存器内的数据分配

1.8.2 存储器内的数据分配

存储器内的数据分配如图1.8.2所示，运算例子如图1.8.3所示。

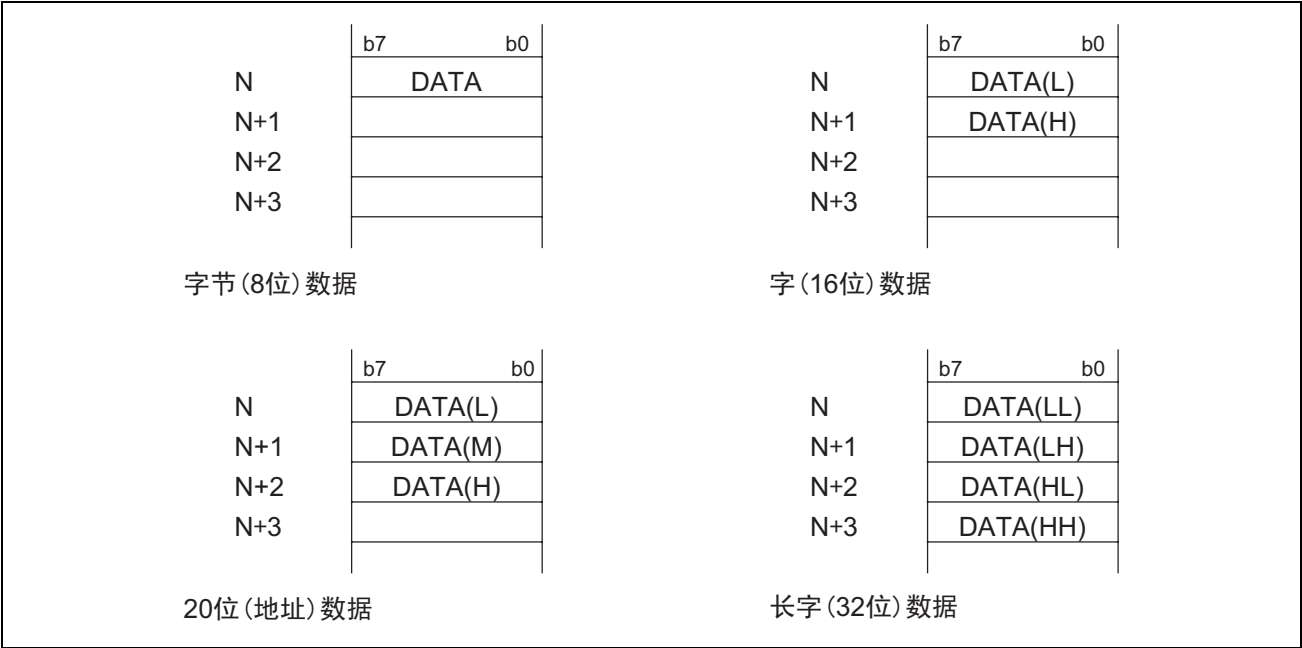


图1.8.2 存储器内的数据分配

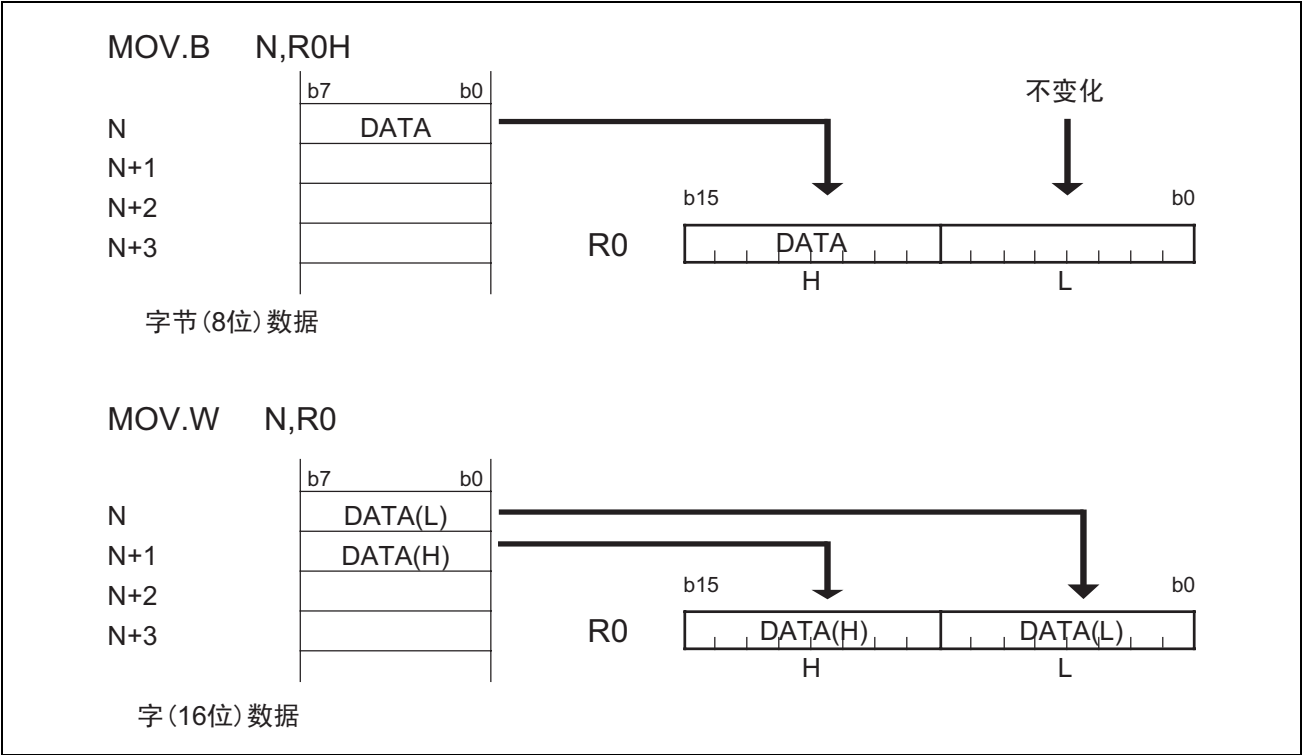


图1.8.3 运算例子

1.9 指令格式

指令格式能分成一般格式、快速格式、短格式、零格式4种。格式能选择的指令字节数为零格式最少，以短格式、快速格式、一般格式的顺序逐渐增多。

格式的特点如下所示：

1.9.1 一般格式（:G）

一般格式的操作码为2字节。此操作码包含操作信息、src^{*1}和dest^{*2}的寻址方式信息。

指令码由操作码（2字节）、src码（0~3字节）以及dest码（0~3字节）构成。

1.9.2 快速格式（:Q）

快速格式的操作码为2字节。此操作码包含操作信息、立即数和dest的寻址方式信息。但是，包含在操作码中的立即数为能以-7~+8或者-8~+7（根据指令不同）表现的数值。

指令码由包含立即数的操作码（2字节）和dest码（0~2字节）构成。

1.9.3 短格式（:S）

短格式的操作码为1字节。此操作码包含操作信息、src和dest的寻址方式信息。但是，能使用的寻址方式有限制。

指令码由操作码（1字节）、src码（0~2字节）以及dest码（0~2字节）构成。

1.9.4 零格式（:Z）

零格式的操作码为1字节。此操作码包含操作（及立即数）信息和dest的寻址方式信息。但是，立即数固定成0。另外，能使用的寻址方式也有限制。

指令码由操作码（1字节）和dest码（0~2字节）构成。

*1 source的简称

*2 destination的简称

1.10 向量表

向量表中有专用页向量表和中断向量表。专用页向量表为固定向量表。中断向量表有固定向量表和可变向量表。

1.10.1 固定向量表

固定向量表为地址固定的向量表。从FFE00₁₆地址到FFFDB₁₆地址配置为专用页向量表，从FFFDC₁₆地址到FFFFF₁₆地址配置为中断向量表的一部分。固定向量表如图1.10.1所示。

专用页向量表中1个向量表由2个字节构成。在各向量表中设定子程序的起始地址的低16位。并且，每个向量表都有专用页号（18~255），在JSRS及JMPS指令中，使用此专用页号。

中断向量表对于1个向量表由4字节构成。在各向量表中设定中断程序的起始地址。

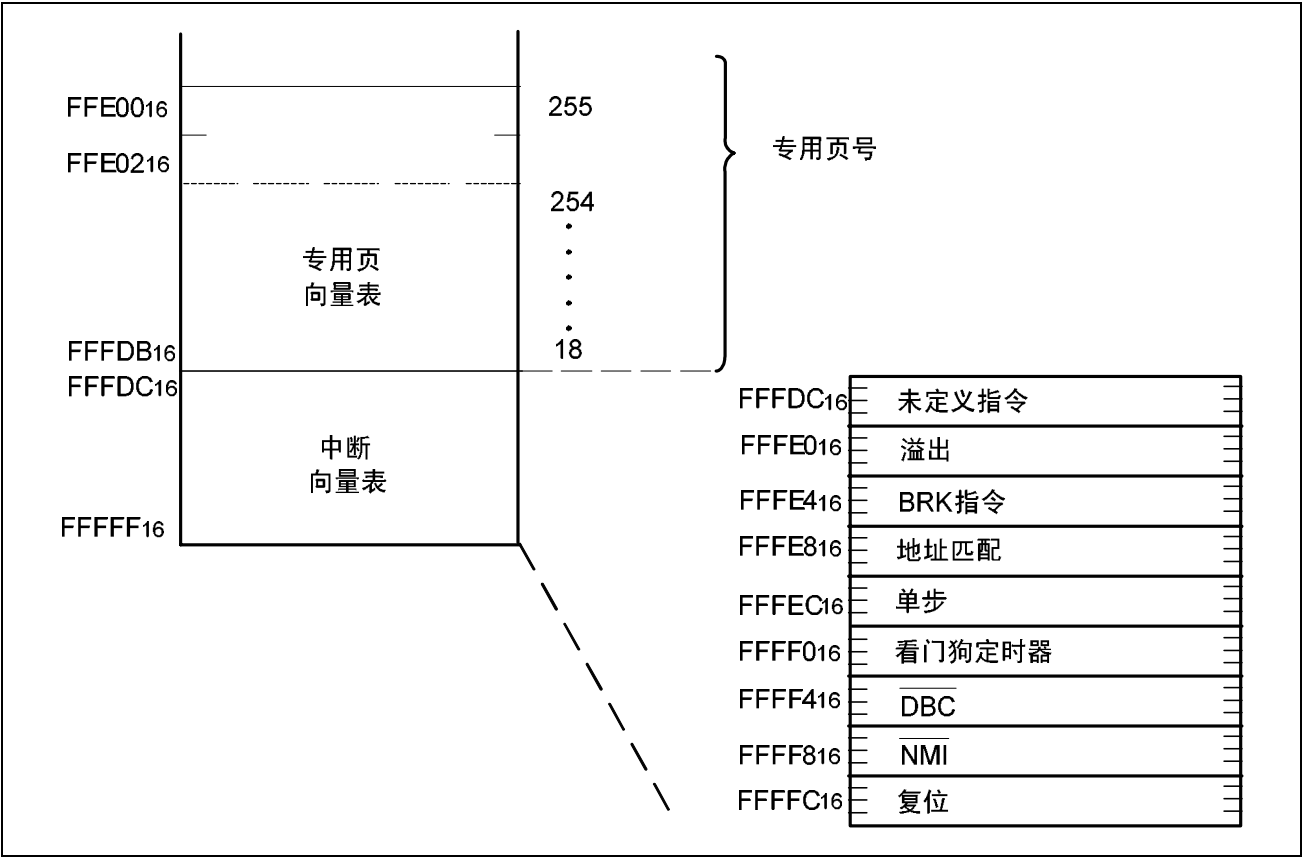


图1.10.1 固定向量表

1.10.2 可变向量表

可变向量表为能改变地址的向量表。可变向量表是以中断表寄存器（INTB）的内容所表示的值为起始地址（IntBase）的256字节的中断向量表。可变向量表如图1.10.2所示。

可变向量表对于1个向量表由4字节构成。在各向量表中设定中断程序的起始地址。

另外，每个向量表都有软件中断号（0～63），在INT指令中使用此软件中断号。

根据产品种类，内置的外围功能中断被分配在软件中断号0～31中。

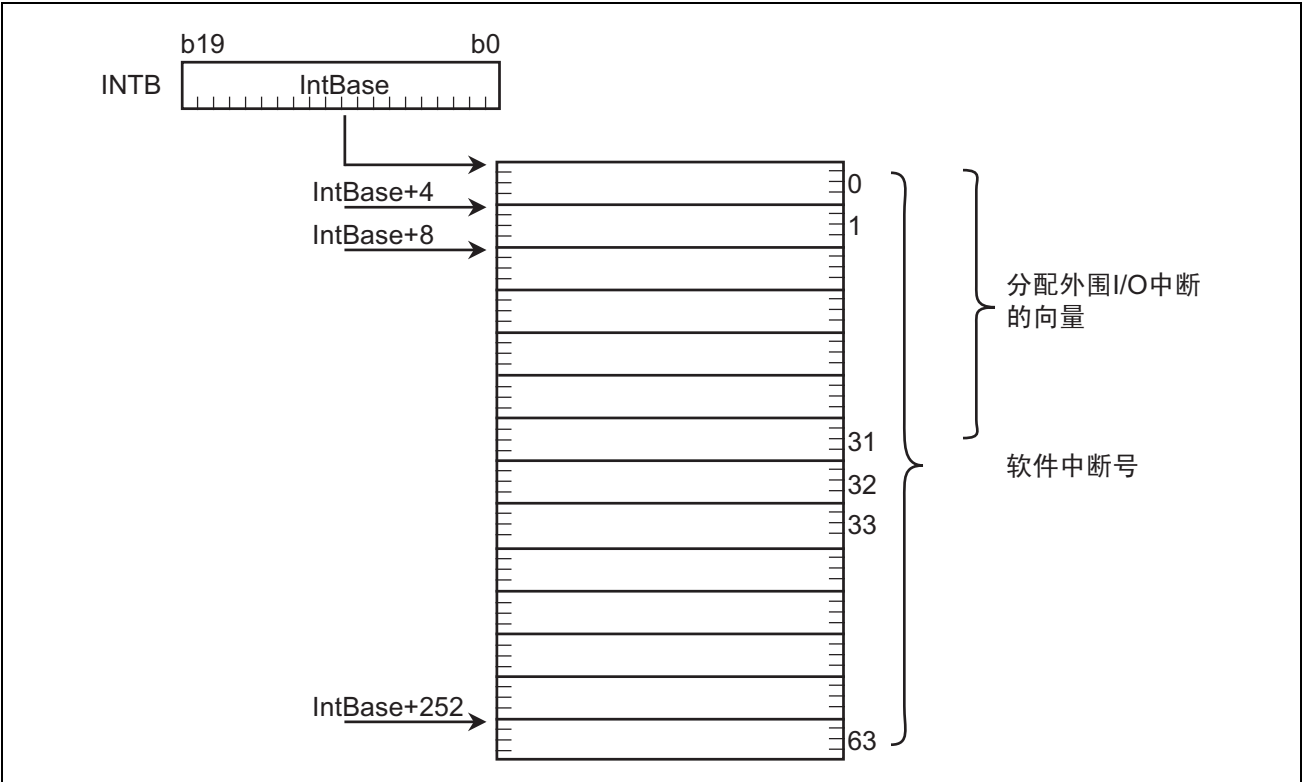


图1.10.2 可变向量表

第 2 章

寻址方式

- 2.1 寻址方式
- 2.2 本章的阅读方法
- 2.3 一般指令寻址
- 2.4 特殊指令寻址
- 2.5 位指令寻址

2.1 寻址方式

在本章中，按寻址方式说明有关表示寻址方式的符号和操作。

寻址方式有以下所示的3种：

2.1.1 一般指令寻址

它是存取从00000₁₆地址到0FFFF₁₆地址的区域的寻址。

一般指令寻址的各名称如下所示：

- 立即
- 寄存器直接
- 绝对
- 地址寄存器间接
- 地址寄存器相对
- SB相对
- FB相对
- 堆栈指针相对

2.1.2 特殊指令寻址

它是存取从00000₁₆地址到FFFFF₁₆地址的区域的寻址、以及存取专用寄存器的寻址。

特殊指令寻址的各名称如下所示：

- 20位绝对
- 20位带位移量的地址寄存器相对
- 32位地址寄存器间接
- 32位寄存器直接
- 专用寄存器直接
- 程序计数器相对

2.1.3 位指令寻址

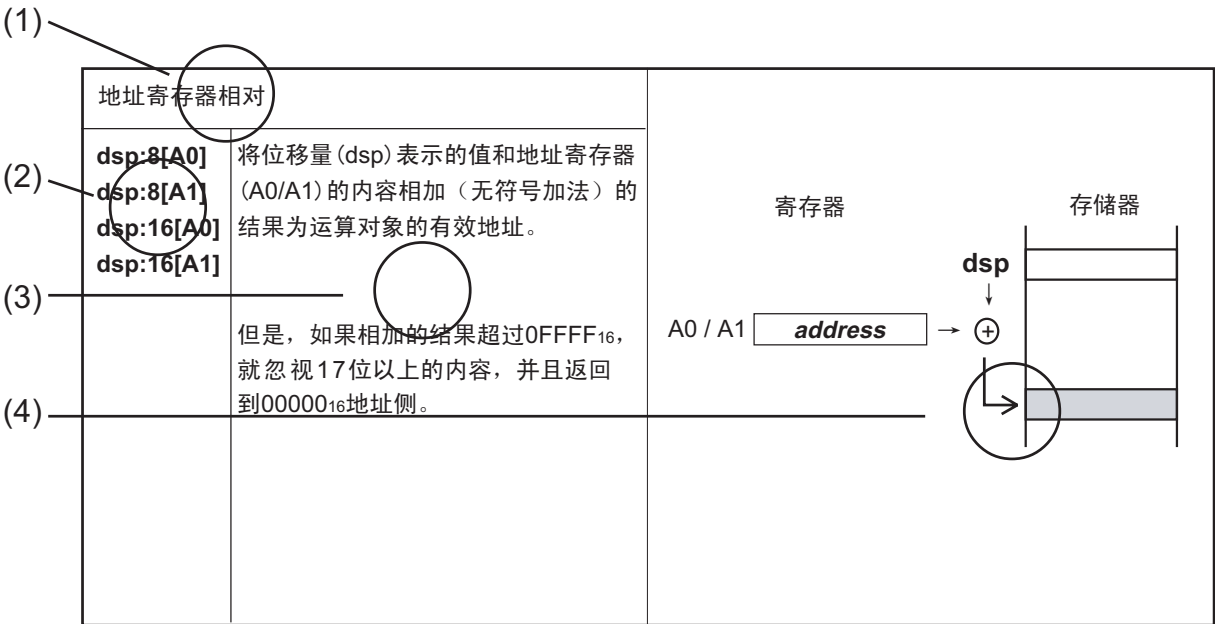
它是以位单位存取从00000₁₆地址到0FFFF₁₆地址的区域的寻址。

位指令寻址的各名称如下所示：

- 寄存器直接
- 绝对
- 地址寄存器间接
- 地址寄存器相对
- SB相对
- FB相对
- FLG直接

2.2 本章的阅读方法

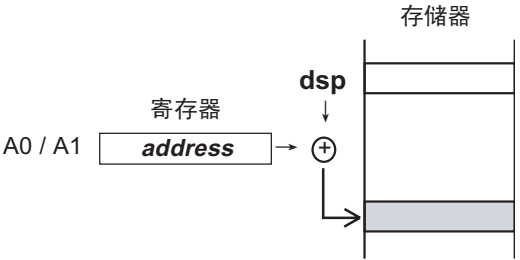
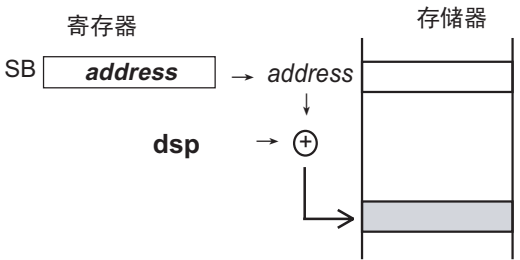
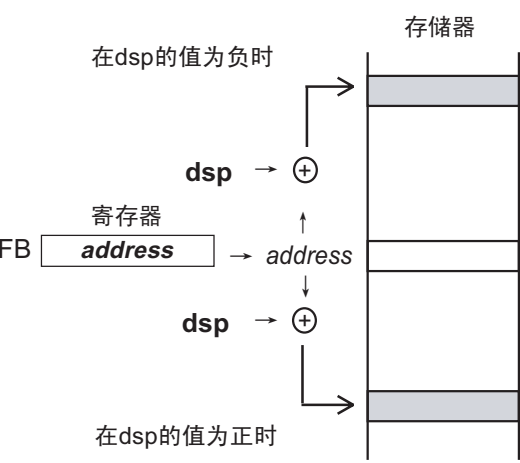
有关本章的阅读方法，举例说明如下：



- (1) 名称
表示寻址的名称。
- (2) 符号
表示寻址方式的符号。
- (3) 解说
说明地址的操作和有效地址的范围。
- (4) 操作图
用图说明地址的操作。

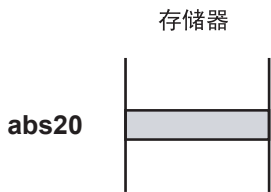
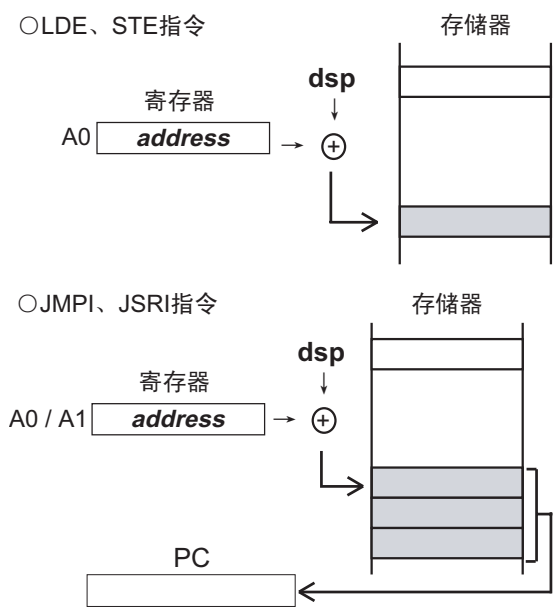
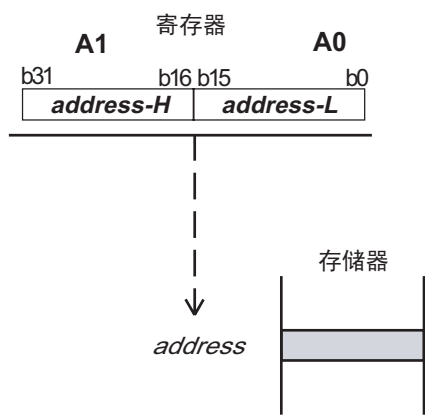
2.3 一般指令寻址

立即		
#IMM #IMM8 #IMM16 #IMM20	以#IMM表示的立即数为运算对象。	#IMM8 b7b0 #IMM16 b15b8b7b0 #IMM20 b19b15b8b7b0
寄存器直接		
R0L R0H R1L R1H R0 R1 R2 R3 A0 A1	以指定的寄存器为运算对象。	寄存器 R0L / R1L b0 R0H / R1H b15b8 R0 / R1 / R2 / R3 / A0 / A1 b15b8b7b0
绝对		
abs16	以abs16表示的值为运算对象的有效地址。 有效地址的范围为00000 ₁₆ ~0FFFF ₁₆ 。	存储器 abs16
地址寄存器间接		
[A0] [A1]	以地址寄存器（A0/A1）的内容表示的值为运算对象的有效地址。 有效地址的范围为00000 ₁₆ ~0FFFF ₁₆ 。	寄存器 A0 / A1 address 存储器

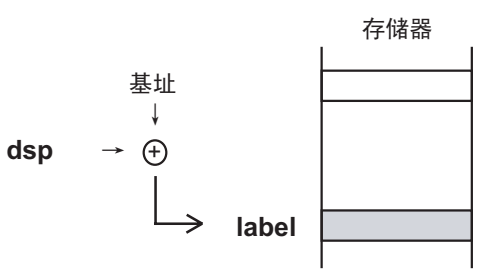
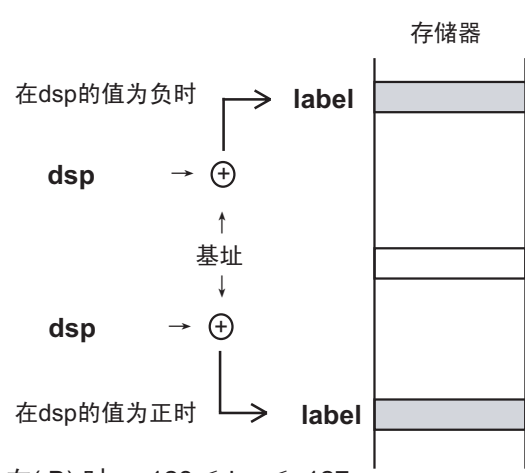
地址寄存器相对		
dsp:8[A0] dsp:8[A1] dsp:16[A0] dsp:16[A1]	将位移量 (dsp) 表示的值和地址寄存器 (A0/A1) 的内容相加 (无符号加法) 的结果为运算对象的有效地址。 但是, 如果相加的结果超过 0FFFF₁₆, 就忽视17位以上的内容, 并且返回到00000₁₆地址侧。	
SB相对		
dsp:8[SB] dsp:16[SB]	将静态基址寄存器 (SB) 的内容表示的地址和由位移量 (dsp) 表示的值相加 (无符号加法) 的结果为运算对象的有效地址。 但是, 如果相加的结果超过 0FFFF₁₆, 就忽视17位以上的内容, 并且返回到00000₁₆地址侧。	
FB相对		
dsp:8[FB]	将帧基址寄存器 (FB) 的内容表示的地址和由位移量 (dsp) 表示的值相加 (带符号加法) 的结果为运算对象的有效地址。 但是, 如果相加的结果超过 00000₁₆~0FFFF₁₆, 就忽视17位以上的内容, 并且返回到 00000₁₆地址侧或者0FFFF₁₆地址侧。	

堆栈指针相对	
dsp:8[SP]	将堆栈指针（SP）的内容表示的地址和由位移量（dsp）表示的值相加（带符号加法）的结果为运算对象的有效地址。堆栈指针（SP）以U标志指示的堆栈指针为对象。 但是，如果相加的结果超过00000₁₆~0FFFF₁₆，就忽视17位以上的内容，并且返回到00000₁₆地址侧或者0FFFF₁₆地址侧。 此寻址能用于MOV指令。

2.4 特殊指令寻址

20位绝对		
abs20	以abs20表示的值为运算对象的有效地址。 有效地址的范围为00000₁₆~FFFFF₁₆。 此寻址能用于LDE、STE、JSR、JMP指令。	
20位带位移量的地址寄存器相对		
dsp:20[A0] dsp:20[A1]	将位移量（dsp）表示的地址和地址寄存器（A0/A1）的内容相加（无符号加法）的结果为运算对象的有效地址。 但是，如果相加的结果超过FFFFF₁₆，就忽视21位以上的内容，并且返回到00000₁₆地址侧。 此寻址能用于LDE、STE、JMPL、JSRI指令。 能使用的寻址方式和指令的组合如下所示： dsp:20[A0] →LDE、STE、JMPL、JSRI指令 dsp:20[A1] →JMPL、JSRI指令	
32位地址寄存器间接		
[A1A0]	将地址寄存器（A0/A1）相连组成的32位地址为运算对象的有效地址。 但是，如果相连后的寄存器值超过FFFFF₁₆，就忽视21位以上的内容。 此寻址能用于LDE、STE指令。	

32位寄存器直接		
R2R0 R3R1 A1A0	将指定的2个寄存器相连组成的32位寄存器为运算对象。 此寻址能用于SHL、SHA、JMPI、JSRI指令。 能使用的寄存器和指令的组合如下所示： R2R0、R3R1 →SHL、SHA、JMPI、JSRI指令 A1A0 →JMPI、JSRI指令	○SHL、SHA指令 R2R0 R3R1 b31 b16 b15 b0 ○JMPI、JSRI指令 R2R0 R3R1 A1A0 b31 b16 b15 b0 ↓ PC
专用寄存器直接		
INTBL INTBH ISP SP SB FB FLG	以指定的专用寄存器为运算对象。 此寻址能用于LDC、STC、PUSHC、POPC指令。 如果指定SP，就以U标志指示的堆栈指针为对象。	寄存器 INTBL b15 b0 INTBH b15 b4 b3 b0 ISP b15 b0 USP b15 b0 SB b15 b0 FB b15 b0 FLG b15 b0

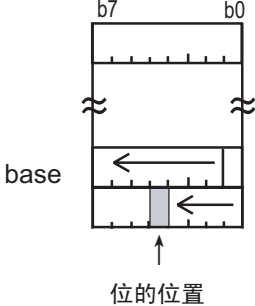
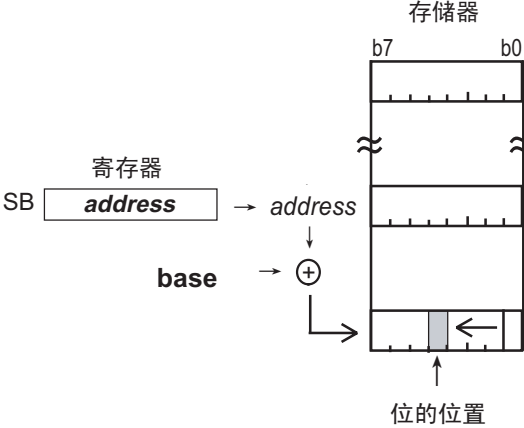
程序计数器相对	
label ○在转移距离说明符 (.length) 为 (.S) 时 将基址和由位移量 (dsp) 表示的值相加 (无符号加法) 的结果为有效地址。 此寻址能用于JMP指令。	 +0≤dsp≤+7 *1 基址为 (指令的起始地址+2)。
○在转移距离说明符 (.length) 为 (.B) 或者 (.W) 时 将基址和由位移量 (dsp) 表示的值相加 (带符号加法) 的结果为有效地址。 但是, 如果相加的结果超过 00000₁₆~FFFFF₁₆, 就忽视21位以上的数据, 并且返回到 00000₁₆地址侧或者FFFFF₁₆地址侧。 此寻址能用于JMP、JSR指令。	 在dsp的值为负时 在dsp的值为正时 在(.B) 时, -128≤dsp≤+127 在(.W) 时, -32768≤dsp≤+32767 *2 基址根据指令的不同而不同。

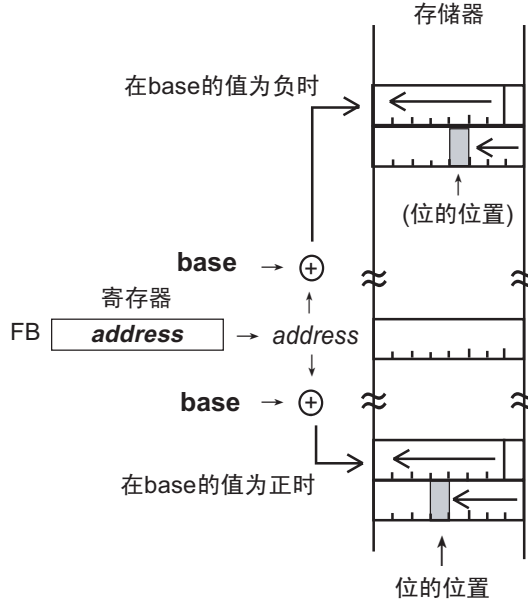
2.5 位指令寻址

此寻址能用于以下指令：

BCLR、**BSET**、**BNOT**、**BTST**、**BNTST**、**BAND**、**BNAND**、**BOR**、**BNOR**、**BXOR**、**BNXOR**、**BMCond**、**BTSTS**、**BTSTC**

寄存器直接		bit , R0
bit,R0 bit,R1 bit,R2 bit,R3 bit,A0 bit,A1	以指定的寄存器的位为运算对象。 位的位置（bit）能指定0~15。	
绝对		base
bit,base:16	以base表示的地址的位0为起点间隔bit表示的位数的位为运算对象。 以00000 ₁₆ 地址~01FFF ₁₆ 地址的位为对象。	
地址寄存器间接		00000₁₆
[A0] [A1]	以00000 ₁₆ 地址的位0为起点间隔地址寄存器（A0/A1）表示的位数的位为运算对象。 以00000 ₁₆ 地址~01FFF ₁₆ 地址的位为对象。	

地址寄存器相对 base:8[A0] base:8[A1] base:16[A0] base:16[A1]	以base表示的地址的位0为起点间隔地址寄存器（A0/A1）表示的位数的位为运算对象。 但是，如果对象位的地址超过0FFFF₁₆，就忽视17位以上的数据，并且返回到00000₁₆地址侧。 能用地址寄存器（A0/A1）指定的地址范围为从base开始的8192字节。	
SB相对 bit,base:8[SB] bit,base:11[SB] bit,base:16[SB]	将静态基址寄存器（SB）的内容表示的地址和由base表示的值相加（无符号加法），以相加后的地址的位0为起点间隔bit表示的位数的位为运算对象。 但是，如果对象位的地址超过0FFFF₁₆，就忽视17位和17位以上的数据，并且返回到00000₁₆地址侧。 能用bit,base:8、bit,base:11、bit,base:16指定的地址范围分别为从静态基址寄存器（SB）的值开始的32字节、256字节、8192字节。	

FB相对		
bit,base:8[FB]	将帧基址寄存器（FB）的内容表示的值和由base表示的值相加（带符号加法），以相加后的地址的位0为起点间隔bit表示的位数的位为运算对象。 但是，如果对象位的地址超过00000₁₆~0FFFF₁₆，就忽视17位和17位以上的数据，并且返回到00000₁₆地址侧或者0FFFF₁₆地址侧。 能用bit,base:8指定的地址范围是以帧基址寄存器（FB）的值为中心的32字节。	
FLG直接		
U I O B S Z D C	以指定的标志为运算对象。 此寻址能用于FCLR、FSET指令。	

第 3 章

功能

- 3.1 本章的阅读方法
- 3.2 功能

3.1 本章的阅读方法

在本章中，按各指令说明有关语法、操作、功能、可选择的src/dest、标志变化、记述例以及相关指令。
有关本章的阅读方法，举例说明如下：

第3章 功能

(1) **MOV**

传送

MOVE

MOV

【指令码/周期数】

Page= 195

(2) 【语法】

(3) **MOV.size (:format) src,dest**

G, Q, Z, S (可指定)

B, W

(4) 【操作】

dest ← src

(5) 【功能】

将src传送到dest

在对长度说明符(.size)指定(.B)后，如果dest为地址寄存器，就将src零扩展成16位，然后进行传送。另外，如果src为地址寄存器，就传送地址寄存器的低8位。

(6) 【可选择的src/dest】

(按格式分类的src/dest请参照下一页。)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	dsp:8[SP]	R2R0	R3R1	A1A0	dsp:8[SP]

(7) 【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S : 当传送结果为dest的MSB= “1” 时置 “1”， 否则清 “0”。

Z : 当传送结果为0时置 “1”， 否则清 “0”。

(8) 【记述例】

MOV.B:S #0ABH,R0L

MOV.W #1,R2

(9) 【相关指令】

LDE,STE,XCHG

92

(1) 助记符

表示在本页中说明的助记符。

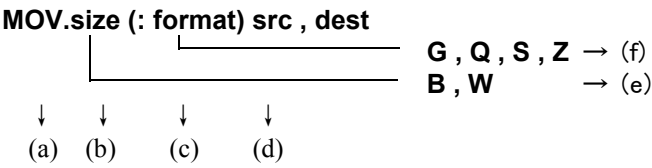
(2) 指令码/周期数

表示指令码和周期数的记载页。

有关指令码和周期数，请参照此页。

(3) 语法

用符号表示指令的语法。在省略（:format）时，汇编程序选择最适合的说明符。



(a) 助记符 **MOV**

描述助记符。

(b) 长度说明符 **.size**

描述处理的数据长度。能指定的长度如下所示：

.B 字节(8位)

.W 字(16位)

.L 长字(32位)

也存在不具有长度说明符的指令。

(c) 指令格式说明符 **(: format)**

描述指令格式。在省略（:format）时，汇编程序选择最适合的说明符。在描述（:format）后，被描述的内容优先。

能指定的指令格式如下所示：

:G 一般格式

:Q 快速格式

:S 短格式

:Z 零格式

也存在不具有指令格式说明符的指令。

(d) 操作数 **src, dest**

描述操作数。

(e) 表示能用(b)指定的长度。

(f) 表示能用(c)指定的指令格式。

第3章 功能

(1) MOV

传送

MOV

(2) 【语法】

MOV.size (:format) src,dest

【指令码/周期数】
Page= 195

(3)

G, Q, Z, S (可指定)
B, W

(4) 【操作】

dest ← src

(5) 【功能】

- 将src传送到dest
- 在对长度说明符(.size)指定(.B)后，如果dest为地址寄存器，就将src零扩展成16位，然后进行传送。另外，如果src为地址寄存器，就传送地址寄存器的低8位。

(6) 【可选择的src/dest】

(按格式分类的src/dest请参照下一页。)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	dsp:8[SP]	R2R0	R3R1	A1A0	dsp:8[SP]

(7) 【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件
S : 当传送结果为dest的MSB= “1” 时置 “1”，否则清 “0”。
Z : 当传送结果为0时置 “1”，否则清 “0”。

(8) 【记述例】

MOV.B:S #0ABH,R0L
MOV.W #-1,R2

(9) 【相关指令】

LDE,STE,XCHG

92

(4) 操作

用符号说明指令的操作。

(5) 功能

说明指令的功能和注意事项。

(6) 可选择的src/dest (label)

在指令有操作数时，表示对操作数能选择的格式。

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	#IMM
R2R0	R3R1	A1A0	dsp:8[SP]	R2R0	R3R1	A1A0	dsp:8[SP]

- (a) 表示对src(source)能选择的项目
- (b) 表示对dest(destination)能选择的项目
- (c) 表示能选择的寻址方式
- (d) 表示不能选择的寻址方式
- (e) 斜线的左侧（R0H）表示处理数据长度为字节（8位）时的寻址方式
斜线的右侧（R1）表示处理数据长度为字（16位）时的寻址方式

(7) 标志变化

表示指令执行后的标志变化。表中所示符号的意义如下：

- “—” 不变化。
- “○” 根据条件变化。

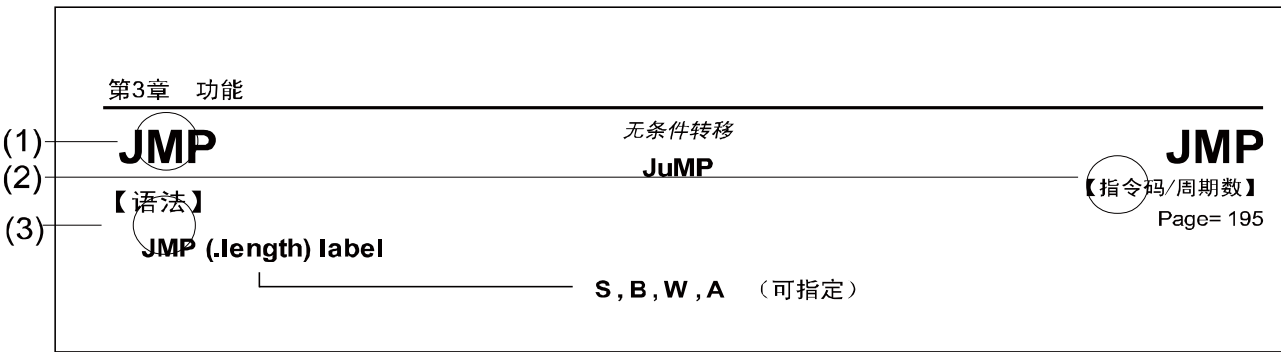
(8) 记述例

表示指令的记述例。

(9) 相关指令

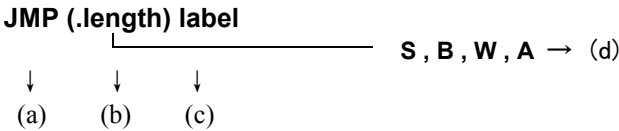
表示与此指令类似的操作或者相反的操作的有关指令。

有关JMP、JPMI、JSR、JSRI各指令的语法，举例说明如下：



(3) 语法

用符号表示指令的语法。



(a) 助记符 **JMP**

描述助记符。

(b) 转移距离说明符 **.length**

描述转移的距离。对于JMP、JSR指令，在省略(.length)时，汇编程序选择最适合的说明符；在描述(.length)后，被描述的内容优先。

能指定的转移距离如下所示：

- .S 3位PC向前相对 (+2~+9)
- .B 8位PC相对
- .W 16位PC相对
- .A 20位绝对

(c) 操作数 **label**

描述操作数。

(d) 表示能用(b)指定的转移距离。

3.2 功能

ABS

【语法】

ABS.size dest
B, W

【操作】

dest ← | dest |

【功能】

- 取dest的绝对值，结果保存到dest。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1 A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O : 当演算前的dest为-128(B)或者-32768(W)时置“1”，否则清“0”。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。
- C : 不定。

【记述例】

ABS.B R0L
ABS.W A0

绝对值
ABSolute

ABS

【指令码/周期数】

Page=140

ADC

带进位的加法演算
ADdition with Carry

ADC

【语法】

ADC.size src,dest
B, W

【指令码/周期数】

Page=140

【操作】

dest ← src + dest + C

【功能】

- 将dest、src和C标志相加，结果保存到dest。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O : 当带符号演算的结果超出+32767(.W)~-32768(.W)或者+127(.B)~-128(.B)时置“1”，否则清“0”。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。
- C : 当无符号演算的结果超出+65535(.W)或者+255(.B)时置“1”，否则清“0”。

【记述例】

ADC.B #2,R0L
ADC.W A0,R0
ADC.B A0,R0L ; 将A0的低8位和R0L进行演算。
ADC.B R0L,A0 ; 在将R0L零扩展后，和A0进行演算。

【相关指令】 ADCF,ADD,SBB,SUB

ADCF

进位标志的加法演算
ADdition Carry Flag

ADCF

【语法】
ADCF.size dest
B, W

【指令码/周期数】
Page=142

【操作】
dest ← dest + C

【功能】
• 将dest和 C 标志相加，结果保存到dest。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O : 当带符号演算的结果超出+32767(.W)~-32768(.W)或者+127(.B)~-128(.B)时置“1”，否则清“0”。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。
- C : 当无符号演算的结果超出+65535(.W)或者+255(.B)时置“1”，否则清“0”。

【记述例】

ADCF.B R0L
ADCF.W Ram:16[A0]

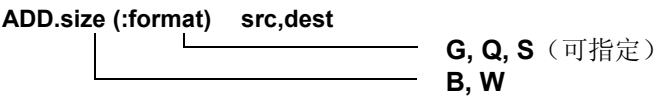
【相关指令】 ADC,ADD,SBB,SUB

ADD

无进位的加法演算
ADDition

ADD

【语法】



【指令码/周期数】

Page=142

【操作】

dest ← dest + src

【功能】

- 将dest和src相加，结果保存到dest。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。
- 在对长度说明符(.size)指定(.B)后，如果dest为堆栈指针，就将src零扩展成16位，然后进行演算。

【可选择的src/dest】

(按格式分类的src/dest请参照下一页)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP*2
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

*2 演算对象为由U标志指示的堆栈指针。对src只能选择#IMM。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O : 当带符号演算的结果超出+32767(.W)~-32768(.W)或者+127(.B)~-128(.B)时置“1”，否则清“0”。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。
- C : 当无符号演算的结果超出+65535(.W)或者+255(.B)时置“1”，否则清“0”。

【记述例】

ADD.B A0,R0L ; 将A0的低8位和R0L进行演算。
ADD.B R0L,A0 ; 在将R0L零扩展后，和A0进行演算。
ADD.B Ram:8[SB],R0L
ADD.W #2,[A0]

【相关指令】 ADC,ADCF,SBB,SUB

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP* ²
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

*2 演算对象为由U标志指示的堆栈指针。对src只能选择#IMM。

Q 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM* ³	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP* ²
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*2 演算对象为由U标志指示的堆栈指针。对src只能选择#IMM。

*3 能取得的范围为 $-8 \leq \#IMM \leq +7$ 。

S 格式*⁴

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	
R0L* ⁵	R0H* ⁵	dsp:8[SB]	dsp:8[FB]	R0L* ⁵	R0H* ⁵	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	

*4 对长度说明符(.size)只能指定(.B)。

*5 对src和dest不能选择相同的寄存器。

ADJNZ

加法演算和条件转移
ADdition then Jump on Not Zero

ADJNZ

【语法】

ADJNZ.size src,dest,label
B, W

【指令码/周期数】

Page=148

【操作】

dest ← dest + src
if dest≠0 then jump label

【功能】

- 将dest和src相加，结果保存到dest。
- 在相加后的结果不为0时，转移到label，否则执行下一条指令。
- 本指令操作码和SBJNZ相同。

【可选择的src/dest/label】

src	dest			label
#IMM*1	R0L/R0	R0H/R1	R1L/R2	$PC^{*2} - 126 \leq label \leq PC^{*2} + 129$
	R1H/R3	A0/A0	A1/A1	
	[A0]	[A1]	dsp:8[A0]	
	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	
	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	
	abs16			

*1 能取得的范围为 $-8 \leq \#IMM \leq +7$ 。

*2 PC指示指令的起始地址。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

ADJNZ.W #−1,R0,label

【相关指令】 SBJNZ

AND

逻辑与
AND

AND

【语法】

ADD.size (:format) src,dest

└──────────────────┬──────────┘

 G, S (可指定)

 B, W

【指令码/周期数】
Page=149

【操作】

dest ← src ∧ dest

【功能】

- 将dest和src进行逻辑与，结果保存到dest。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】 (按格式分类的src/dest请参照下一页。)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。

【记述例】

AND.B Ram:8[SB],R0L

AND.B:G A0,R0L ; 将A0的低8位和R0L进行演算。

AND.B:G R0L,A0 ; 在将R0L零扩展后，和A0进行演算。

AND.B:S #3,R0L

【相关指令】 OR,XOR,TST

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

S 格式*²

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	
R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]	R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	

*2 对长度说明符(.size)只能指定(.B)。

*3 对src和dest不能选择相同的寄存器。

BAND

【语法】
BAND src

【操作】
 $C \leftarrow \text{src} \wedge C$

【功能】

- 将 C 标志和src进行逻辑与，结果保存到 C 标志。

【可选择的src/dest】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件
C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BAND flag
BAND 4,Ram
BAND 16,Ram:16[SB]
BAND [A0]

【相关指令】 BOR,BXOR,BNAND,BNOR,BNXOR

位逻辑与
Bit AND carry flag

BAND

【指令码/周期数】
Page=152

BCLR

【语法】

BCLR (:format) dest

【操作】

dest ← 0

【功能】

• 给dest置“0”。

【可选择的dest】

位清除
Bit CLear

G, S (可指定)

BCLR

【指令码/周期数】

Page=152

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
○	bit, base:11[SB] ^{*1}		

^{*1} 只有在 S 格式时才能选择的dest。

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

BCLR flag

BCLR 4,Ram:8[SB]

BCLR 16,Ram:16[SB]

BCLR [A0]

【相关指令】

BSET,BNOT,BNTST,BTST,BTSTC,BTSTS

BM*Cnd*

【语法】

BM*Cnd* dest

【操作】

if true then dest ← 1
else dest ← 0

【功能】

- 将由*Cnd*表示的条件的真假值传送到dest。真时传送“1”，假时传送“0”。
- *Cnd*有以下的种类。

<i>Cnd</i>	条件		式	<i>Cnd</i>	条件		式
GEU/C	C=1	大于等于/C标志为“1”	≤	LTU/NC	C=0	小于/C标志为“0”	>
EQ/Z	Z=1	等于/Z标志为“1”	=	NE/NZ	Z=0	不等于/Z标志为“0”	≠
GTU	$C \wedge \bar{Z}=1$	大于	<	LEU	$C \wedge \bar{Z}=0$	小于等于	≥
PZ	S=0	正或者零	$0 \leq$	N	S=1	负	$0 >$
GE	$S \vee O=0$	等于或者带符号大于	≤	LE	$(S \vee O) \vee Z=1$	等于或者带符号小于	≥
GT	$(S \vee O) \vee Z=0$	带符号大于	<	LT	$S \vee O=1$	带符号小于	>
O	O=1	O标志为“1”		NO	O=0	O标志为“0”	

【可选的dest】

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	*1

*1 在对dest指定C标志时，变化。

【记述例】

BMN 3,Ram:8[SB]
BMZ C

【相关指令】 J*Cnd*

BM*Cnd*

【指令码/周期数】

Page=154

BNAND

【语法】

BNAND src

【操作】

$$C \leftarrow \overline{\text{src}} \wedge C$$

【功能】

- 将 C 标志和src的非进行逻辑与演算，结果保存到 C 标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件

C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BNAND flag
BNAND 4,Ram
BNAND 16,Ram:16[SB]
BNAND [A0]

【相关指令】 BAND,BOR,BXOR,BNOR,BNXOR

反转位的逻辑与
Bit Not AND carry flag

BNAND

【指令码/周期数】

Page=155

BNOR

【语法】

BNOR src

【操作】

$C \leftarrow \overline{\text{src}} \vee C$

【功能】

- 将C标志和src的非进行逻辑或演算，结果保存到C标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件

C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BNOR flag
BNOR 4,Ram
BNOR 16,Ram:16[SB]
BNOR [A0]

【相关指令】 BAND,BOR,BXOR,BNAND,BNXOR

反转位的逻辑或
Bit Not OR carry flag

BNOR

【指令码/周期数】

Page=156

BNOT

【语法】

BNOT(:format) dest
G, S (可指定)

【操作】

dest ← $\overline{\text{dest}}$

【功能】

- 将dest的非保存到dest。

【可选择的dest】

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
$\odot$	bit, base:11[SB] ^{*1}		

*1 只有在 S 格式时才能选择的dest。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

BNOT flag
BNOT 4,Ram:8[SB]
BNOT 16,Ram:16[SB]
BNOT [A0]

【相关指令】 BCLR,BSET,BNTST,BTST,BTSTC,BTSTS

BNTST

【语法】

BNTST src

【操作】

Z ← $\overline{\text{src}}$
C ← src

【功能】

- 将src的非传送到Z标志和C标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	○	—	○

条件

- Z : 当src为“0”时置“1”，否则清“0”。
- C : 当src为“0”时置“1”，否则清“0”。

【记述例】

BNTST flag
BNTST 4,Ram:8[SB]
BNTST 16,Ram:16[SB]
BNTST [A0]

【相关指令】 BCLR,BSET,BNOT,BTST,BTSTC,BTSTS

反转位的测试
Bit Not TeST

BNTST

【指令码/周期数】

Page=157

BNXOR

【语法】

BNXOR src

【操作】

$$C \leftarrow \overline{\text{src}} \vee C$$

【功能】

- 将C标志和src的非进行逻辑异或，结果保存到C标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件

C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BNXOR flag
BNXOR 4,Ram
BNXOR 16,Ram:16[SB]
BNXOR [A0]

【相关指令】 BAND,BOR,BXOR,BNAND,BNOR

反转位的逻辑异或
Bit Not eXclusive OR carry flag

BNXOR

【指令码/周期数】

Page=158

BOR

【语法】

BOR src

【操作】

C ← src ∨ C

【功能】

- 将 C 标志和src进行逻辑或演算，结果保存到C标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件

C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BOR flag
BOR 4,Ram
BOR 16,Ram:16[SB]
BOR [A0]

【相关指令】 BAND,BXOR,BNAND,BNOR,BNXOR

位逻辑或
Bit OR carry flag

BOR

【指令码/周期数】

Page=158

BRK

【语法】

BRK

【操作】

SP ← SP - 2
M(SP) ← (PC +1)_H,FLG
SP ← SP - 2
M(SP) ← (PC +1)_{ML}
PC ← M(FFFE₄₁₆)

【功能】

- 产生BRK中断。
- BRK中断为非屏蔽中断。

【标志变化】^{*1}

标志	U	I	O	B	S	Z	D	C
变化	○	○	—	—	—	—	○	—

条件

- U : 变为“0”。
- I : 变为“0”。
- D : 变为“0”。

【记述例】

BRK

【相关指令】 INT,INTO

调试中断
BReaK

BRK

【指令码/周期数】

Page=159

^{*1} 将BRK指令执行前的标志保存到堆栈区，中断后的标志变化如左图。

BSET

【语法】

BSET (:format) dest

置位
Bit SET

G, S (可指定)

BSET

【指令码/周期数】

Page=159

【操作】

dest ← 1

【功能】

- 给dest置“1”。

【可选择的dest】

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
Ⓞ	bit, base:11[SB]*1		

*1 只有在 S 格式时才能选择的dest。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

BSET flag
BSET 4,Ram:8[SB]
BSET 16,Ram:16[SB]
BSET [A0]

【相关指令】 BCLR,BNOT,BNTST,BTST,BTSTC,BTSTS

BTST

【语法】

BTST (:format) src
G, S (可指定)

【操作】

Z ← src
C ← src

【功能】

- 将src的非传送到 Z 标志，并且将src传送到 C 标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit, base:11[SB] ^{*1}		

*1 只有在 S 格式时才能选择的src。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	○	—	○

条件

- Z : 当src为 “0” 时置 “1” ， 否则清 “0” 。
- C : 当src为 “1” 时置 “1” ， 否则清 “0” 。

【记述例】

BTST flag
BTST 4,Ram:8[SB]
BTST 16,Ram:16[SB]
BTST [A0]

【相关指令】 BCLR,BSET,BNOT,BNTST,BTSTC,BTSTS

位测试
Bit TeST

BTST

【指令码/周期数】

Page=160

BTSTC

【语法】
BTSTC dest

【操作】
Z ← dest
C ← dest
dest ← 0

【功能】
• 将dest的非传送到 Z 标志，并且将dest传送到 C 标志，然后给dest置 “0” 。

【可选择的dest】

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	○	—	○

条件
Z : 当dest为 “0” 时置 “1” ， 否则清 “0” 。
C : 当dest为 “1” 时置 “1” ， 否则清 “0” 。

【记述例】
BTSTC flag
BTSTC 4,Ram
BTSTC 16,Ram:16[SB]
BTSTC [A0]

【相关指令】 BCLR,BSET,BNOT,BNTST,BTST,BTSTS

位测试和清除
Bit TeST & Clear

BTSTC
【指令码/周期数】
Page=161

BTSTS

【语法】

BTSTS dest

【操作】

Z ← $\overline{\text{dest}}$
C ← dest
dest ← 1

【功能】

- 将dest的非传送到 Z 标志，并且将dest传送到 C 标志，然后给dest置 “1” 。

【可选择的dest】

dest			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit,base:8[SB]	bit,base:8[FB]
base:16[A0]	base:16[A1]	bit,base:16[SB]	bit,base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	○	—	○

条件

- Z : 当dest为 “0” 时置 “1” ， 否则清 “0” 。
- C : 当dest为 “1” 时置 “1” ， 否则清 “0” 。

【记述例】

BTSTS flag
BTSTS 4,Ram
BTSTS 16,Ram:16[SB]
BTSTS [A0]

【相关指令】 BCLR,BSET,BNOT,BNTST,BTST,BTSTC

位测试和置位
Bit TeST & Set

BTSTS

【指令码/周期数】

Page=162

BXOR

【语法】
BXOR src

【操作】
 $C \leftarrow \text{src} \vee C$

【功能】

- 将 C 标志和src进行逻辑异或演算，结果保存到 C 标志。

【可选择的src】

src			
bit,R0	bit,R1	bit,R2	bit,R3
bit,A0	bit,A1	[A0]	[A1]
base:8[A0]	base:8[A1]	bit, base:8[SB]	bit, base:8[FB]
base:16[A0]	base:16[A1]	bit, base:16[SB]	bit, base:16
C	bit,base:11[SB]		

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	○

条件
C：当演算结果为“1”时置“1”，否则清“0”。

【记述例】

BXOR flag
BXOR 4,Ram
BXOR 16,Ram:16[SB]
BXOR [A0]

【相关指令】 BAND,BOR,BNAND,BNOR,BNXOR

位逻辑异或
Bit eXclusive OR carry flag

BXOR

【指令码/周期数】
Page=162

CMP

【语法】

CMP.size (:format)

src,dest

【操作】

dest ← src

比较
CoMPare

G, Q, S (可指定)
B, W

CMP

【指令码/周期数】

Page=163

- 【功能】
- 根据从dest减去src的结果，改变标志寄存器的各标志。
 - 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】(按格式分类的src/dest请参照下一页)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O : 当带符号的演算结果超出+32767(.W)～-32768(.W)或者+127(.B)～-128(.B)时置“1”，否则清“0”。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为“0”时置“1”，否则清“0”。
- C : 当无符号的演算结果大于等于0时置“1”，否则清“0”。

【记述例】

CMP.B:S #10,R0L

CMP.W:G R0,A0

CMP.W #-3,R0

CMP.B #5,Ram:8[FB]

CMP.B A0,R0L

；将A0的低8位和R0L比较。

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

Q 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM* ²	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*2 能取得的范围为 $-8 \leq \text{#IMM} \leq +7$ 。

S 格式*³

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	
R0L* ⁴	R0H* ⁴	dsp:8[SB]	dsp:8[FB]	R0L* ⁴	R0H* ⁴	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	

*3 对长度说明符(.size)只能指定(.B)。

*4 对src和dest不能选择相同的寄存器。

DADC

【语法】

DADC.size src,dest

B, W

带进位的10进制加法演算

Decimal ADDition with Carry

DADC

【指令码/周期数】

Page=167

【操作】

dest ← src + dest + C

【功能】

- 将dest、src和C标志以10进制方式相加，结果保存到dest。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	○

条件

S ：当演算结果的MSB为“1”时置“1”，否则清“0”。

Z ：当演算结果为0时置“1”，否则清“0”。

C ：当无符号的演算结果超出+9999(W)或者+99(B)时置“1”，否则清“0”。

【记述例】

DADC.B #3,R0L

DADC.W R1,R0

【相关指令】 DADD,DSUB,DSBB

DEC

【语法】

DEC.size

dest

DECrement

B, W

DEC

【指令码/周期数】

Page=171

【操作】

dest ← dest − 1

【功能】

- 从dest减去 1，结果保存到dest。

【可选择的dest】

dest			
R0L ^{*1}	R0H ^{*1}	dsp:8[SB] ^{*1}	dsp:8[FB] ^{*1}
abs16 ^{*1}	A0 ^{*2}	A1 ^{*2}	

*1 对长度说明符(.size)只能指定(.B)。

*2 对长度说明符(.size)只能指定(.W)。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S：当演算结果的MSB为“1”时置“1”，否则清“0”。。

Z：当演算结果为0时置“1”，否则清“0”。

【记述例】

DEC.W A0
DEC.B R0L

【相关指令】 INC

DIV

【语法】

DIV.size src

带符号的除法演算

B, W

DIV

【指令码/周期数】

Page=172

【操作】

在长度说明符(.size)为(.B)时

$$R0L(\text{商})、R0H(\text{余数}) \leftarrow R0 \div \text{src}$$

在长度说明符(.size)为(.W)时

$$R0(\text{商})、R2(\text{余数}) \leftarrow R2R0 \div \text{src}$$

【功能】

- 将R2R0(R0)*1除以带符号的src。商保存到R0(R0L)*1、余数保存到R2(R0H)*1。余数的符号和被除数的符号相同。(0)*1内为对长度说明符(.size)指定(.B)时的情况。
- 在对长度说明符(.size)指定(.B)后，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。
- 在对长度说明符(.size)指定(.B)后，如果演算结果的商超出8位或者除数为0，O标志就变为“1”。此时，R0L和R0H不定。
- 在对长度说明符(.size)指定(.W)后，如果演算结果的商超出16位或者除数为0，O标志就变为“1”。此时，R0和R2不定。

【可选择的src】

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM
R2/R0	R3/R1	A1/A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	—	—	—	—

条件

O : 当演算结果的商超出16位(W)或8位(B)、或者除数为“0”时置“1”，否则清“0”。

【记述例】

DIV.B A0 ; A0的低8位为除数。

DIV.B #4

DIV.W R0

【相关指令】 DIVU,DIVX,MUL,MULU

DIVU

无符号的除法演算
DIVide Unsigned

DIVU

【语法】

DIVU.size src
B, W

【指令码/周期数】

Page=173

【操作】

在长度说明符(.size)为(.B)时
R0L(商)、R0H(余数) $\leftarrow R0 \div src$
在长度说明符(.size)为(.W)时
R0(商)、R2(余数) $\leftarrow R2R0 \div src$

【功能】

- 将R2R0(R0)*1除以不带符号的src。商保存到R0(R0L)*1、余数保存到R2(R0H)*1。()*1内为对长度说明符(.size)指定(.B)时的情况。
- 在对长度说明符(.size)指定(.B)后，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。
- 在对长度说明符(.size)指定(.B)后，如果演算结果的商超出8位或者除数为0，O标志就变为“1”。此时，R0L和R0H不定。
- 在对长度说明符(.size)指定(.W)后，如果演算结果的商超出16位或者除数为0，O标志就变为“1”。此时，R0和R2不定。

【可选择的src】

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	—	—	—	—

条件

O：当演算结果的商超出16位(.W)或8位(.B)、或者除数为“0”时置“1”，否则清“0”。

【记述例】

DIVU.B A0 ; A0的低8位为除数。
DIVU.B #4
DIVU.W R0

【相关指令】 DIV,DIVX,MUL,MULU

DIVX

带符号的除法演算
DIVide eXtension

DIVX

【语法】

DIVX.size src
B, W

【指令码/周期数】

Page=174

【操作】

在长度说明符(.size)为(.B)时
R0L(商)、R0H(余数) $\leftarrow R0 \div src$
在长度说明符(.size)为(.W)时
R0(商)、R2(余数) $\leftarrow R2R0 \div src$

【功能】

- 将R2R0(R0)*1除以带符号的src。商保存到R0(R0L)*1、余数保存到R2(R0H)*1。余数的符号和除数的符号相同。()*1内为对长度说明符(.size)指定(.B)时的情况。
- 在对长度说明符(.size)指定(.B)后，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。
- 在对长度说明符(.size)指定(.B)后，如果演算结果的商超出8位或者除数为0，O标志就变为“1”。此时，R0L和R0H不定。
- 在对长度说明符(.size)指定(.W)后，如果演算结果的商超出16位或者除数为0，O标志就变为“1”。此时，R0和R2不定。

【可选择的src】

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	—	—	—	—

条件
O：当演算结果的商超出16位(.W)或8位(.B)、或者除数为“0”时置“1”，否则清“0”。

【记述例】

DIVX.B A0 ; A0的低8位为除数。
DIVX.B #4
DIVX.W R0

【相关指令】 DIV,DIVU,MUL,MULU

DSBB

【语法】

DSBB.size **src,dest**

B, W

【操作】

$$\text{dest} \leftarrow \text{dest} - \text{src} - \overline{C}$$

【功能】

- 以10进制方式从dest减去src和C标志的非，结果保存到dest。

【可选的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	○

条件

- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。
- C : 当演算结果大于等于0时置“1”，否则清“0”。

【记述例】

DSBB.B #3,R0L

DSBB.W R1,R0

【相关指令】 DADC,DADD,DSUB

带借位的10进制减法演算 Decimal SuBtract with Borrow

DSBB

【指令码/周期数】

Page=175

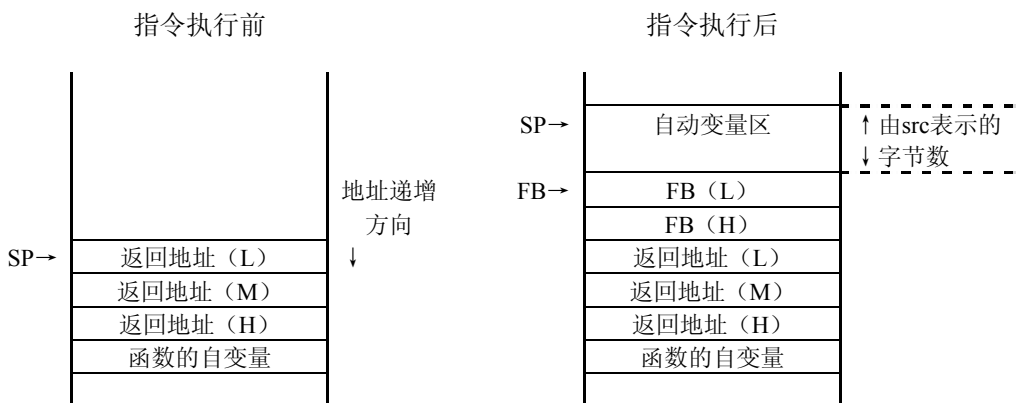
ENTER

【语法】
ENTER src

【操作】
SP ← SP - 2
M(SP) ← FB
FB ← SP
SP ← SP - src

【功能】

- 产生栈帧。src为栈帧的长度。
- 调用子程序后，在调用子程序的起始位置执行ENTER指令前后的堆栈区状态如下所示：



【可选择的src】

src
#IMM8

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
ENTER #3

【相关指令】 EXITD

栈帧的建立 ENTER function

ENTER

【指令码/周期数】
Page=179

EXITD

【语法】
EXITD

栈帧的释放
EXIT and Deallocate stack frame

EXITD

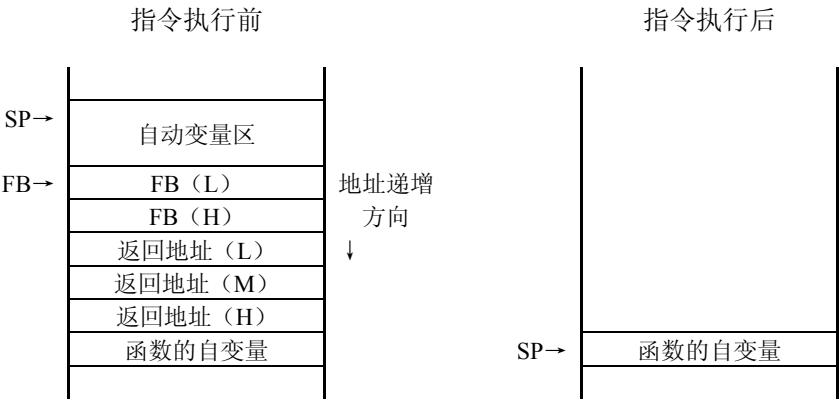
【指令码/周期数】
Page=180

【操作】

SP ← FB
FB ← M(SP)
SP ← SP + 2
PCML ← M(SP)
SP ← SP + 2
PCH ← M(SP)
SP ← SP + 1

【功能】

- 进行栈帧的释放和从子程序的返回。
- 本指令必须和ENTER指令配对使用。
- 在执行ENTER指令后的子程序的最后，执行EXITD指令前后的堆栈区状态如下所示：



【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
EXITD

【相关指令】 ENTER

EXTS

【语法】

EXTS.size dest
B, W

【操作】

dest ← EXT(dest)

【功能】

- 对dest进行符号扩展，结果保存到dest。
- 如果对长度说明符(.size)指定(.B)，就将dest符号扩展成16位。
- 如果对长度说明符(.size)指定(.W)，就将R0符号扩展成32位。此时，对高位字节使用R2。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

- S : 在对长度说明符(.size)指定(.B)后，如果演算结果的MSB为“1”，就置“1”，否则清“0”。在对长度说明符(.size)指定(.W)时，不变化。
- Z : 在对长度说明符(.size)指定(.B)后，如果演算结果为0，就置“1”，否则清“0”。在对长度说明符(.size)指定(.W)时，不变化。

【记述例】

EXTS.B R0L
EXTS.W R0

符号扩展
EXTend Sign

EXTS

【指令码/周期数】

Page=180

FCLR

【语法】
FCLR dest

【操作】
dest ← 0

【功能】
• 给dest置“0”。

【可选择的dest】

dest							
C	D	Z	S	B	O	I	U

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	*1	*1	*1	*1	*1	*1	*1	*1

*1 选择的标志变为“0”。

【记述例】
FCLR I
FCLR S

【相关指令】 FSET

标志寄存器的位清除
Flag register CLear

FCLR

【指令码/周期数】
Page=181

FSET

【语法】
FSET dest

【操作】
dest ← 1

【功能】
• 给dest置“1”。

【可选择的dest】

dest							
C	D	Z	S	B	O	I	U

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	*1	*1	*1	*1	*1	*1	*1	*1

*1 选择的标志变为“1”。

【记述例】
FSET I
FSET S

【相关指令】 FCLR

标志寄存器的置位 Flag register SET

FSET

【指令码/周期数】
Page=182

INC

【语法】

INC.size

dest

【操作】

dest ← dest + 1

【功能】

• 给dest加 1，结果保存到dest。

【可选择的dest】

递增

INCRement

B, W

INC

【指令码/周期数】

Page=182

dest			
R0L ^{*1}	R0H ^{*1}	dsp:8[SB] ^{*1}	dsp:8[FB] ^{*1}
abs16 ^{*1}	A0 ^{*2}	A1 ^{*2}	

*1 对长度说明符(.size)只能指定(.B)。

*2 对长度说明符(.size)只能指定(.W)。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S：当演算结果的MSB为“1”时置“1”，否则清“0”。

Z：当演算结果为0时置“1”，否则清“0”。

【记述例】

INC.W A0

INC.B R0L

【相关指令】 DEC

INT

【语法】
INT src

INT指令中断
INTerrupt

INT

【指令码/周期数】
Page=183

【操作】
SP ← SP - 2
M(SP) ← (PC + 2)_H,FLG
SP ← SP - 2
M(SP) ← (PC + 2)_{ML}
PC ← M(IntBase +src × 4)

【功能】

- 产生由src指定的软件中断。src为软件中断号。
- 在src小于等于31时，U标志变为“0”，使用ISP。
- 在src大于等于32时，使用U标志指示的堆栈指针。
- 通过INT指令产生的中断为非屏蔽中断。

【可选择的src】

src
#IMM*1*2

*1 #IMM为软件中断号。
*2 能取得的范围为0≤#IMM≤63。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	○	○	—	—	—	—	○	—

条件
U : 当软件中断号为31以下时清“0”。当软件中断号为32以上时不变化。
I : 变为“0”。
D : 变为“0”。

*3 将INT指令执行前的标志保存到堆栈区，中断后的标志变化如左图。

【记述例】
INT #0

【相关指令】 BRK,INTO

INTO

【语法】
INTO

溢出中断
INTerrupt on Overflow

INTO

【指令码/周期数】
Page=184

【操作】

SP ← SP - 2
M(SP) ← (PC + 1)_H, FLG
SP ← SP - 2
M(SP) ← (PC + 1)_{ML}
PC ← M(FFFE₀₁₆)

【功能】

- 在 O 标志为 “1” 时，产生溢出中断；在 O 标志为 “0”，执行下一条指令。
- 溢出中断为非屏蔽中断。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	○	○	—	—	—	—	○	—

条件

- U : 变为 “0”。
- I : 变为 “0”。
- D : 变为 “0”。

*1 将INTO指令执行前的标志保存到堆栈区，中断后的标志变化如左图。

【记述例】

INTO

【相关指令】 BRK,INT

JCnd

条件转移
Jump on Condition

JCnd

【语法】

JCnd label

【指令码/周期数】

Page=184

【操作】

if true then jump label

【功能】

- 按如下条件判断前一条指令的执行结果，进行转移。如果由Cnd表示的条件为真，就转移到label。如果为假，就执行下一条指令。
- Cnd有以下的种类：

<i>Cnd</i>	条件		式	<i>Cnd</i>	条件		式
GEU/C	C=1	大于等于/C标志为“1”	≤	LTU/NC	C=0	小于/C标志为“0”	>
EQ/Z	Z=1	等于/Z标志为“1”	=	NE/NZ	Z=0	不等于/Z标志为“0”	≠
GTU	$C \wedge \overline{Z}=1$	大于	<	LEU	$C \wedge \overline{Z}=0$	小于等于	≥
PZ	S=0	正或者零	$0 \leq$	N	S=1	负	$0 >$
GE	$S \vee O=0$	等于或者带符号大于	≤	LE	$(S \vee O) \vee Z=1$	等于或者带符号小于	≥
GT	$(S \vee O) \vee Z=0$	带符号大于	<	LT	$S \vee O=1$	带符号小于	>
O	O=1	O标志为“1”		NO	O=0	O标志为“0”	

【可选择的label】

label	Cnd
$PC^*1-127 \leq label \leq PC^*1+128$	GEU/C,GTU,EQ/Z,N,LTU/NC,LEU,NE/NZ,PZ
$PC^*1-126 \leq label \leq PC^*1+129$	LE,O,GE,GT,NO,LT

*1 PC指示指令的起始地址。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

JEQ label
JNE label

【相关指令】 BM Cnd

JMP

无条件转移
JuMP

JMP

【语法】
JMP(.length) label

【指令码/周期数】
Page=185

S, B, W, A（可指定）

【操作】
PC ← label

【功能】
• 转移到label。

【可选择的label】

.length	label
.S	$PC^{*1}+2 \leq label \leq PC^{*1}+9$
.B	$PC^{*1}-127 \leq label \leq PC^{*1}+128$
.W	$PC^{*1}-32767 \leq label \leq PC^{*1}+32768$
.A	abs20

*1 PC指示指令的起始地址。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
JMP label

【相关指令】 JMPL, JMPS

JMPI

【语法】

JMPI.length src

W, A

间接转移

JuMP Indirect

JMPI

【指令码/周期数】

Page=187

【操作】

在转移距离说明符(.length)为(.W)时

PC ← PC ± src

在转移距离说明符(.length)为(.A)时

PC ← src

【功能】

- 转移到src指示的地址。当src为存储器时，必须指定低地址的保存地址。
- 在对转移距离说明符(.length)指定(.W)时，转移到将指令的起始地址和src相加（带符号）的地址。另外，当src为存储器时，所需的存储容量为2字节。
- 在对转移距离说明符(.length)指定(.A)后，如果src为存储器，所需的存储容量就为3字节。

【可选择的src】

在对转移距离说明符(.length)指定(.W)

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

在对转移距离说明符(.length)指定(.A)时

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

JMPI.A A1A0

JMPI.W R0

【相关指令】 JMP,JMPS

JMPS

【语法】
JMPS src

【操作】

PCH ← 0F16
PCML ← M(FFFFE16—src×2)

【功能】

- 转移到专用页向量表的各表中设定的地址与F000016相加的地址。可转移的区为F000016地址到FFFFF16地址。
- 专用页向量表被分配在从FFE0016到FFFDA16地址的区。
- src是专用页号。专用页号的FFE0016地址是255，FFFDA16地址是18。

【可选择的src】

src
#IMM*1*2

- *1 #IMM是专用页号。
*2 取值范围为18≤#IMM≤255。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

JMPS #20

【相关指令】 JMP,JMPI

专用页转移
JuMP Special page

JMPS

【指令码/周期数】
Page=188

JSR

子程序调用
Jump SubRoutine

JSR

【语法】

JSR(.length) label
_____ W, A (可指定)

【指令码/周期数】

Page=189

【操作】

SP ← SP - 1
M(SP) ← (PC +n)H
SP ← SP - 2
M(SP) ← (PC +n)ML
PC ← label
*1 n为指令的字节数。

【功能】

- 向label进行子程序转移。

【可选择的label】

.length	label
.W	$PC^{*1}-32767 \leq label \leq PC^{*1}+32768$
.A	abs20

*1 PC指示指令的起始地址。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

JSR.W func
JSR.A func

【相关指令】 JSRI,JSRS

JSRI

间接子程序调用 Jump SubRoutine Indirect

JSRI

【语法】

JSRI.length src
W, A

【指令码/周期数】

Page=190

【操作】

在转移距离说明符(.length)为(.W)时

SP ← SP - 1
M(SP) ← (PC + n)H
SP ← SP - 2
M(SP) ← (PC + n)ML
PC ← PC ± src
*1 n为指令的字节数。

在转移距离说明符(.length)为(.A)时

SP ← SP - 1
M(SP) ← (PC + n)H
SP ← SP - 2
M(SP) ← (PC + n)ML
PC ← src

【功能】

- 向src指示的地址进行子程序转移。当src为存储器时，必须指定低地址的保存地址。
- 在对转移距离说明符(.length)指定(.W)时，向指令的起始地址和src相加（带符号）的地址进行子程序转移。另外，当src为存储器时，所需的存储容量为2字节。
- 在对转移距离说明符(.length)指定(.A)后，如果src为存储器，所需的存储容量就为3字节。

【可选择的src】

在对转移距离说明符(.length)指定(.W)时

src			
R0L R0	R0H/R1	R1L R2	R1H/R3
A0 A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

在对转移距离说明符(.length)指定(.A)时

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

JSRI.A A1A0
JSRI.W R0

【相关指令】 JSR,JSRS

JSRS

专用页子程序调用
Jump SubRoutine Special
page

JSRS

【语法】
JSRS src

【指令码/周期数】
Page=191

【操作】

SP ← SP - 1
M(SP) ← (PC + 2)H
SP ← SP - 2
M(SP) ← (PC + 2)ML
PCH ← 0F16
PCML ← M(FFFFE16 - src × 2)

【功能】

- 转移到专用页向量表的各表中设定的地址与F000016相加的地址。可转移的区为F000016地址到FFFFF16地址。
- 专用页向量表被分配在从FFE0016到FFFDA16地址的区。
- src是专用页号。专用页号的FFE0016地址是255，FFFDA16地址是18。

【可选择的src】

src
#IMM*1*2

*1 #IMM是专用页号。
*2 取值范围为18 ≤ #IMM ≤ 255。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
JSRS #18

【相关指令】 JSR,JSRI

LDC

【语法】

LDC src, dest

【操作】

dest ← src

【功能】

- 将src传送到dest表示的专用寄存器。当src为存储器时，所需的存储容量为2字节。
- 在传送到INTBL或者INTBH时，必须连续传送。
- 在此指令后，不能立即响应中断请求。

【可选择的src/dest】

src				dest			
R0L R0	R0H/R1	R1L R2	R1H/R3	FB	SB	SP*1	ISP
A0 A0	A1/A1	[A0]	[A1]	FLG	INTBH	INTBL	
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]				
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16				
dsp:20[A0]	dsp:20[A1]	abs20	#IMM				
R2R0	R3R1	A1A0					

*1 以U标志指示的堆栈指针为对象。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	*2	*2	*2	*2	*2	*2	*2	*2

*2 只有当dest为FLG时变化。

【记述例】

LDC R0,SB
LDC A0,FB

【相关指令】 POPC,PUSHC,STC,LDINTB

向专用寄存器的传送
LoaD Control register

LDC

【指令码/周期数】

Page=191

LDCTX

【语法】

LDCTX **abs16,abs20**

环境恢复

LDCTX

【指令码/周期数】

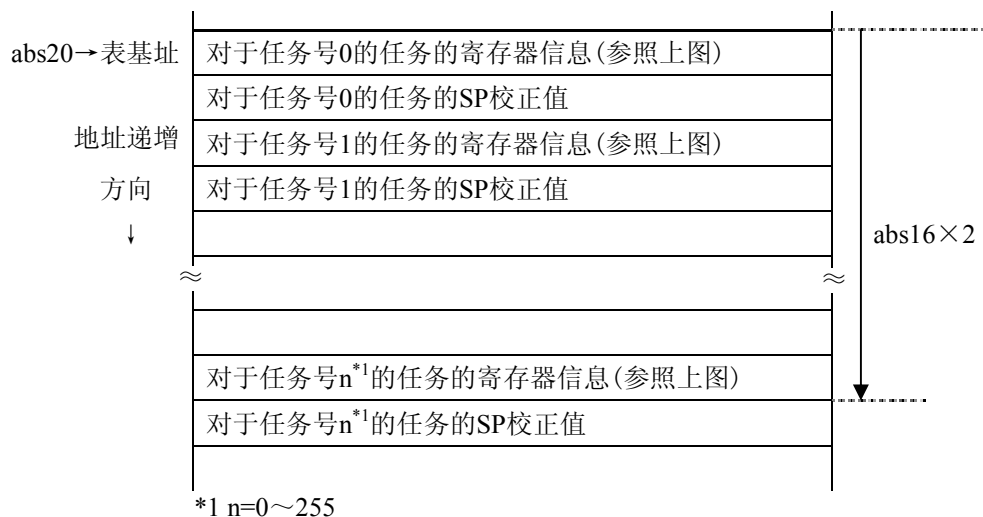
Page=192

【功能】

- 将任务的环境从堆栈区恢复。
- 必须给abs16设定保存任务号的RAM地址，并且给abs20设定数据表的起始地址。
- 根据任务号，从数据表中指定所需的寄存器信息，按照此寄存器信息将堆栈区的数据传送到各寄存器。然后，对堆栈指针（SP）加上SP的校正值。必须给SP的校正值设定传送的寄存器的字节数。
- 传送的寄存器信息的构成如下。用“1”表示传送的寄存器，用“0”表示不传送的寄存器。



- 数据表的构成如下。用abs20表示的地址为表基址，在离基址2倍abs16的内容间隔的地址中保存的数据表示寄存器信息，下一个地址表示堆栈指针的校正值。



*1 n=0~255

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

LDCTX Ram,Rom_TBL

【相关指令】 STCTX

LDINTB

【语法】
LDINTB src

【操作】
INTBHL ← src

【功能】

- 将src传送到INTB。
- LDINTB指令是由以下指令构成的宏指令。
LDC #IMM,INTBH
LDC #IMM,INTBL

【可选择的src】

src
#IMM20

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
LDINTB #0F0000H

【相关指令】 LDC,STC,PUSHC,POPC

向INTB寄存器的传送
LoaD INTB register

LDINTB

【指令码/周期数】
Page=194

LDIPL

【语法】
LDIPL src

【操作】
IPL ← src

【功能】
• 将src传送到IPL。

【可选择的src】

src
#IMM*1

*1 能取得的范围为 $0 \leq \text{#IMM} \leq 7$ 。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
LDIPL #2

中断允许级的设定
LoaD Interrupt Permission Level

LDIPL

【指令码/周期数】
Page=195

MOV

【语法】

MOV.size (:format) src,dest

【操作】

dest ← src

传送
MOVE

G, Q, Z, S (可指定)
B, W

MOV

【指令码/周期数】

Page=195

- 【功能】
- 将src传送到dest。
 - 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行传送。另外，如果src为A0或者A1，就传送A0或者A1的低8位。

【可选择的src/dest】(按格式分类的src/dest请参照下一页。)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM*2	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	dsp:8[SP]*3	R2R0	R3R1	A1A0	dsp:8[SP]*2*3

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

*2 在src为#IMM时，对dest不能选择dsp:8[SP]。

*3 演算对象为由U标志指示的堆栈指针。另外，对src和dest不能同时选择dsp:8[SP]。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S

: 当传送结果的dest的MSB为“1”时置“1”，否则清“0”。

Z

: 当传送结果为0时置“1”，否则清“0”。

【记述例】

MOV.B:S

#0ABH,R0L

MOV.W

#-1,R2

【相关指令】 LDE,STE,XCHG

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM* ²	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0	dsp:8[SP]* ³	R2R0	R3R1	A1A0	dsp:8[SP]* ^{2*3}

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

*2 在src为#IMM时，对dest不能选择dsp:8[SP]。

*3 演算对象为由U标志指示的堆栈指针。另外，对src和dest不能同时选择dsp:8[SP]。

Q 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	dsp:16[FB]	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM* ⁴	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*4 能取得的范围为 $-8 \leq \text{#IMM} \leq 7$ 。

S 格式

src				dest			
R0L* ^{5*6*7}	R0H* ^{5*6*8}	dsp:8[SB]* ⁵	dsp:8[FB]* ⁵	R0L* ^{5*6}	R0H* ^{5*6}		
abs16* ⁵	#IMM			A0* ^{5*8}	A1* ^{5*7}		
R0L* ^{5*6}	R0H* ^{5*6}	dsp:8[SB]	dsp:8[FB]	R0L* ^{5*6}	R0H* ^{5*6}	dsp:8[SB]* ⁵	dsp:8[FB]* ⁵
abs16	#IMM			abs16* ⁵	A0	A1	
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L* ⁵	R0H* ⁵	dsp:8[SB]* ⁵	dsp:8[FB]* ⁵
abs16	#IMM* ⁹			abs16* ⁵	A0* ⁹	A1* ⁹	

*5 对长度说明符(.size)只能指定(.B)。

*6 对src和dest不能选择相同的寄存器。

*7 在src为R0L时，作为地址寄存器，对dest只能选择A1。

*8 在src为R0H时，作为地址寄存器，对dest只能选择A0。

*9 对长度说明符(.size)能指定(.B)和(.W)。

Z 格式

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#0			abs16	A0	A1	

MOVA

【语法】
MOVA src,dest

【操作】
dest ← EVA(src)

【功能】
• 将src的有效地址传送到dest。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
MOVA Ram:16[SB],A0

【相关指令】 PUSHA

有效地址的传送
MOVE effective Address

MOVA

【指令码/周期数】
Page=202

MOVDir

【语法】
MOVDir src,dest

4 位数据的传送
MOVE nibble

MOVDir

【指令码/周期数】
Page=203

【操作】

Dir	操作
HH	H4:dest ← H4:src
HL	L4:dest ← H4:src
LH	H4:dest ← L4:src
LL	L4:dest ← L4:src

【功能】

- 给src或者dest的任何一个选择R0L。

Dir	功能
HH	将src的高4位传送到dest的高4位
HL	将src的高4位传送到dest的低4位
LH	将src的低4位传送到dest的高4位
LL	将src的低4位传送到dest的低4位

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R2
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	
R0L/R0	R0H/R1	R1L	R1H	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

MOVHH R0L,[A0]
MOVHL R0L,[A0]

MUL

【语法】

MOV_L.size src,dest
B, W

【操作】

dest ← dest × src

【功能】

- 将src和dest进行带符号乘法演算，结果保存到dest。
- 如果对长度说明符(.size)指定(.B)，src和dest都以8位演算，结果以16位保存。如果对src或者dest中的任何一个指定A0或者A1，就演算A0或者A1的低8位内容。
- 如果对长度说明符(.size)指定(.W)，src和dest都以16位演算，结果以32位保存。如果给dest选择R0、R1或者A0，结果就被分别保存到R2R0、R3R1或者A1A0。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

MUL.B A0,R0L ; 将R0L和A0的低8位进行演算。
MUL.W #3,R0
MUL.B R0L,R1L
MUL.W A0,Ram

【相关指令】 DIV,DIVU,DIVX,MULU

带符号的乘法演算
MULTiple

MUL

【指令码/周期数】

Page=205

MULU

无符号的乘法演算
MULTiple Unsigned

MULU

【语法】

MULU.size src,dest
B, W

【指令码/周期数】

Page=207

【操作】

dest ← dest × src

【功能】

- 将src和dest进行无符号乘法演算，结果保存到dest。
- 如果对长度说明符(.size)指定(.B)，src和dest都以8位演算，结果以16位保存。如果对src或者dest中的任何一个选择A0或者A1，就演算A0或者A1的低8位内容。
- 如果对长度说明符(.size)指定(.W)，src和dest都以16位演算，结果以32位保存。如果对dest选择R0、R1或者A0，结果就被分别保存到R2R0、R3R1或者A1A0。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

MULU.B A0,R0L ; 将R0L和A0的低8位进行演算。
MULU.W #3,R0
MULU.B R0L,R1L
MULU.W A0,Ram

【相关指令】 DIV,DIVU,DIVX,MUL

NEG

【语法】

NEG.size dest

B, W

2的补码
NEGate

NEG

【指令码/周期数】

Page=209

【操作】

dest ← 0 − dest

【功能】

- 取dest的2的补码，结果保存到dest。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0, A0	A1, A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O：当演算前的dest为−128(.B)或者−32768(.W)时置“1”，否则清“0”。
- S：当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z：当演算结果为0时置“1”，否则清“0”。
- C：当演算结果为0时置“1”，否则清“0”。

【记述例】

NEG.B R0L
NEG.W A1

【相关指令】 NOT

NOP

【语法】
NOP

空操作
No OPeration

NOP

【指令码/周期数】
Page=209

【操作】
 $PC \leftarrow PC + 1$

【功能】
• 给PC加 1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
NOP

NOT

【语法】

NOT.size (:format) dest

【操作】

dest ← $\overline{\text{dest}}$

【功能】

- 将dest的非保存到dest。

【可选择的dest】

全位反转

NOT

【指令码/周期数】

Page=210

G, S (可指定)

B, W

NOT

【指令码/周期数】

Page=210

dest			
R0L ^{*1} /R0	R0H ^{*1} /R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB] ^{*1}	dsp:8[FB] ^{*1}
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16 ^{*1}
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

^{*1} 能在G格式和S格式中选择。其它的dest能在G格式中选择。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S : 当演算结果的MSB为“1”时置“1”，否则清“0”。

Z : 当演算结果为0时置“1”，否则清“0”。

【记述例】

NOT.B R0L

NOT.W A1

【相关指令】

NEG

OR

【语法】

OR.size (:format) src,dest

逻辑或

OR

G, S (可指定)

B, W

OR

【指令码/周期数】

Page=211

【操作】

dest ← src ∨ dest

- 【功能】
- 将dest和src进行逻辑或演算，结果保存到dest。
 - 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】(按格式分类的src/dest请参照下一页。)

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

- 条件
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。

【记述例】

OR.B Ram:8[SB],R0L ; 将A0的低8位和R0L进行演算。

OR.B:G A0,R0L ; 在将R0L零扩展后，和A0进行演算。

OR.B:G R0L,A0

OR.B:S #3,R0L

【相关指令】 AND,XOR,TST

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

S 格式*²

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	
R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]	R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]
abs16				abs16	A0	A1	

*2 对长度说明符(.size)只能指定(.B)。

*3 对src和dest不能选择相同的寄存器。

POPC

【语法】
POPC dest

【操作】
dest ← M(SP)
SP*1 ← SP + 2
*1 如果dest为SP或者在U标志为“0”的状态下dest为ISP，SP就不被加2。

【功能】

- 从堆栈区恢复到由dest表示的专用寄存器。
- 在恢复中断表寄存器时，必须连续恢复INTBH和INTBL。
- 在此指令后，不能立即响应中断请求。

【可选择的dest】

dest						
FB	SB	SP*2	ISP	FLG	INTBH	INTBL

*2 以 U 标志指示的堆栈指针为对象。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	*3	*3	*3	*3	*3	*3	*3	*3

*3 只有当dest为FLG时变化。

【记述例】
POPC SB

【相关指令】 PUSHC,LDC,STC,LDINTB

POP Control register

POPC

【指令码/周期数】
Page=215

POPM

【语法】
POPM dest

【操作】
dest ← M(SP)
SP ← SP + N^{*1} × 2
*1 恢复的寄存器数。

【功能】

- 将由dest选择的寄存器全部从堆栈区恢复。
- 以如下的优先顺序从堆栈区恢复：



【可选择的dest】

Dest ^{*2}							
R0	R1	R2	R3	A0	A1	SB	FB

*2 能选择多个dest。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
POPM R0,R1,A0,SB,FB

【相关指令】 POP,PUSH,PUSHM

多个寄存器的恢复 POP Multiple

POPM

【指令码/周期数】
Page=215

PUSH

寄存器/存储器/立即数的保存

PUSH

【语法】

PUSH.size (:format) src

G, S (可指定)
B, W

【指令码/周期数】
Page=216

【操作】

在长度说明符(.size)为(.B)时

SP ← SP - 1

M(SP) ← src

在长度说明符(.size)为(.W)时

SP ← SP - 2

M(SP) ← src

【功能】

- 将src保存到堆栈区。

【可选择的src】

src			
R0L ^{*1} /R0	R0H ^{*1} /R1	R1L/R2	R1H/R3
A0.A0 ^{*1}	A1.A1 ^{*1}	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM
R2R0	R3R1	A1A0	

*1 能在G格式和S格式中选择。其它的src能在G格式中选择。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

PUSH.B #5
PUSH.W #100H
PUSH.B R0L
PUSH.W A0

【相关指令】

POP,POPM,PUSHM

PUSHA

【语法】
PUSHA src

【操作】
SP ← SP - 2
M(SP) ← EVA(src)

【功能】
• 将src的有效地址保存到堆栈区。

【可选择的src】

src			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
PUSHA Ram:8[FB]
PUSHA Ram:16[SB]

【相关指令】 MOVA

有效地址的保存
PUSH effective Address

PUSHA

【指令码/周期数】
Page=218

PUSHC

【语法】
PUSHC src

【操作】

SP ← SP - 2
M(SP) ← src *1

*1 如果src为SP或者在U标志为“0”的状态下src为ISP，就将减2前的SP保存。

【功能】

- 将由src表示的专用寄存器保存到堆栈区。

【可选择的src】

src						
FB	SB	SP*2	ISP	FLG	INTBH	INTBL

*2 以U标志指示的堆栈指针为对象。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

PUSHC SB

【相关指令】 POPC,LDC,STC,LDINTB

专用寄存器的保存
PUSH Control register

PUSHC

【指令码/周期数】
Page=218

PUSHM

【语法】

PUSHM **src**

【操作】

$SP \leftarrow SP - N^{*1} \times 2$

$M(SP) \leftarrow src$

*1 保存的寄存器数。

【功能】

- 将由src选择的寄存器全部保存到堆栈区。
- 以如下的优先顺序保存到堆栈区：



【可选择的src】

src ^{*2}							
R0	R1	R2	R3	A0	A1	SB	FB

*2 能选择多个src。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

PUSHM R0,R1,A0,SB,FB

【相关指令】 POP,PUSH,POPM

多个寄存器的保存
PUSH Multiple

PUSHM

【指令码/周期数】

Page=219

REIT

【语法】
REIT

从中断的返回
REturn from InTerrupt

REIT

【指令码/周期数】
Page=219

【操作】
PCML ← M(SP)
SP ← SP + 2
PCH,FLG ← M(SP)
SP ← SP + 2

【功能】
• 在响应中断请求时，恢复被保存的PC和FLG，从中断程序返回。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	*1	*1	*1	*1	*1	*1	*1	*1

*1 返回到响应中断请求前的FLG状态。

【记述例】
REIT

RMPA

乘加演算 Repeat MultiPle & Addition

RMPA

【语法】

RMPA.size

B, W

【指令码/周期数】

Page=220

【操作】*1

Repeat

 $R2R0(R0)^{*2} \leftarrow R2R0(R0)^{*2} + M(A0) \times M(A1)$ $A0 \leftarrow A0 + 2(1)^{*2}$ $A1 \leftarrow A1 + 2(1)^{*2}$ $R3 \leftarrow R3 - 1$

Until R3 = 0

*1 在给R3设定0后执行时，本指令被忽视。

*2 ()^{*2}内为对长度说明符(.size)指定(.B)的情况。

【功能】

- 将A0作为被乘数地址、将A1作为乘数地址、将R3作为演算次数，进行乘加演算。演算以带符号进行，结果保存到R2R0(R0)^{*1}。
- 如果在演算中溢出，O标志就变为“1”，结束演算。在R2R0(R0)^{*1}中保存最后的加法演算结果。A0、A1以及R3为不定。
- 指令结束时的A0或者A1的内容表示最后读取数据的下一个地址。
- 如果在指令执行中发生中断请求，就在乘加演算的加法演算结束后(在R3的内容被减1后)接受中断。
- 必须给R2R0(R0)^{*1}设定初始值。
- *1 ()^{*1}内为对长度说明符(.size)指定(.B)的情况。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	—	—	—	—

条件

O：在演算中超出+2147483647(.W)~−2147483648(.W)或者+32767(.B)~−32768(.B)时置“1”，否则清“0”。

【记述例】

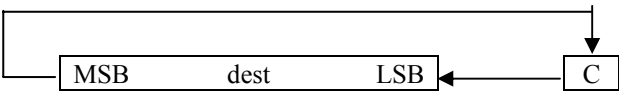
RMPA.B

ROLC

【语法】

ROLC.size dest
B, W

【操作】



【功能】

- 包含 C 标志将dest循环左移1位。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	○

条件

- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果的dest为0时置“1”，否则清“0”。
- C : 当移出的位为“1”时置“1”，否则清“0”。

【记述例】

ROLC.B R0L
ROLC.W R0

【相关指令】 RORC,ROT,SHA,SHL

带进位的循环左移
ROTate to Left with Carry

ROLC

【指令码/周期数】

Page=220

RORC

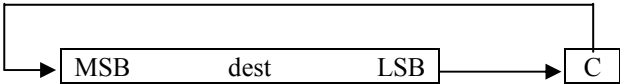
带进位的循环右移
ROtate to Right with Carry

RORC

【语法】
RORC.size dest
B, W

【指令码/周期数】
Page=221

【操作】



【功能】

- 包含 C 标志将dest循环右移1位。

【可选择的dest】

dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	○

条件

- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果的dest为0时置“1”，否则清“0”。
- C : 当移出的位为“1”时置“1”，否则清“0”。

【记述例】

RORC.B R0L
RORC.W R0

【相关指令】 ROLC,ROT,SHA,SHL

ROT

【语法】

ROT.size src,dest

循环移位 ROTate

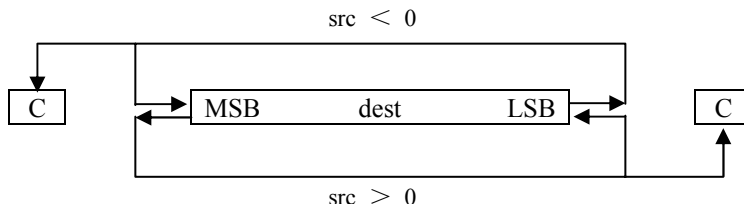
B, W

ROT

【指令码/周期数】

Page=222

【操作】



【功能】

- 将dest进行循环移位，循环的位数由src表示。从LSB(MSB)溢出的位被传送到MSB(LSB)和C标志。
- 循环方向由src的符号指定。当src为正时向左循环移位，为负时向右循环移位。
- 如果src为立即，循环的位数就为-8~-1和+1~+8。不能设定成-9以下、0或者+9以上。
- 在src为寄存器时，如果对长度说明符(.size)指定(B)，循环的位数就为-8~+8。能设定成0，但是不循环移位。另外，标志寄存器的各标志也不变化。如果设定成-9以下或者+9以上，循环移位的结果不定。
- 在src为寄存器时，如果对长度说明符(.size)指定(W)，循环的位数就为-16~+16。能设定成0，但是不循环移位。另外，标志寄存器的各标志也不变化。如果设定成-17以下或者+17以上，循环移位的结果不定。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H ^{*1}	R0L/R0	R0H/R1 ^{*1}	R1L/R2	R1H/R3 ^{*1}
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM ^{*2}	dsp:20[A0]	dsp:20[A1]	abs20	
R2/R0	R3/R1	A1/A0		R2/R0	R3/R1	A1/A0	

*1 在src为R1H时，不能给dest选择R1或者R1H。

*2 能取得的范围为 $-8 \leq \#IMM \leq +8$ 。但是，不能设定成0。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	○

*1 当循环位数为0时标志不变化。

条件

S₁ : 当演算结果为MSB等于“1”时置“1”, 否则清“0”。

Z : 当演算结果为0时置“1”，否则清“0”。

C : 当最后移出的位为“1”时置“1”，否则清“0”。

【记述例】

ROT.B #1,R0L ; 左循环

ROT.B	#-1,R0L	: 右循环
-------	---------	-------

ROT.W R1H,R2

【相关指令】 ROLC,RORC,SHA,SHL

RTS

【语法】
RTS

从子程序的返回
ReTurn from Subroutine

RTS

【指令码/周期数】
Page=223

【操作】

PCML ← M(SP)
SP ← SP + 2
PCH ← M(SP)
SP ← SP + 1

【功能】

- 从子程序返回。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
RTS

SBB

带借位的减法运算
SuBtract with Borrow

SBB

【语法】

SBB.size src,dest
B, W

【指令码/周期数】

Page=224

【操作】

dest ← dest - src - $\overline{C}$

【功能】

- 从dest减去src和C标志的非，结果保存到dest。
- 在对长度说明符(.size)指定(.B)时，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	○	—	○	○	—	○

条件

- O：当带符号演算的结果超出+32767(.W)～-32768(.W)或者+127(.B)～-128(.B)时置“1”，否则清“0”。
- S：当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z：当演算结果为0时置“1”，否则清“0”。
- C：当无符号演算的结果大于等于0时置“1”，否则清“0”。

【记述例】

SBB.B #2,R0L
SBB.W A0,R0
SBB.B A0,R0L ; 将A0的低8位和R0L进行演算。
SBB.B R0L,A0 ; 在将R0L零扩展后，和A0进行演算。

【相关指令】 ADC,ADCF,ADD,SUB

SBJNZ

【语法】

SBJNZ.size

src,dest,label

B, W

减法演算和条件转移

SuBtract then Jump on Not Zero

SBJNZ

【指令码/周期数】

Page=226

【操作】

dest ← dest − src

if dest ≠0 then jump label

- 【功能】
- 从dest减去src，结果保存到dest。
 - 在相减后的结果不为0时，转移到label；否则执行下一条指令。
 - 本指令的操作码和ADJNZ相同。

【可选择的src/dest/label】

src	dest			label
#IMM*1	R0L/R0	R0H/R1	R1L/R2	$PC^{*2} - 126 \leq label \leq PC^{*2} + 129$
	R1H/R3	A0/A0	A1/A1	
	[A0]	[A1]	dsp:8[A0]	
	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	
	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	
	abs16			

*1 能取得的范围为−7≤#IMM≤+8。

*2 PC指示指令的起始地址。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

SBJNZ.W #1,R0,label

【相关指令】 ADJNZ

SHA

【语法】

SHA.size src,dest

算术移位
SHift Arithmetic

SHA

【指令码/周期数】

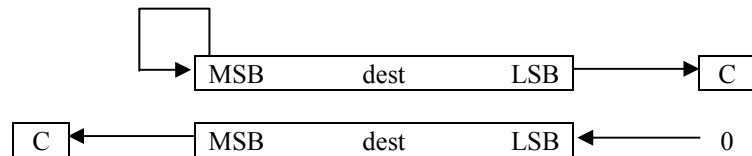
Page=227

B, W, L

【操作】

当src < 0时

当src > 0时



【功能】

- 将dest进行算术移位，移位的位数由src表示。从LSB(MSB)溢出的位被传送到C标志。
- 移位方向由src的符号指定。当src为正时向左移位，为负时向右移位。
- 如果src为立即，移位的位数就为-8~-1和+1~+8。不能设定成-9以下、0或者+9以上。
- 在src为寄存器时，如果对长度说明符(.size)指定(.B)，移位的位数就为-8~-+8。能设定成0，但是不移位。另外，标志寄存器的各标志也不变化。如果设定成-9以下或者+9以上，移位的结果不定。
- 在src为寄存器时，如果对长度说明符(.size)指定(.W)或者(.L)，移位的位数就为-16~-+16。能设定成0，但是不移位。另外，标志寄存器的各标志也不变化。如果设定成-17以下或者+17以上，移位的结果不定。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H ^{*1} /R3	R0L/R0	R0H/R1 ^{*1}	R1L/R2	R1H/R3 ^{*1}
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM ^{*2}	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0 ^{*3}	R3R1 ^{*3}	A1A0	

*1 在src为R1H时，不能对dest选择R1或者R1H。

*2 能取得的范围为-8≤#IMM≤+8。但是，不能设定成0。

*3 对长度说明符(.size)只能指定(.L)。对其它的dest能指定(.B)或者(.W)。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	-	-	○	-	○	○	-	○

*1 当移位的位数为0时标志不变化。

条件

- O : 当演算结果为MSB从“1”变化到“0”、或者从“0”变化到“1”时置“1”，否则清“0”。但是，在对长度说明符(.size)指定(.L)时，不变化。
- S : 当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。但是，在对长度说明符(.size)指定(.L)时，不定。
- C : 当最后移出的位为“1”时置“1”，否则清“0”。但是，在对长度说明符(.size)指定(.L)时，不定。

【记述例】

SHA.B #3,R0L ; 算术左位
SHA.B #-3,R0L ; 算术右位
SHA.L R1H,R2R0

【相关指令】 ROLC,RORC,ROT,SHL

SHL

逻辑移位
SHift Logical

SHL

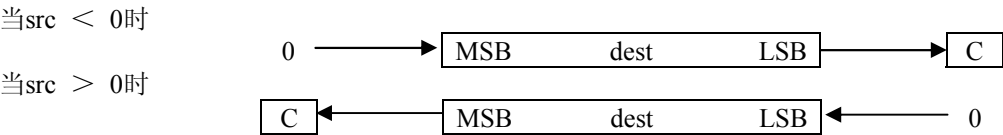
【语法】

SHL.size src,dest
B, W, L

【指令码/周期数】

Page=230

【操作】



【功能】

- 将dest进行逻辑移位，移位的位数由src表示。从LSB(MSB)溢出的位被传送到C标志。
- 移位方向由src的符号指定。当src为正时向左移位，为负时向右移位。
- 如果src为立即，移位的位数就为-8~-1和+1~+8。不能设定成-9以下、0或者+9以上。
- 在src为寄存器时，如果对长度说明符(.size)指定(.B)，移位的位数就为-8~+8。能设定成0，但是不移位。另外，标志寄存器的各标志也不变化。如果设定成-9以下或者+9以上，移位的结果不定。
- 在src为寄存器时，如果对长度说明符(.size)指定(.W)或者(.L)，移位的位数就为-16~+16。能设定成0，但是不移位。另外，标志寄存器的各标志也不变化。如果设定成-17以下或者+17以上，移位的结果不定。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H ^{*1} /R3	R0L/R0	R0H/R1 ^{*1}	R1L/R2	R1H/R3 ^{*1}
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM ^{*2}	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0 ^{*3}	R3R1 ^{*3}	A1A0	

- *1 在src为R1H时，不能对dest选择R1或者R1H。
- *2 能取得的范围为-8≤#IMM≤+8。但是，不能设定成0。
- *3 对长度说明符(.size)只能指定(.L)。对其它的dest能指定(.B)或者(.W)。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	-	-	-	-	○	○	-	○

*1 当移位的位数为0时，标志不变化。

条件

- S：当演算结果的MSB为“1”时置“1”，否则清“0”。
- Z：当演算结果为0时置“1”，否则清“0”。但是，在对长度说明符(.size)指定(.L)时，不定。
- C：当最后移出的位为“1”时置“1”，否则清“0”。但是，在对长度说明符(.size)指定(.L)时，不定。

【记述例】

SHL.B #3,R0L ; 逻辑左位

SHL.B #-3,R0L ; 逻辑右位

SHL.L R1H,R2R0

【相关指令】 ROLC,RORC,ROT,SHA

SMOVB

反向字符串传送
String MOVe Backward

SMOVB

【语法】

SMOVB.size
_____ B, W

【指令码/周期数】

Page=232

【操作】

在长度说明符(.size)为(.B)时

Repeat

$M(A1) \leftarrow M(2^{16} \times R1H + A0)$
 $A0^{*2} \leftarrow A0 - 1$
 $A1 \leftarrow A1 - 1$
 $R3 \leftarrow R3 - 1$

Until R3 =0

- *1 在给R3设定0后执行时，本指令被忽视。
- *2 当A0下溢时，R1H的内容被减1。

在长度说明符(.size)为(.W)时

Repeat

$M(A1) \leftarrow M(2^{16} \times R1H + A0)$
 $A0^{*2} \leftarrow A0 - 2$
 $A1 \leftarrow A1 - 2$
 $R3 \leftarrow R3 - 1$

Until R3 =0

【功能】

- 从由20位表示的传送源地址到由16位表示的传送目标地址，向地址递减方向进行字符串传送。
- 在R1H中设定传送源地址的高4位，在A0中设定传送源地址的低16位，在A1中设定传送目标地址，在R3中设定传送次数。
- 指令结束时的A0或者A1表示最后读取数据的下一个地址。
- 如果在指令执行中发生中断请求，就在1个数据传送结束后接受中断。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

SMOVB.B

【相关指令】 SMOVF,SSTR

SMOVF

正向字符串传送
String MOVE Forward

SMOVF

【语法】

SMOVF.size
_____ B, W

【指令码/周期数】

Page=233

【操作】

在长度说明符(.size)为(.B)时

Repeat

$M(A1) \leftarrow M(2^{16} \times R1H + A0)$

$A0^{*2} \leftarrow A0 + 1$

$A1 \leftarrow A1 + 1$

$R3 \leftarrow R3 - 1$

Until R3 =0

在长度说明符(.size)为(.W)时

Repeat

$M(A1) \leftarrow M(2^{16} \times R1H + A0)$

$A0^{*2} \leftarrow A0 + 2$

$A1 \leftarrow A1 + 2$

$R3 \leftarrow R3 - 1$

Until R3 =0

*1 在给R3设定0后执行时，本指令被忽视。

*2 当A0溢出时，R1H的内容被加1。

【功能】

- 从由20位表示的传送源地址到由16位表示的传送目标地址，向地址递增方向进行字符串传送。
- 在R1H中设定传送源地址的高4位，在A0中设定传送源地址的低16位，在A1中设定传送目标地址，在R3中设定传送次数。
- 指令结束时的A0或者A1表示最后读取数据的下一个地址。
- 如果在指令执行中发生中断请求，就在 1 个数据传送结束后接受中断。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

SMOVF.W

【相关指令】 SMOVB,SSTR

SSTR

【语法】

SSTR.size
B, W

字符串保存
String SToRe

SSTR

【指令码/周期数】

Page=233

【操作】*1

在长度说明符(.size)为(.B)时

Repeat
M(A1) ← R0L
A1 ← A1 +1
R3 ← R3 - 1
Until R3 =0

在长度说明符(.size)为(.W)时

Repeat
M(A1) ← R0
A1 ← A1 +2
R3 ← R3 - 1
Until R3 =0

*1 在给R3设定0后执行时，本指令被忽视。

【功能】

- 将R0作为保存的数据，将A1作为传送的地址，将R3作为传送次数，进行字符串保存。
- 指令结束时的A0或者A1的内容表示最后写入数据的下一个地址。
- 如果在指令执行中发生中断请求，就在 1 个数据传送结束后接受中断。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

SSTR.B

【相关指令】 SMOVB,SMOVF

STC

【语法】
STC src, dest

从专用寄存器的传送
STore from Control register

STC

【指令码/周期数】
Page=234

【操作】
dest ← src

【功能】

- 给dest传送用src表示的专用寄存器。当dest为存储器时，必须指定低地址的保存地址。
- 当dest为存储器时，如果src为PC，所需的存储容量为3字节；如果src不为PC，所需的存储容量为2字节。

【可选择的src/dest】

src				dest			
FB	SB	SP*1	ISP	R0L/R0	R0H/R1	R1L/R2	R1H/R3
FLG	INTBH	INTBL		A0/A0	A1/A1	[A0]	[A1]
				dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
				dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
				dsp:20[A0]	dsp:20[A1]	abs20	
				R2R0	R3R1	A1A0	
PC				R0L/R0	R0H/R1	R1L/R2	R1H/R3
				A0/A0	A1/A1	[A0]	[A1]
				dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
				dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
				dsp:20[A0]	dsp:20[A1]	abs20	
				R2R0	R3R1	A1A0	

*1 以U标志指示的堆栈指针为对象。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

STC SB,R0
STC FB,A0

【相关指令】 POPC,PUSHC,LDC,LDINTB

STCTX

【语法】
STCTX abs16,abs20

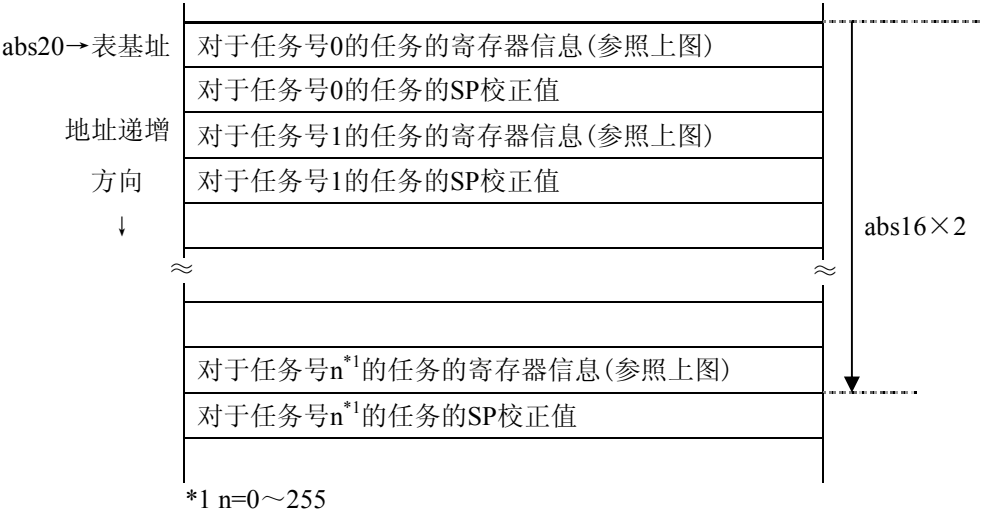
【操作】

【功能】

- 将任务的环境保存到堆栈区。
- 必须给abs16设定保存任务号的RAM地址，并且给abs20设定数据表的起始地址。
- 根据任务号，从数据表中指定所需的寄存器信息，按照此寄存器信息将各寄存器传送到堆栈区。然后，从堆栈指针（SP）减去SP的校正值。必须给SP的校正值设定传送的寄存器的字节数。
- 传送的寄存器信息的构成如下。用“1”表示传送的寄存器，用“0”表示不传送的寄存器。



- 数据表的构成如下。以abs20表示的地址为表基址，在离基址2倍abs16的内容间隔的地址中保存的数据表示寄存器信息，下一个地址表示堆栈指针的校正值。



*1 n=0~255

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

STCTX Ram,Rom_TBL

【相关指令】 LDCTX

环境保存
STore ConTeXt

STCTX

【指令码/周期数】
Page=235

STNZ

【语法】

STNZ src,dest

【操作】

if Z =0 then dest ← src

【功能】

- 当 Z 标志为 “0” 时，将src传送到dest。

【可选择的src/dest】

src	dest			
#IMM8	R0L abs16	R0H A0	dsp:8[SB] A1	dsp:8[FB]

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

STNZ #5,Ram:8[SB]

【相关指令】 STZ,STZX

条件传送
STore on Not Zero

STNZ

【指令码/周期数】

Page=237

STZ

【语法】
STZ src,dest

条件传送
STore on Zero

STZ

【指令码/周期数】
Page=237

【操作】
if Z =1 then dest ← src

【功能】
• 当 Z 标志为 “1” 时，将src传送到dest。

【可选择的src/dest】

src	dest			
#IMM8	R0L abs16	R0H A0	dsp:8[SB] A1	dsp:8[FB]

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
STZ #5,Ram:8[SB]

【相关指令】 STNZ,STZX

STZX

【语法】

STZX src1,src2,dest

【操作】

If Z=1 then
dest ← src1
else
dest ← src2

【功能】

- 当Z标志为“1”时，将src1传送到dest；当Z标志为“0”时，将src2传送到dest。

【可选择的src/dest】

src	dest			
#IMM8	R0L abs16	R0H A0	dsp:8[SB] A1	dsp:8[FB]

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

STZX #1,#2,Ram:8[SB]

【相关指令】 STZ,STNZ

条件传送
STore on Zero eXtention

STZX

【指令码/周期数】

Page=238

【按格式分类的src/dest】

G 格式

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0* ¹	A1/A1* ¹	[A0]	[A1]	A0/A0* ¹	A1/A1* ¹	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	SP/SP
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，不能对src和dest同时选择A0或者A1。

S 格式*²

src				dest			
R0L	R0H	dsp:8[SB]	dsp:8[FB]	R0L	R0H	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	
R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]	R0L* ³	R0H* ³	dsp:8[SB]	dsp:8[FB]
abs16	#IMM			abs16	A0	A1	

*2 对长度说明符(.size)只能指定(.B)。

*3 对src和dest不能选择相同的寄存器。

TST

【语法】

TST.size **src,dest**

测试
TeST

B, W

TST

【指令码/周期数】

Page=241

【操作】

$$\text{dest} \wedge \text{src}$$

【功能】

- 通过src和dest的逻辑与结果，改变标志寄存器的各标志。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0 ^{*1}	A1/A1 ^{*1}	[A0]	[A1]	A0/A0 ^{*1}	A1/A1 ^{*1}	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时, 对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

S : 当演算结果的MSB为“1”时置“1”，否则清“0”。

Z : 当演算结果为0时置“1”，否则清“0”。

【记述例】

TST.B #3,R0L

TST.B A0,R0L

；将A0的低8位和ROL进行演算。

TST.B R0L,A0

；在将ROL零扩展后，和A0进行演算。

【相关指令】 AND,OR,XOR

UND

【语法】
UND

未定义指令中断
UNDefined instruction

UND

【指令码/周期数】
Page=243

【操作】

SP ← SP - 2
M(SP) ← (PC +1) H,FLG
SP ← SP - 2
M(SP) ← (PC +1) ML
PC ← M(FFFDC16)

【功能】

- 产生未定义指令中断。
- 未定义指令中断为非屏蔽中断。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	○	○	—	—	—	—	○	—

条件

- U :变为“0”。
I :变为“0”。
D :变为“0”。

*1 将UND指令执行前的标志保存到堆栈区，中断后的标志变化如左图。

【记述例】

UND

WAIT

【语法】
WAIT

等待
WAIT

WAIT

【指令码/周期数】
Page=243

【操作】

【功能】

- 停止程序的执行。如果接受高于IPL优先级的中断或者产生复位，就开始程序的执行。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
WAIT

XCHG

【语法】

XCHG.size src,dest
B, W

【操作】

dest $\longleftrightarrow$ src

【功能】

- 交换src和dest的内容。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展后的16位数据存入A0或者A1，将A0或者A1的低8位存入src。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0	A1/A1	[A0]	[A1]	A0/A0	A1/A1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0	[A1A0]	R2R0	R3R1	A1A0	

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】

XCHG.B R0L,A0 ; 将对A0的低 8 位和R0L零扩展后的值进行交换。
XCHG.W R0,A1
XCHG.B R0L,[A0]

【相关指令】 MOV,LDE,STE

交换 eXCHanGe

XCHG

【指令码/周期数】

Page=244

XOR

逻辑异或
eXclusive OR

XOR

【语法】

XOR.size src,dest
B, W

【指令码/周期数】

Page=245

【操作】

dest ← dest ∨ src

【功能】

- 将src和dest进行逻辑异或演算，结果保存到dest。
- 在对长度说明符(.size)指定(.B)后，如果dest为A0或者A1，就将src零扩展成16位，然后进行演算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为演算对象。

【可选择的src/dest】

src				dest			
R0L/R0	R0H/R1	R1L/R2	R1H/R3	R0L/R0	R0H/R1	R1L/R2	R1H/R3
A0/A0*1	A1/A1*1	[A0]	[A1]	A0/A0*1	A1/A1*1	[A0]	[A1]
dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]	dsp:8[A0]	dsp:8[A1]	dsp:8[SB]	dsp:8[FB]
dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16	dsp:16[A0]	dsp:16[A1]	dsp:16[SB]	abs16
dsp:20[A0]	dsp:20[A1]	abs20	#IMM	dsp:20[A0]	dsp:20[A1]	abs20	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*1 在对长度说明符(.size)指定(.B)时，对src和dest不能同时选择A0或者A1。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	○	○	—	—

条件

- S : 当演算结果为MSB等于“1”时置“1”，否则清“0”。
- Z : 当演算结果为0时置“1”，否则清“0”。

【记述例】

XOR.B A0,R0L ; 将A0的低8位和R0L进行演算。
XOR.B R0L,A0 ; 在将R0L零扩展后，和A0进行演算。
XOR.B #3,R0L
XOR.W A0,A1

【相关指令】 AND,OR,TST

第 4 章

指令码/周期数

- 4.1 本章的阅读方法
- 4.2 指令码/周期数

4.1 本章的阅读方法

本章将说明每一个操作码的指令码和周期数。
有关本章的阅读方法，举例说明如下：

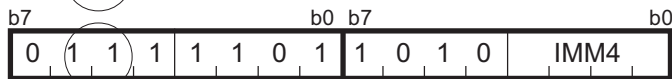
①

LDIPL

②

(1) LDIPL #IMM

③



④

【字节数/周期数】

字节数/周期数	2/2
---------	-----

①

MOV

②

(1) MOV.size:G #IMM, dest

③



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0000	dsp:8[An]	dsp:8[A0]	1000
	R0H/R1	0001		dsp:8[A1]	1001
	R1L/R2	0010	dsp:8[SB/FB]	dsp:8[SB]	1010
	R1H/R3	0011		dsp:8[FB]	1011
An	A0	0100	dsp:16[An]	dsp:16[A0]	1100
	A1	0101		dsp:16[A1]	1101
[An]	[A0]	0110	dsp:16[SB]	dsp:16[SB]	1110
	[A1]	0111	abs16	abs16	1111

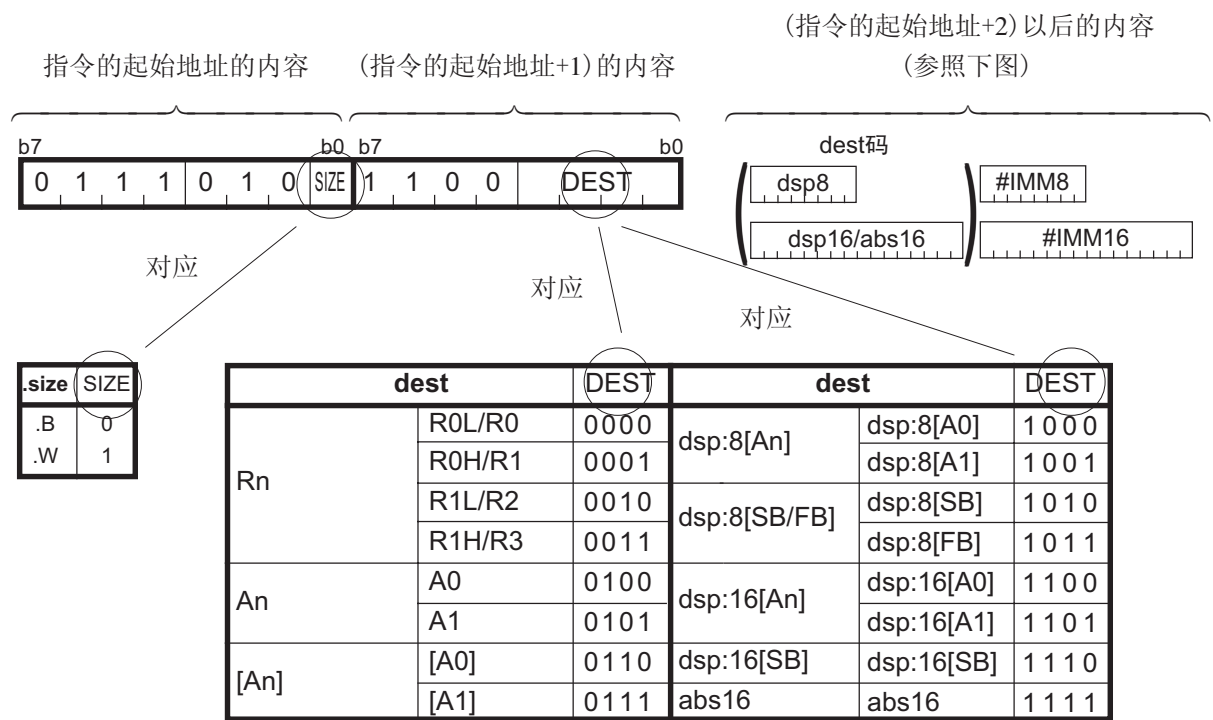
④

【字节数/周期数】

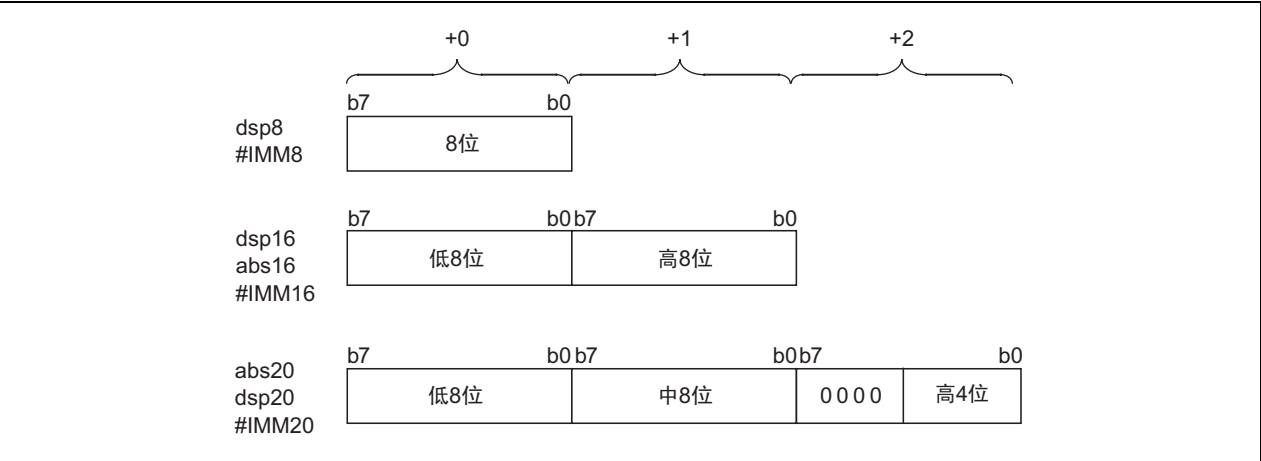
dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/3	4/3	4/3	5/3	5/3	5/3

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

- ① 助记符
表示在本页中说明的助记符。
- ② 语法
用符号表示指令的语法。
- ③ 指令码
表示指令码。() 中的内容根据选择的src/dest被省略。



(指令的起始地址+2) 以后的内容配置如下:

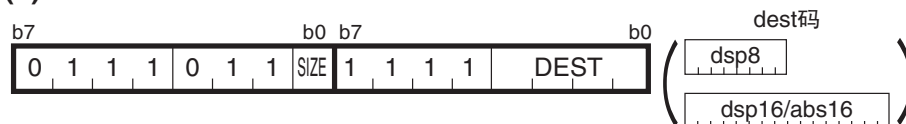


- ④ 字节数/周期数表
表示在执行此指令时所需的周期数和指令的字节数。
但是，周期数根据软件等待等的影响可能会增加。
斜线的左侧为字节数，斜线的右侧为周期数。

4.2 指令码/周期数

ABS

(1) ABS.size dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/3	2/3	2/5	3/5	3/5	4/5	4/5	4/5

ADC

(1) ADC.size #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

ADC

(2) ADC.size src, dest



.size	SIZE
.B	0
.W	1

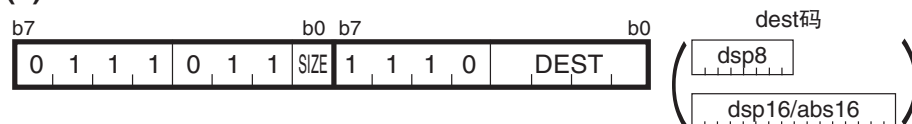
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

ADCF

(1) ADCF.size dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

ADD

(1) ADD.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

ADD

(2) ADD.size:Q #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	-8	1 0 0 0
+1	0 0 0 1	-7	1 0 0 1
+2	0 0 1 0	-6	1 0 1 0
+3	0 0 1 1	-5	1 0 1 1
+4	0 1 0 0	-4	1 1 0 0
+5	0 1 0 1	-3	1 1 0 1
+6	0 1 1 0	-2	1 1 1 0
+7	0 1 1 1	-1	1 1 1 1

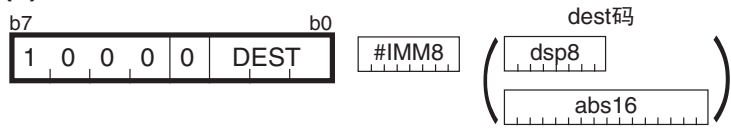
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

ADD

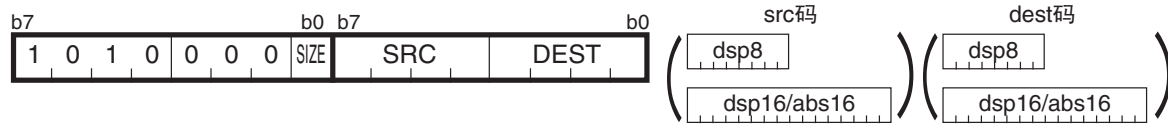
(3) ADD.B:S #IMM8, dest



dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/3	4/3

ADD**(4) ADD.size:G src, dest**

.size	SIZE
.B	0
.W	1

src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

ADD

(5) ADD.B:S src, R0L/R0H

b7

00100

b0

DEST

SRC

src码

dsp8

abs16

src		SRC
Rn	R0L/R0H	0 0
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

dest	DEST
R0L	0
R0H	1

【字节数/周期数】

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

ADD

(6) ADD.size:G #IMM, SP

b7

01111110

b0

SIZE

b7

11101011

b0

#IMM8

#IMM16

.size	SIZE
.B	0
.W	1

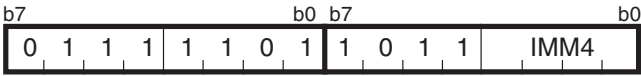
【字节数/周期数】

字节数/周期数	3/2
---------	-----

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

ADD

(7) ADD.size:Q #IMM, SP



*1 无论对长度说明符(.size)选择(.B)还是选择(.W)，指令代码都相同。

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	-8	1 0 0 0
+1	0 0 0 1	-7	1 0 0 1
+2	0 0 1 0	-6	1 0 1 0
+3	0 0 1 1	-5	1 0 1 1
+4	0 1 0 0	-4	1 1 0 0
+5	0 1 0 1	-3	1 1 0 1
+6	0 1 1 0	-2	1 1 1 0
+7	0 1 1 1	-1	1 1 1 1

【字节数/周期数】

字节数/周期数	2/1
---------	-----

ADJNZ

(1) ADJNZ.size #IMM, dest, label



dsp8(label码)=label表示的地址-(指令的起始地址+2)

.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	-8	1 0 0 0
+1	0 0 0 1	-7	1 0 0 1
+2	0 0 1 0	-6	1 0 1 0
+3	0 0 1 1	-5	1 0 1 1
+4	0 1 0 0	-4	1 1 0 0
+5	0 1 0 1	-3	1 1 0 1
+6	0 1 1 0	-2	1 1 1 0
+7	0 1 1 1	-1	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/3	3/3	3/5	4/5	4/5	5/5	5/5	5/5

*1 在转移到label时，表中的周期数增加4个周期。

AND

(1) AND.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

	dest	DEST		dest	DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

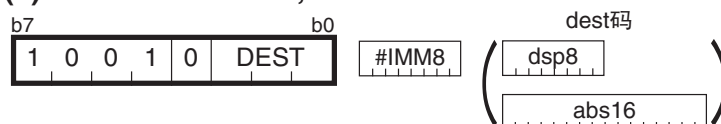
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时, 表中的字节数增加1字节。

AND

(2) AND.B:S #IMM8, dest



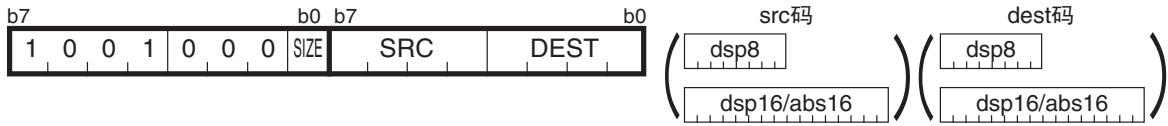
	dest	DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/3	4/3

AND

(3) AND.size:G src, dest



.size	SIZE
.B	0
.W	1

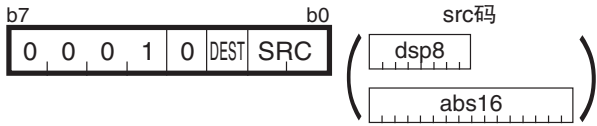
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

AND

(4) AND.B:S src, R0L/R0H



src		SRC
Rn	R0L/R0H	0 0
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

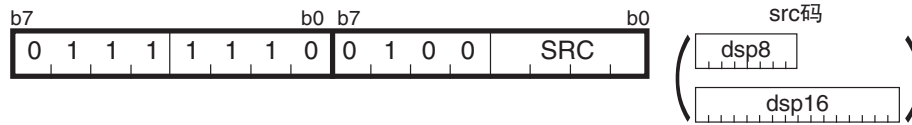
dest	DEST
R0L	0
R0H	1

【字节数/周期数】

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

BAND

(1) BAND src



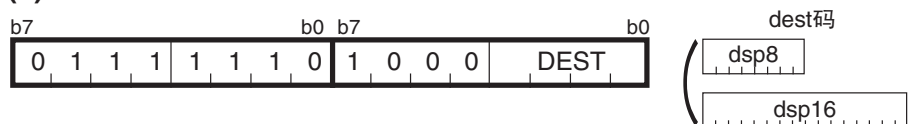
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BCLR

(1) BCLR:G dest



dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/2	3/2	2/6	3/6	3/3	4/6	4/3	4/3

BCLR

(2) BCLR:S bit, base:11[SB]



【字节数/周期数】

字节数/周期数	2/3
---------	-----

BM*Cnd*

(1) BM*Cnd* dest



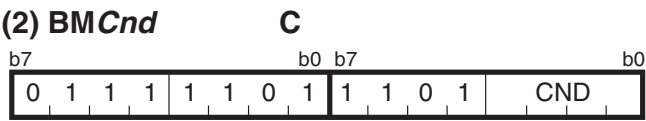
dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

<i>Cnd</i>	CND	<i>Cnd</i>	CND
GEU/C	0 0 0 0 0 0 0 0	LTU/NC	1 1 1 1 1 0 0 0
GTU	0 0 0 0 0 0 0 1	LEU	1 1 1 1 1 0 0 1
EQ/Z	0 0 0 0 0 0 1 0	NE/NZ	1 1 1 1 1 0 1 0
N	0 0 0 0 0 0 1 1	PZ	1 1 1 1 1 0 1 1
LE	0 0 0 0 0 1 0 0	GT	1 1 1 1 1 1 0 0
O	0 0 0 0 0 1 0 1	NO	1 1 1 1 1 1 0 1
GE	0 0 0 0 0 1 1 0	LT	1 1 1 1 1 1 1 0

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	4/6	4/6	3/10	4/10	4/7	5/10	5/7	5/7

BM*Cnd*



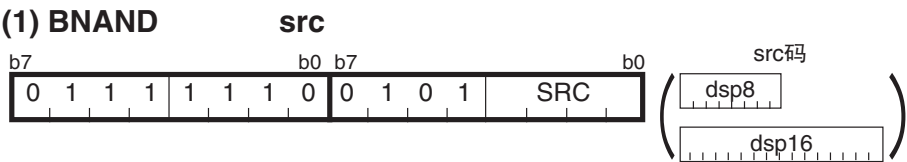
<i>Cnd</i>	CND	<i>Cnd</i>	CND
GEU/C	0000	PZ	0111
GTU	0001	LE	1000
EQ/Z	0010	O	1001
N	0011	GE	1010
LTU/NC	0100	GT	1100
LEU	0101	NO	1101
NE/NZ	0110	LT	1110

【字节数/周期数】

字节数/周期数	2/1
---------	-----

*1 在条件为真时，表中的周期数增加1个周期。

BNAND



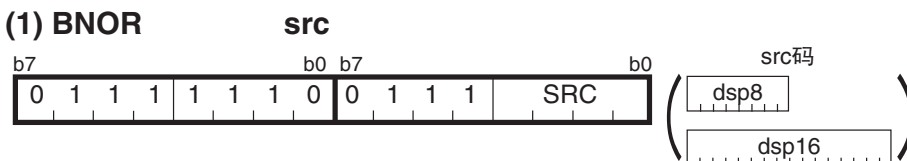
src		SRC	src		SRC
bit,Rn	bit,R0	0000	base:8[An]	base:8[A0]	1000
	bit,R1	0001		base:8[A1]	1001
	bit,R2	0010	bit,base:8 [SB/FB]	bit,base:8[SB]	1010
	bit,R3	0011		bit,base:8[FB]	1011
bit,An	bit,A0	0100	base:16[An]	base:16[A0]	1100
	bit,A1	0101		base:16[A1]	1101
[An]	[A0]	0110	bit,base:16[SB]	bit,base:16[SB]	1110
	[A1]	0111	bit,base:16	bit,base:16	1111

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BNOR

(1) BNOR



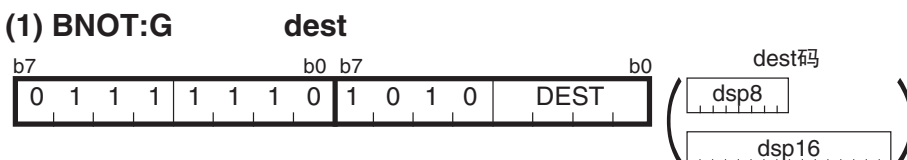
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BNOT

(1) BNOT:G



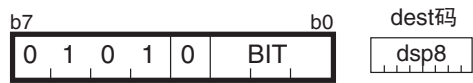
dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/2	3/2	2/6	3/6	3/3	4/6	4/3	4/3

BNOT

(2) BNOT:S bit, base:11[SB]



【字节数/周期数】

字节数/周期数	2/3
---------	-----

BNTST

(1) BNTST src



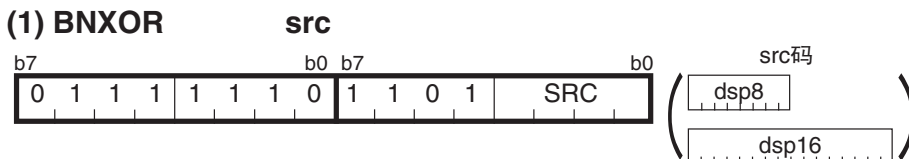
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8[SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8[An]	bit,base:8[SB/FB]	base:16[An]	bit,base:16[SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BNXOR

(1) BNXOR



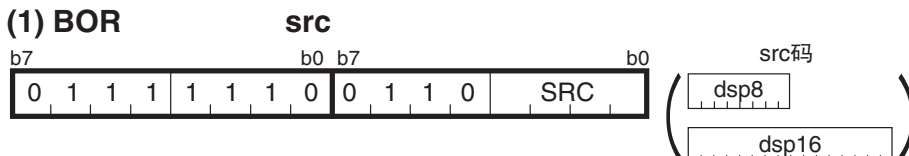
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BOR

(1) BOR



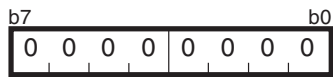
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BRK

(1) BRK



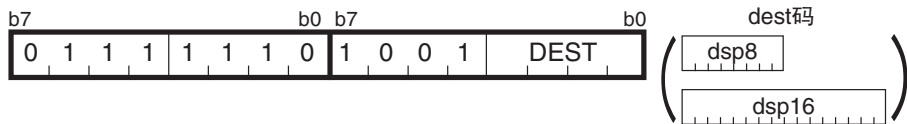
【字节数/周期数】

字节数/周期数	1/27
---------	------

*1 在通过中断表寄存器(INTB)指定BRK中断的转移目标地址时，表中的周期数增加2个周期。
此时，必须在FFFE4₁₆地址～FFFE7₁₆地址中设定FF₁₆。

BSET

(1) BSET:G dest



dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/2	3/2	2/6	3/6	3/3	4/6	4/3	4/3

BSET

(2) BSET:S bit, base:11[SB]

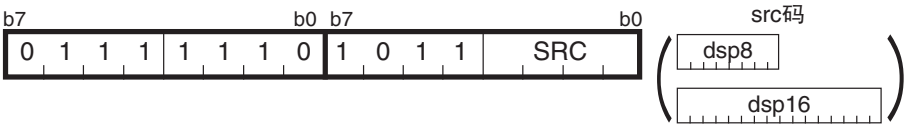


【字节数/周期数】

字节数/周期数	2/3
---------	-----

BTST

(1) BTST:G src



src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8[SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8[An]	bit,base:8[SB/FB]	base:16[An]	bit,base:16[SB]	bit,base:16
字节数/周期数	3/2	3/2	2/6	3/6	3/3	4/6	4/3	4/3

BTST

(2) BTST:S bit, base:11[SB]

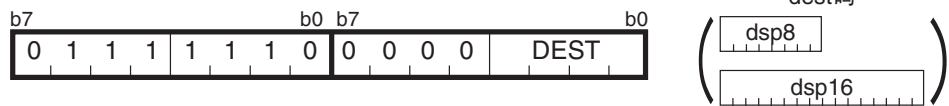


【字节数/周期数】

字节数/周期数	2/3
---------	-----

BTSTC

(1) BTSTC dest



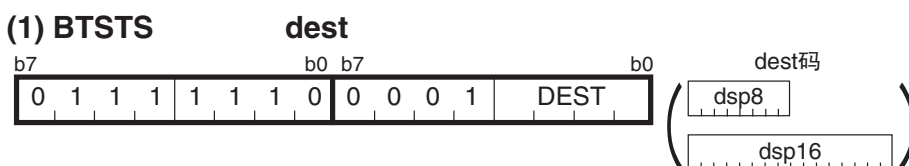
dest			DEST	dest			DEST
bit,Rn	bit,R0		0 0 0 0	base:8[An]	base:8[A0]		1 0 0 0
	bit,R1		0 0 0 1		base:8[A1]		1 0 0 1
	bit,R2		0 0 1 0	bit,base:8[SB/FB]	bit,base:8[SB]		1 0 1 0
	bit,R3		0 0 1 1		bit,base:8[FB]		1 0 1 1
bit,An	bit,A0		0 1 0 0	base:16[An]	base:16[A0]		1 1 0 0
	bit,A1		0 1 0 1		base:16[A1]		1 1 0 1
[An]	[A0]		0 1 1 0	bit,base:16[SB]	bit,base:16[SB]		1 1 1 0
	[A1]		0 1 1 1	bit,base:16	bit,base:16		1 1 1 1

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8[An]	bit,base:8[SB/FB]	base:16[An]	bit,base:16[SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BTSTS

(1) BTSTS



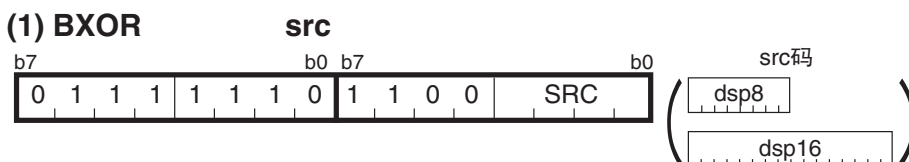
dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

dest	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

BXOR

(1) BXOR



src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	bit,base:16[SB]	bit,base:16[SB]	1 1 1 0
	[A1]	0 1 1 1	bit,base:16	bit,base:16	1 1 1 1

【字节数/周期数】

src	bit,Rn	bit,An	[An]	base:8 [An]	bit,base:8 [SB/FB]	base:16 [An]	bit,base:16 [SB]	bit,base:16
字节数/周期数	3/3	3/3	2/7	3/7	3/4	4/7	4/4	4/4

CMP

(1) CMP.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

CMP

(2) CMP.size:Q #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	-8	1 0 0 0
+1	0 0 0 1	-7	1 0 0 1
+2	0 0 1 0	-6	1 0 1 0
+3	0 0 1 1	-5	1 0 1 1
+4	0 1 0 0	-4	1 1 0 0
+5	0 1 0 1	-3	1 1 0 1
+6	0 1 1 0	-2	1 1 1 0
+7	0 1 1 1	-1	1 1 1 1

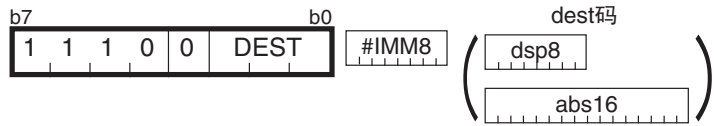
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

CMP

(3) CMP.B:S #IMM8, dest



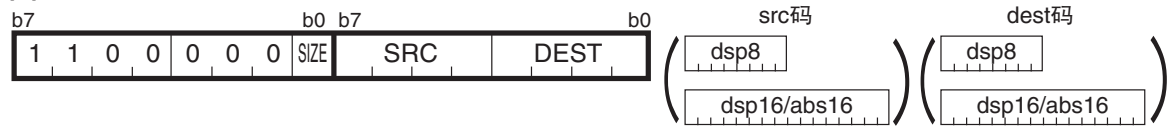
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/3	4/3

CMP

(4) CMP.size:G src, dest



.size	SIZE
.B	0
.W	1

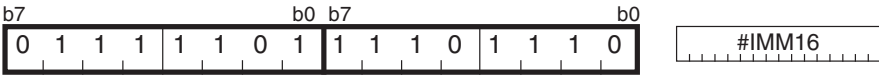
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

DADC

(2) DADC.W #IMM16, R0

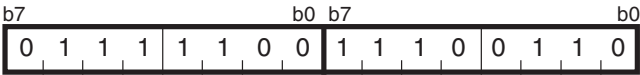


【字节数/周期数】

字节数/周期数	4/5
---------	-----

DADC

(3) DADC.B R0H, R0L

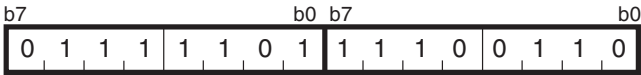


【字节数/周期数】

字节数/周期数	2/5
---------	-----

DADC

(4) DADC.W R1, R0

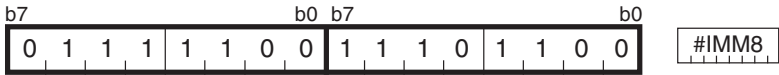


【字节数/周期数】

字节数/周期数	2/5
---------	-----

DADD

(1) DADD.B #IMM8, R0L

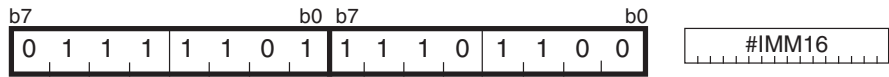


【字节数/周期数】

字节数/周期数	3/5
---------	-----

DADD

(2) **DADD.W** **#IMM16, R0**

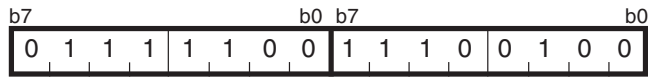


【字节数/周期数】

字节数/周期数	4/5
---------	-----

DADD

(3) **DADD.B** **R0H, R0L**

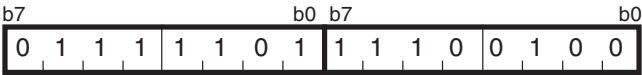


【字节数/周期数】

字节数/周期数	2/5
---------	-----

DADD

(4) DADD.W R1, R0

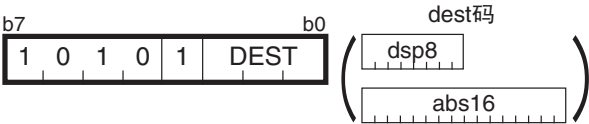


【字节数/周期数】

字节数/周期数	2/5
---------	-----

DEC

(1) DEC.B dest



dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/1	2/3	3/3

DEC

(2) DEC.W dest



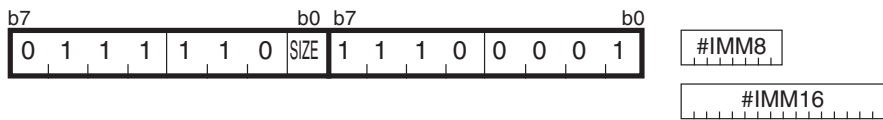
dest	DEST
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/1
---------	-----

DIV

(1) DIV.size #IMM



.size	SIZE
.B	0
.W	1

【字节数/周期数】

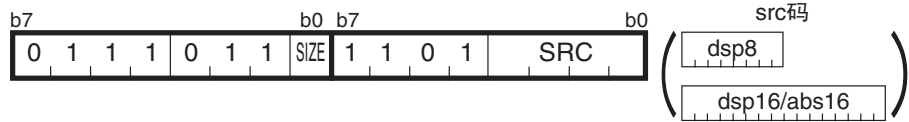
字节数/周期数	3/22
---------	------

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节，周期数增加6个周期。
*2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。

DIV

(2) DIV.size

src



.size	SIZE	src		SRC	src		SRC
.B	0	Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
.W	1		R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
			R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
			R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
		An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
			A1	0 1 0 1		dsp:16[A1]	1 1 0 1
		[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
			[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

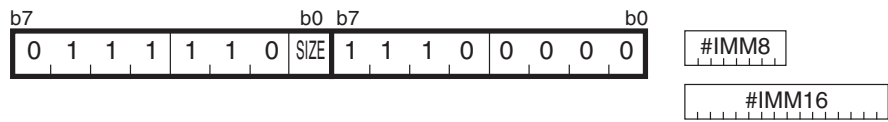
src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/22	2/22	2/24	3/24	3/24	4/24	4/24	4/24

- *1 在长度说明符(.size)为(.W)时，表中的周期数增加6个周期。
*2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。

DIVU

(1) DIVU.size

#IMM



.size	SIZE
.B	0
.W	1

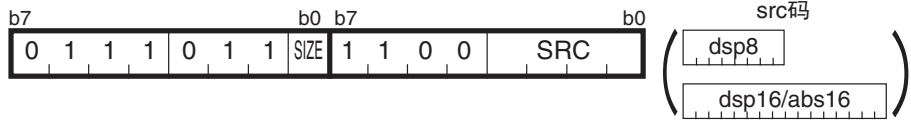
【字节数/周期数】

字节数/周期数	3/18
---------	------

- *2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。
*3 在长度说明符(.size)为(.W)时，表中的字节数增加1字节，周期数增加7个周期。

DIVU

(2) DIVU.size src



.size	SIZE	src		SRC	src		SRC
.B	0	Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
.W	1		R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
			R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
			R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
		An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
			A1	0 1 0 1		dsp:16[A1]	1 1 0 1
		[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
			[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

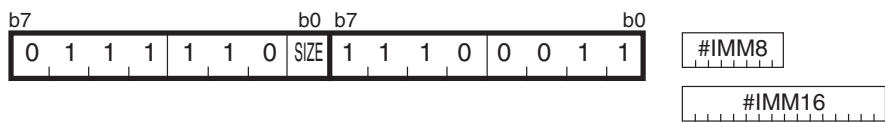
src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/18	2/18	2/20	3/20	3/20	4/20	4/20	4/20

*1 在长度说明符(.size)为(.W)时，表中的周期数增加7个周期。

*2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。

DIVX

(1) DIVX.size #IMM



.size	SIZE
.B	0
.W	1

【字节数/周期数】

字节数/周期数	3/22
---------	------

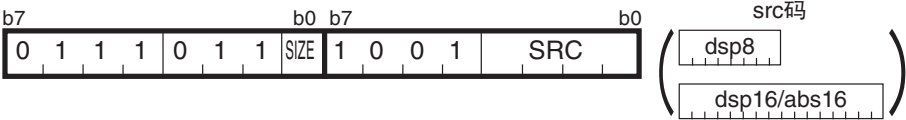
*2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。

*3 在长度说明符(.size)为(.W)时，表中的字节数增加1字节，周期数增加6个周期。

DIVX

(2) DIVX.size

src



.size	SIZE
.B	0
.W	1

src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1	dsp:8[An]	dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1	dsp:8[SB/FB]	dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

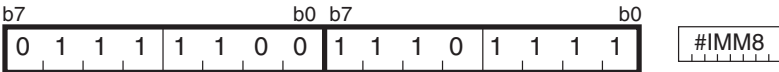
src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/22	2/22	2/24	3/24	3/24	4/24	4/24	4/24

- *1 在长度说明符(.size)为(.W)时，表中的周期数增加6个周期。
*2 在发生溢出时或者根据除数和被除数的值，周期数可能会减少。

DSBB

(1) DSBB.B

#IMM8, R0L

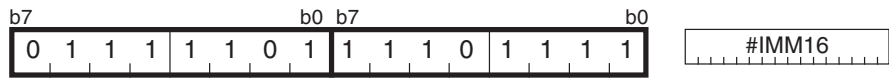


【字节数/周期数】

字节数/周期数	3/4
---------	-----

DSBB

(2) DSBB.W #IMM16, R0

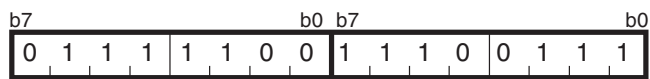


【字节数/周期数】

字节数/周期数	4/4
---------	-----

DSBB

(3) DSBB.B R0H, R0L

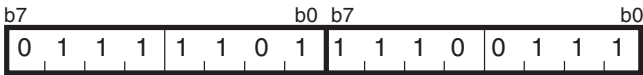


【字节数/周期数】

字节数/周期数	2/4
---------	-----

DSBB

(4) DSBB.W R1, R0

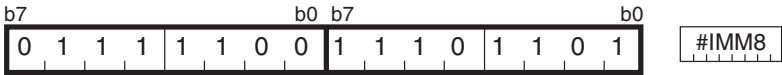


【字节数/周期数】

字节数/周期数	2/4
---------	-----

DSUB

(1) DSUB.B #IMM8, R0L

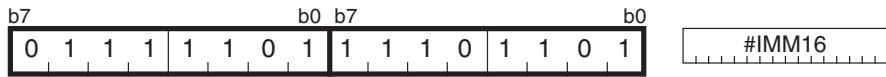


【字节数/周期数】

字节数/周期数	3/4
---------	-----

DSUB

(2) DSUB.W #IMM16, R0

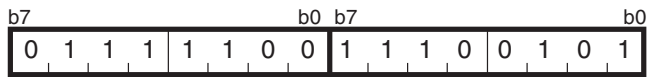


【字节数/周期数】

字节数/周期数	4/4
---------	-----

DSUB

(3) DSUB.B R0H, R0L

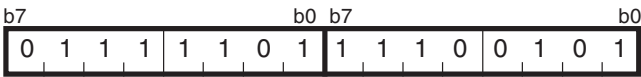


【字节数/周期数】

字节数/周期数	2/4
---------	-----

DSUB

(4) DSUB.W R1, R0



【字节数/周期数】

字节数/周期数	2/4
---------	-----

ENTER

(1) ENTER #IMM8

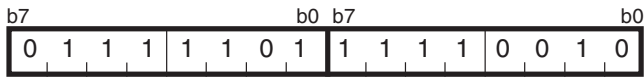


【字节数/周期数】

字节数/周期数	3/4
---------	-----

EXITD

(1) EXITD

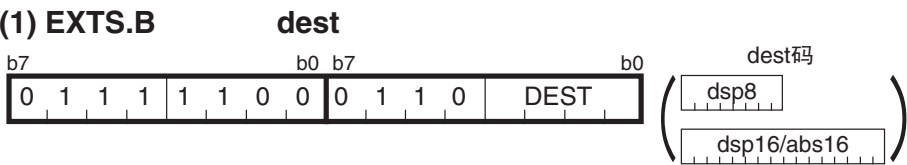


【字节数/周期数】

字节数/周期数	2/9
---------	-----

EXTS

(1) EXTS.B



dest		DEST	dest		DEST
Rn	R0L	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	---	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
---	---	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

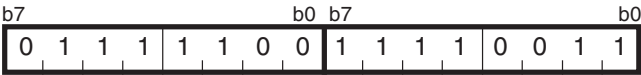
*1 记号“---”为不能选择。

【字节数/周期数】

dest	Rn	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/3	2/5	3/5	3/5	4/5	4/5	4/5

EXTS

(2) EXTS.W R0

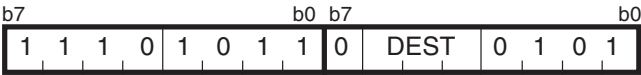


【字节数/周期数】

字节数/周期数	2/3
---------	-----

FCLR

(1) FCLR dest

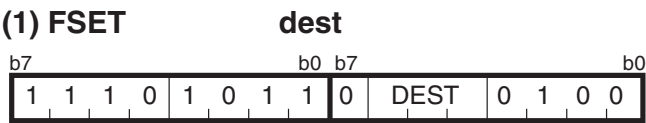


dest	DEST
C	0 0 0
D	0 0 1
Z	0 1 0
S	0 1 1
B	1 0 0
O	1 0 1
I	1 1 0
U	1 1 1

【字节数/周期数】

字节数/周期数	2/2
---------	-----

FSET

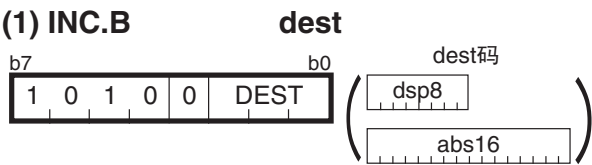


dest	DEST
C	0 0 0
D	0 0 1
Z	0 1 0
S	0 1 1
B	1 0 0
O	1 0 1
I	1 1 0
U	1 1 1

【字节数/周期数】

字节数/周期数	2/2
---------	-----

INC



dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/1	2/3	3/3

INC

(2) INC.W dest



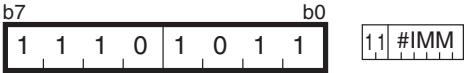
dest	DEST
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/1
---------	-----

INT

(1) INT #IMM

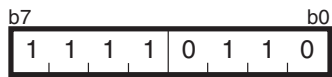


【字节数/周期数】

字节数/周期数	2/19
---------	------

INTO

(1) INTO



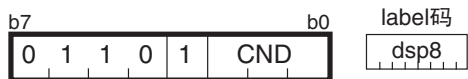
【字节数/周期数】

字节数/周期数	1/1
---------	-----

*1 在O标志为1时，表中的周期数增加19个周期。

JCnd

(1) JCnd label



dsp8=label表示的地址－(指令的起始地址+1)

Cnd	CND	Cnd	CND
GEU/C	0 0 0	LTU/NC	1 0 0
GTU	0 0 1	LEU	1 0 1
EQ/Z	0 1 0	NE/NZ	1 1 0
N	0 1 1	PZ	1 1 1

【字节数/周期数】

字节数/周期数	2/2
---------	-----

*2 在转移到label时，表中的周期数增加2个周期。

J*Cnd*

(2) J*Cnd*

label

b7

0

1

1

1

1

1

0

1

1

1

0

0

CND

b0

label码

dsp8

dsp8=label表示的地址－(指令的起始地址+2)

<i>Cnd</i>	CND	<i>Cnd</i>	CND
LE	1 0 0 0	GT	1 1 0 0
O	1 0 0 1	NO	1 1 0 1
GE	1 0 1 0	LT	1 1 1 0

【字节数/周期数】

字节数/周期数	3/2
---------	-----

*1 在转移到label时，表中的周期数增加2个周期。

JMP

(1) JMP.S label

b7

0

1

1

0

0

dsp

b0

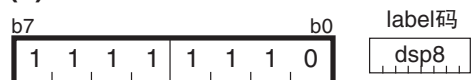
dsp=label表示的地址－(指令的起始地址+2)

【字节数/周期数】

字节数/周期数	1/5
---------	-----

JMP

(2) JMP.B label



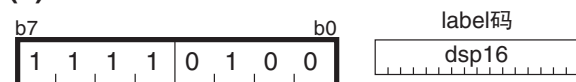
dsp8=label表示的地址－(指令的起始地址+1)

【字节数/周期数】

字节数/周期数	2/4
---------	-----

JMP

(3) JMP.W label



dsp16=label表示的地址－(指令的起始地址+1)

【字节数/周期数】

字节数/周期数	3/4
---------	-----

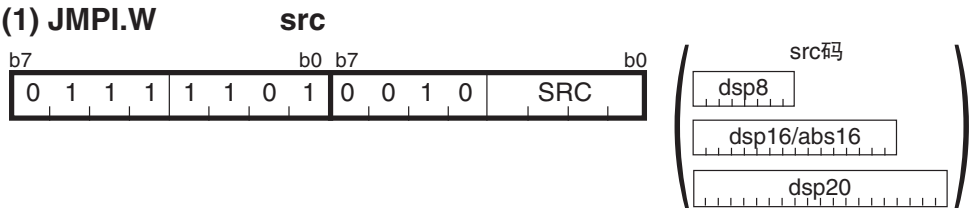
JMP



【字节数/周期数】

字节数/周期数	4/4
---------	-----

JMPI



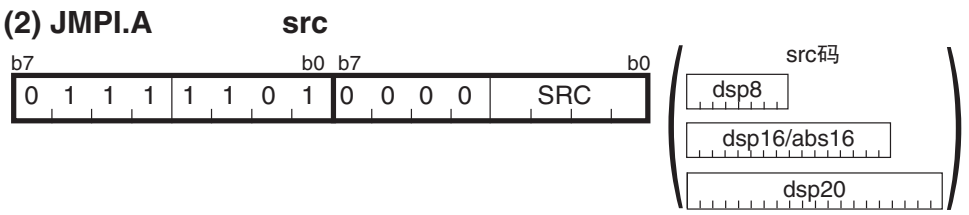
src		SRC	src		SRC
Rn	R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	A1	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:20[An]	dsp:16[SB]	abs16
字节数/周期数	2/7	2/7	2/11	3/11	3/11	5/11	4/11	4/11

JMPI

(2) JMP1.A



src		SRC	src		SRC
Rn	R2R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R3R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A1A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	---	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号“---”为不能选择。

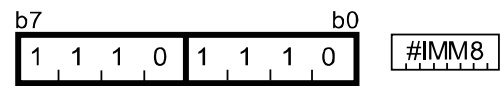
【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:20[An]	dsp:16[SB]	abs16
字节数/周期数	2/6	2/6	2/10	3/10	3/10	5/10	4/10	4/10

JMPS

(1) JMPS

#IMM8

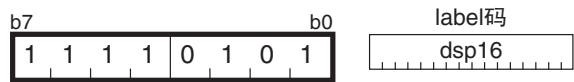


【字节数/周期数】

字节数/周期数	2/9
---------	-----

JSR

(1) JSR.W label



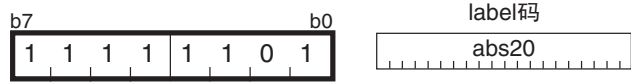
dsp16=label表示的地址－(指令的起始地址+1)

【字节数/周期数】

字节数/周期数	3/8
---------	-----

JSR

(2) JSR.A label

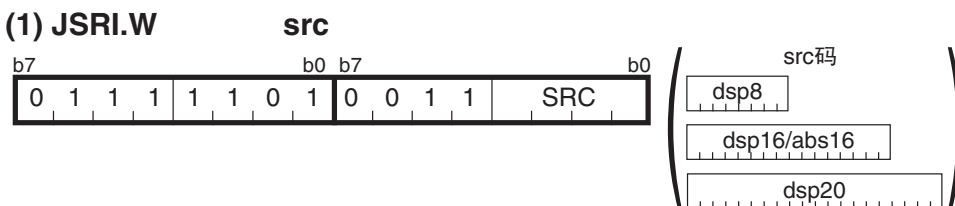


【字节数/周期数】

字节数/周期数	4/9
---------	-----

JSRI

(1) JSRI.W



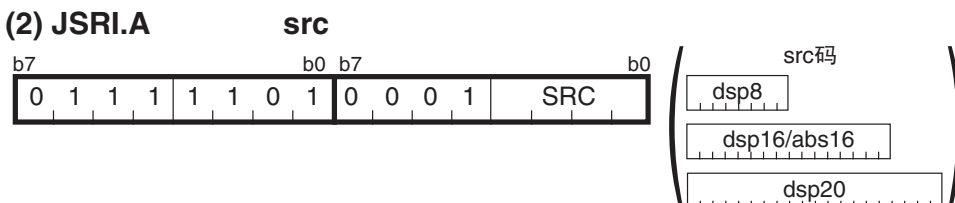
src		SRC	src		SRC
Rn	R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	A1	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:20[An]	dsp:16[SB]	abs16
字节数/周期数	2/11	2/11	2/15	3/15	3/15	5/15	4/15	4/15

JSRI

(2) JSRI.A



src		SRC	src		SRC
Rn	R2R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R3R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A1A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	---	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

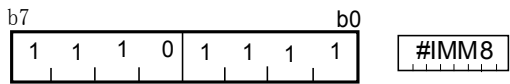
*1 记号“---”为不能选择。

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:20[An]	dsp:16[SB]	abs16
字节数/周期数	2/11	2/11	2/15	3/15	3/15	5/15	4/15	4/15

JSRS

(1)JSRS #IMM8



【字节数/周期数】

字节数/周期数	2/13
---------	------

LDC

(1) LDC #IMM16, dest



dest	DEST
---	0 0 0
INTBL	0 0 1
INTBH	0 1 0
FLG	0 1 1
ISP	1 0 0
SP	1 0 1
SB	1 1 0
FB	1 1 1

*1 记号“---”为不能选择。

【字节数/周期数】

字节数/周期数	4/2
---------	-----

LDC

(2) LDC src, dest



src			SRC	src			SRC	dest	DEST
Rn	R0		0 0 0 0	dsp:8[An]	dsp:8[A0]		1 0 0 0	---	0 0 0
	R1		0 0 0 1		dsp:8[A1]		1 0 0 1	INTBL	0 0 1
	R2		0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]		1 0 1 0	INTBH	0 1 0
	R3		0 0 1 1		dsp:8[FB]		1 0 1 1	FLG	0 1 1
An	A0		0 1 0 0	dsp:16[An]	dsp:16[A0]		1 1 0 0	ISP	1 0 0
	A1		0 1 0 1		dsp:16[A1]		1 1 0 1	SP	1 0 1
[An]	[A0]		0 1 1 0	dsp:16[SB]	dsp:16[SB]		1 1 1 0	SB	1 1 0
	[A1]		0 1 1 1	abs16	abs16		1 1 1 1	FB	1 1 1

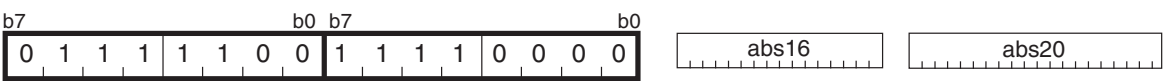
*1 记号 “---” 为不能选择。

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

LDCTX

(1) LDCTX abs16, abs20



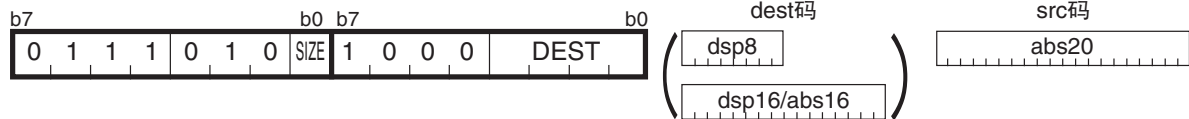
【字节数/周期数】

字节数/周期数	7/11+2×m
---------	----------

*2 m为传送次数。

LDE

(1) LDE.size abs20, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	5/4	5/4	5/5	6/5	6/5	7/5	7/5	7/5

LDE

(2) LDE.size dsp:20[A0], dest



.size	SIZE
.B	0
.W	1

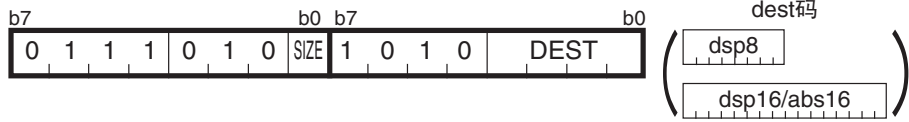
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	5/4	5/4	5/5	6/5	6/5	7/5	7/5	7/5

LDE

(3) LDE.size [A1A0], dest



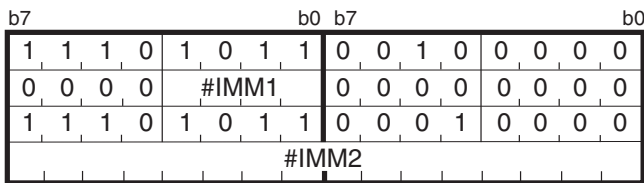
.size	SIZE	dest		DEST	dest		DEST
.B	0	Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
.W	1		R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
			R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
			R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
		An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
			A1	0 1 0 1		dsp:16[A1]	1 1 0 1
		[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
			[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/4	2/4	2/5	3/5	3/5	4/5	4/5	4/5

LDINTB

(1) LDINTB #IMM



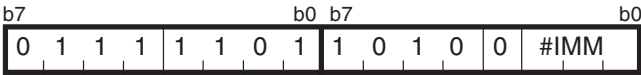
*1 #IMM1为#IMM的高4位
#IMM2为#IMM的低16位。

【字节数/周期数】

字节数/周期数	8/4
---------	-----

LDIPL

(1) LDIPL #IMM



【字节数/周期数】

字节数/周期数	2/2
---------	-----

MOV

(1) MOV.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

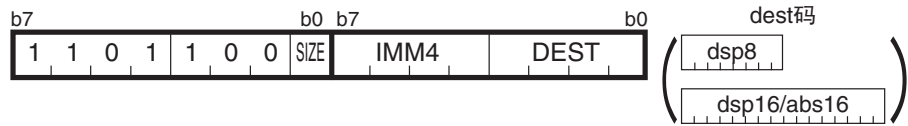
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/3	4/3	4/3	5/3	5/3	5/3

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

MOV

(2) MOV.size:Q #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	-8	1 0 0 0
+1	0 0 0 1	-7	1 0 0 1
+2	0 0 1 0	-6	1 0 1 0
+3	0 0 1 1	-5	1 0 1 1
+4	0 1 0 0	-4	1 1 0 0
+5	0 1 0 1	-3	1 1 0 1
+6	0 1 1 0	-2	1 1 1 0
+7	0 1 1 1	-1	1 1 1 1

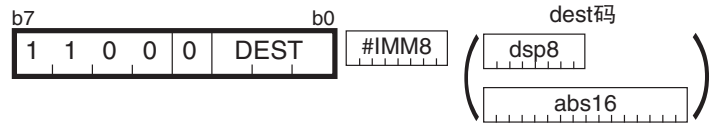
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/2	3/2	3/2	4/2	4/2	4/2

MOV

(3) MOV.B:S #IMM8, dest



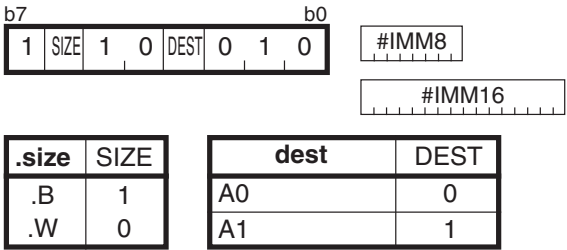
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/2	4/2

MOV

(4) MOV.size:S #IMM, dest



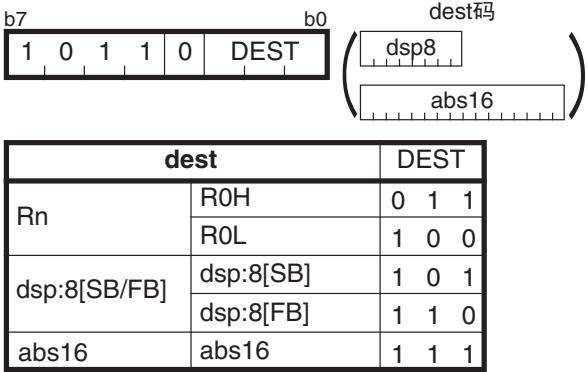
【字节数/周期数】

字节数/周期数	2/1
---------	-----

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节，周期数增加1个周期。

MOV

(5) MOV.B:Z #0, dest

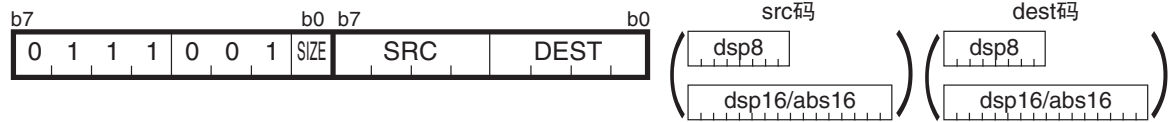


【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/1	2/2	3/2

MOV

(6) MOV.size:G src, dest



.size	SIZE
.B	0
.W	1

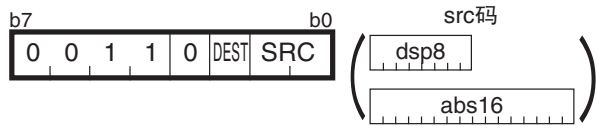
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/2	3/2	3/2	4/2	4/2	4/2
An	2/2	2/2	2/2	3/2	3/2	4/2	4/2	4/2
[An]	2/3	2/3	2/3	3/3	3/3	4/3	4/3	4/3
dsp:8[An]	3/3	3/3	3/3	4/3	4/3	5/3	5/3	5/3
dsp:8[SB/FB]	3/3	3/3	3/3	4/3	4/3	5/3	5/3	5/3
dsp:16[An]	4/3	4/3	4/3	5/3	5/3	6/3	6/3	6/3
dsp:16[SB]	4/3	4/3	4/3	5/3	5/3	6/3	6/3	6/3
abs16	4/3	4/3	4/3	5/3	5/3	6/3	6/3	6/3

MOV

(7) MOV.B:S src, dest



src		SRC
Rn	R0L/R0H	0 0
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

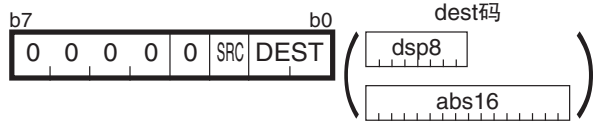
dest	DEST
A0	0
A1	1

【字节数/周期数】

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

MOV

(8) MOV.B:S R0L/R0H, dest



src	SRC
R0L	0
R0H	1

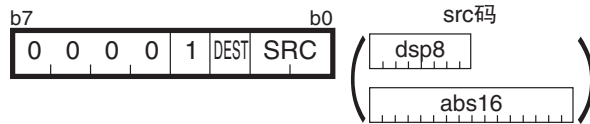
dest		DEST
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

【字节数/周期数】

dest	dsp:8[SB/FB]	abs16
字节数/周期数	2/2	3/2

MOV

(9) MOV.B:S src, R0L/R0H



src		SRC
Rn	R0L/R0H	0 0
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

dest	DEST
R0L	0
R0H	1

【字节数/周期数】

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

MOV

(10) MOV.size:G dsp:8[SP], dest



.size	SIZE
.B	0
.W	1

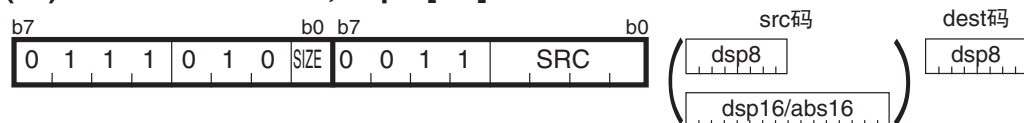
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/3	4/3	4/3	5/3	5/3	5/3

MOV

(11) MOV.size:G src, dsp:8[SP]



.size	SIZE
.B	0
.W	1

src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4

MOVA

(1) MOVA src, dest



src		SRC
dsp:8[An]	dsp:8[A0]	1 0 0 0
	dsp:8[A1]	1 0 0 1
dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	dsp:8[FB]	1 0 1 1
dsp:16[An]	dsp:16[A0]	1 1 0 0
	dsp:16[A1]	1 1 0 1
dsp:16[SB]	dsp:16[SB]	1 1 1 0
abs16	abs16	1 1 1 1

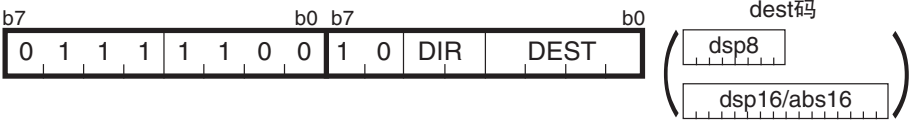
dest	DEST
R0	0 0 0
R1	0 0 1
R2	0 1 0
R3	0 1 1
A0	1 0 0
A1	1 0 1

【字节数/周期数】

src	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	4/2	4/2	4/2

MOV Dir

(1) MOV *Dir* R0L, dest



<i>Dir</i>	DIR
LL	0 0
LH	1 0
HL	0 1
HH	1 1

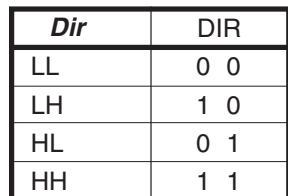
dest		DEST	dest		DEST
Rn	---	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H	0 0 1 1		dsp:8[FB]	1 0 1 1
An	---	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号“---”为不能选择。

【字节数/周期数】

dest	Rn	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
MOVHH, MOVLH	2/4	2/5	3/5	3/5	4/5	4/5	4/5
MOVHL, MOVLH	2/7	2/8	3/8	3/8	4/8	4/8	4/8

(2) MOV *Dir* src, R0L



src		SRC	src		SRC
Rn	R0L	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H	0 0 1 1		dsp:8[FB]	1 0 1 1
An	---	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

src	Rn	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
MOVHH, MOVLL	2/3	2/5	3/5	3/5	4/5	4/5	4/5
MOVHL, MOVLH	2/6	2/8	3/8	3/8	4/8	4/8	4/8

MUL

(1) MUL.size #IMM, dest

b7

0 1 1 1 1 1 0

b0

SIZE

0 1 0 1

b0

DEST

dest码

dsp8

dsp16/abs16

#IMM8

#IMM16

.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST	
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0	
	--- /R1	0 0 0 1		dsp:8[A1]	1 0 0 1	
	R1L/---	0 0 1 0		dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1			dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0	
	---	0 1 0 1		dsp:16[A1]	1 1 0 1	
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0	
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1	

*1 记号 “---” 为不能选择。

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/4	3/4	3/5	4/5	4/5	5/5	5/5	5/5

- *2 在长度说明符(.size)为(.W)时，如果dest为Rn或者An，表中的字节数就增加1字节，周期数增加1个周期。
- *3 在长度说明符(.size)为(.W)时，如果dest不为Rn和An，表中的字节数就增加1个字节，周期数增加2个周期。

MUL

(2) MUL.size src, dest



.size	SIZE
.B	0
.W	1

src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	--- /R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号“---”为不能选择。

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/4	2/4	2/5	3/5	3/5	4/5	4/5	4/5
An	2/4	2/5	2/5	3/5	3/5	4/5	4/5	4/5
[An]	2/6	2/6	2/6	3/6	3/6	4/6	4/6	4/6
dsp:8[An]	3/6	3/6	3/6	4/6	4/6	5/6	5/6	5/6
dsp:8[SB/FB]	3/6	3/6	3/6	4/6	4/6	5/6	5/6	5/6
dsp:16[An]	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6
dsp:16[SB]	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6
abs16	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6

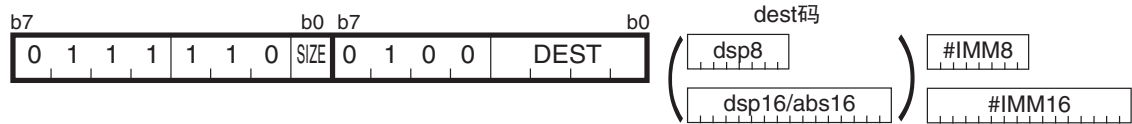
*2 在长度说明符(.size)为(.W)时，如果src为An并且dest为Rn，表中的周期数就增加1个周期。

*3 在长度说明符(.size)为(.W)时，如果src不为An并且dest为Rn或者An，表中的周期数就增加1个周期。

*4 在长度说明符(.size)为(.W)时，如果dest不为Rn和An，表中的周期数就增加2个周期。

MULU

(1) MULU.size #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	--- /R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号 “---” 为不能选择。

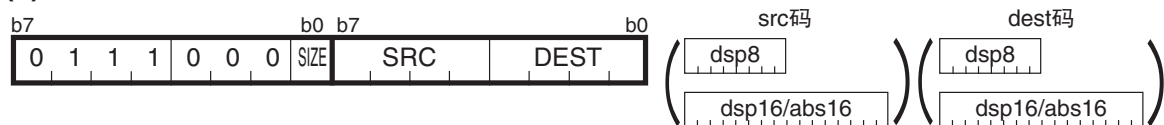
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/4	3/4	3/5	4/5	4/5	5/5	5/5	5/5

- *2 在长度说明符(.size)为(.W)时，如果dest为Rn或者An，表中的字节数就增加1字节，周期数增加1个周期。
- *3 在长度说明符(.size)为(.W)时，如果dest不为Rn和An，表中的字节数就增加1字节，周期数增加2个周期。

MULU

(2) MULU.size src, dest



.size	SIZE
.B	0
.W	1

src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	--- /R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号“---”为不能选择。

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/4	2/4	2/5	3/5	3/5	4/5	4/5	4/5
An	2/4	2/5	2/5	3/5	3/5	4/5	4/5	4/5
[An]	2/6	2/6	2/6	3/6	3/6	4/6	4/6	4/6
dsp:8[An]	3/6	3/6	3/6	4/6	4/6	5/6	5/6	5/6
dsp:8[SB/FB]	3/6	3/6	3/6	4/6	4/6	5/6	5/6	5/6
dsp:16[An]	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6
dsp:16[SB]	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6
abs16	4/6	4/6	4/6	5/6	5/6	6/6	6/6	6/6

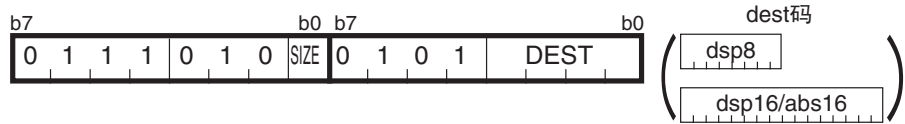
*2 在长度说明符(.size)为(.W)时，如果src为An并且dest为Rn，表中的周期数就增加1个周期。

*3 在长度说明符(.size)为(.W)时，如果src不为An并且dest为Rn或者An，表中的周期数就增加1个周期。

*4 在长度说明符(.size)为(.W)时，如果dest不为Rn和An，表中的周期数就增加2个周期。

NEG

(1) NEG.size dest



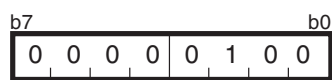
.size	SIZE	dest	DEST	dest	DEST
.B	0	Rn	R0L/R0	dsp:8[An]	dsp:8[A0]
.W	1		R0H/R1		dsp:8[A1]
			R1L/R2	dsp:8[SB/FB]	dsp:8[SB]
			R1H/R3		dsp:8[FB]
		An	A0	dsp:16[An]	dsp:16[A0]
			A1		dsp:16[A1]
		[An]	[A0]	dsp:16[SB]	dsp:16[SB]
			[A1]	abs16	abs16

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

NOP

(1) NOP

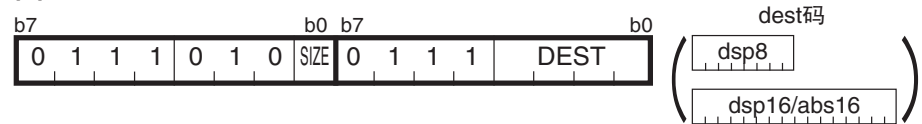


【字节数/周期数】

字节数/周期数	1/1
---------	-----

NOT

(1) NOT.size:G dest



.size	SIZE
.B	0
.W	1

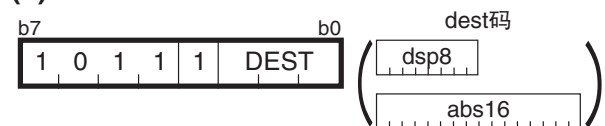
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

NOT

(2) NOT.B:S dest



dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/1	2/3	3/3

OR

(1) OR.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

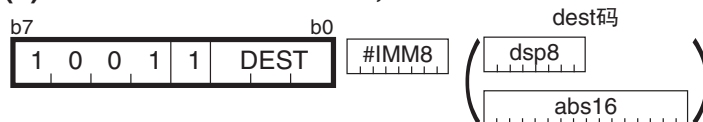
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

OR

(2) OR.B:S #IMM8, dest



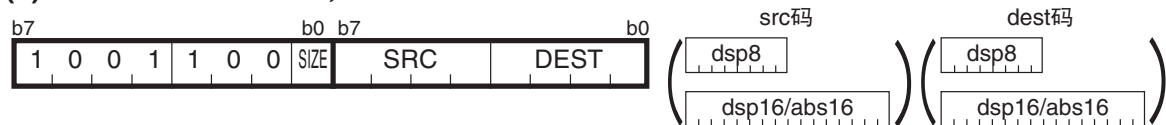
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/3	4/3

OR

(3) OR.size:G src, dest



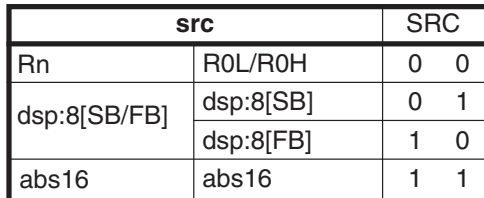
.size	SIZE
.B	0
.W	1

src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

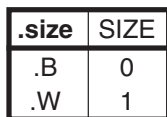
【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

OP

src, R0L/R0H

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

dest

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4

POP

(2) POP.B:S dest



dest	DEST
R0L	0
R0H	1

【字节数/周期数】

字节数/周期数	1/3
---------	-----

POP

(3) POP.W:S dest

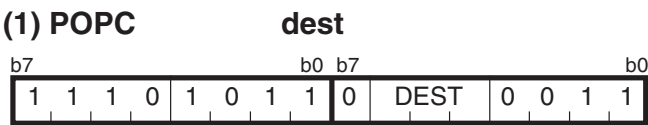


dest	DEST
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/3
---------	-----

POPC



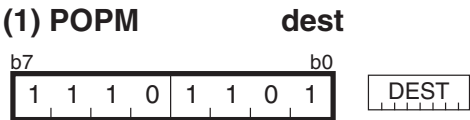
dest	DEST	dest	DEST
---	0 0 0	ISP	1 0 0
INTBL	0 0 1	SP	1 0 1
INTBH	0 1 0	SB	1 1 0
FLG	0 1 1	FB	1 1 1

*1 记号“---”为不能选择。

【字节数/周期数】

字节数/周期数	2/3
---------	-----

POPM



dest							
FB	SB	A1	A0	R3	R2	R1	R0
DEST*2							

*2 对应被选择的寄存器的位为“1”。
对应未被选择的寄存器的位为“0”。

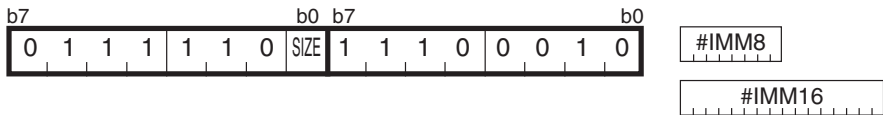
【字节数/周期数】

字节数/周期数	2/3
---------	-----

*3 恢复的寄存器为2个以上时的周期数是2×m(m: 恢复的寄存器数)。

PUSH

(1) PUSH.size:G #IMM



.size	SIZE
.B	0
.W	1

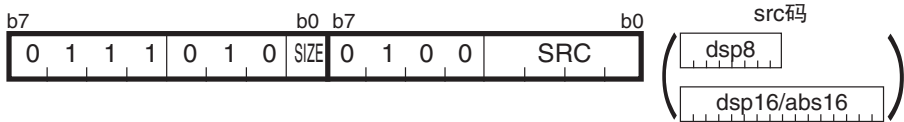
【字节数/周期数】

字节数/周期数	3/2
---------	-----

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

PUSH

(2) PUSH.size:G src



.size	SIZE
.B	0
.W	1

src		SRC	src		SRC
Rn	R0L/R0	0000	dsp:8[An]	dsp:8[A0]	1000
	R0H/R1	0001		dsp:8[A1]	1001
	R1L/R2	0010	dsp:8[SB/FB]	dsp:8[SB]	1010
	R1H/R3	0011		dsp:8[FB]	1011
An	A0	0100	dsp:16[An]	dsp:16[A0]	1100
	A1	0101		dsp:16[A1]	1101
[An]	[A0]	0110	dsp:16[SB]	dsp:16[SB]	1110
	[A1]	0111	abs16	abs16	1111

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/2	2/2	2/4	3/4	3/4	4/4	4/4	4/4

PUSH

(3) PUSH.B:S src



src	SRC
R0L	0
R0H	1

【字节数/周期数】

字节数/周期数	1/2
---------	-----

PUSH

(4) PUSH.W:S src



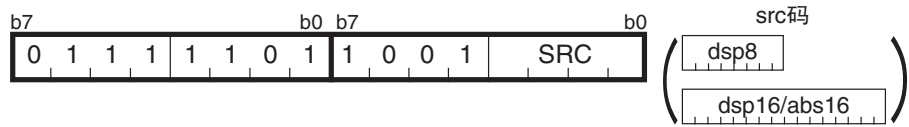
src	SRC
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/2
---------	-----

PUSHA

(1) PUSHA **src**



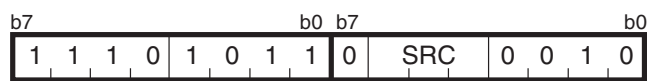
src		SRC
dsp:8[An]	dsp:8[A0]	1 0 0 0
	dsp:8[A1]	1 0 0 1
dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	dsp:8[FB]	1 0 1 1
dsp:16[An]	dsp:16[A0]	1 1 0 0
	dsp:16[A1]	1 1 0 1
dsp:16[SB]	dsp:16[SB]	1 1 1 0
abs16	abs16	1 1 1 1

【字节数/周期数】

src	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs:16
字节数/周期数	3/2	3/2	4/2	4/2	4/2

PUSHC

(1) PUSHC **src**



src	SRC	src	SRC
---	0 0 0	ISP	1 0 0
INTBL	0 0 1	SP	1 0 1
INTBH	0 1 0	SB	1 1 0
FLG	0 1 1	FB	1 1 1

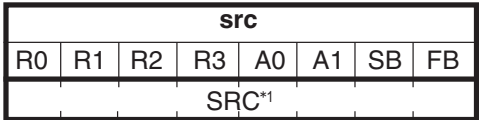
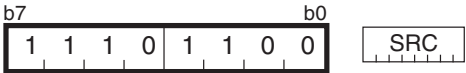
*1 记号“---”为不能选择。

【字节数/周期数】

字节数/周期数	2/2
---------	-----

PUSHM

(1) PUSHM src



*1 对应被选择的寄存器的位为“1”。
对应未被选择的寄存器的位为“0”。

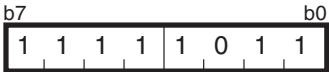
【字节数/周期数】



*2 m为保存的寄存器数。

REIT

(1) REIT

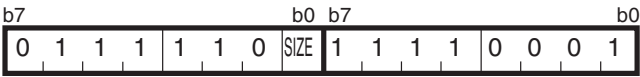


【字节数/周期数】



RMPA

(1) RMPA.size



.size	SIZE
.B	0
.W	1

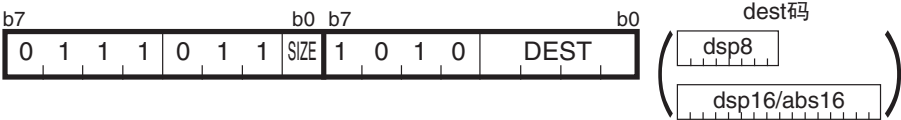
【字节数/周期数】

字节数/周期数	2/4+7×m
---------	---------

- *1 m为运算次数。
- *2 在长度说明符(.size)为(.W)时，表中的周期数为(6+9×m)个周期。

ROLC

(1) ROLC.size dest



.size	SIZE
.B	0
.W	1

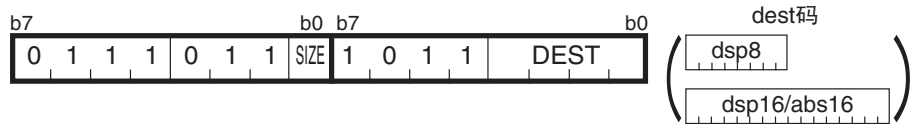
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

RORC

(1) RORC.size dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/3	3/3	3/3	4/3	4/3	4/3

ROT

(1) ROT.size #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
+1	0 0 0 0	-1	1 0 0 0
+2	0 0 0 1	-2	1 0 0 1
+3	0 0 1 0	-3	1 0 1 0
+4	0 0 1 1	-4	1 0 1 1
+5	0 1 0 0	-5	1 1 0 0
+6	0 1 0 1	-6	1 1 0 1
+7	0 1 1 0	-7	1 1 1 0
+8	0 1 1 1	-8	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

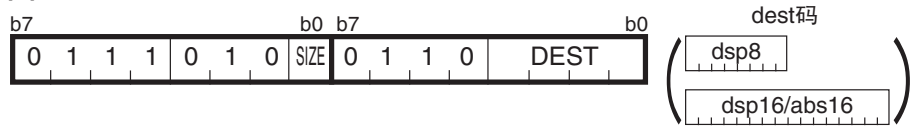
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1+m	2/1+m	2/2+m	3/2+m	3/2+m	4/2+m	4/2+m	4/2+m

*1 m为循环移位的位数。

ROT

(2) ROT.size R1H, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/---	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	--- /R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号 “---” 为不能选择。

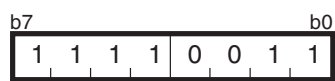
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/2+m	2/2+m	2/3+m	3/3+m	3/3+m	4/3+m	4/3+m	4/3+m

*2 m为循环移位的位数。

RTS

(1) RTS

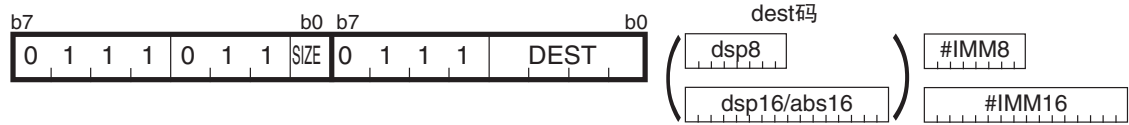


【字节数/周期数】

字节数/周期数	1/6
---------	-----

SBB

(1) SBB.size #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

SBB**(2) SBB.size src, dest**

.size	SIZE
.B	0
.W	1

src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

SBJNZ

(1) SBJNZ.size #IMM, dest, label



dsp8 (label码) = label表示的地址 - (指令的起始地址 + 2)

.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	+8	1 0 0 0
-1	0 0 0 1	+7	1 0 0 1
-2	0 0 1 0	+6	1 0 1 0
-3	0 0 1 1	+5	1 0 1 1
-4	0 1 0 0	+4	1 1 0 0
-5	0 1 0 1	+3	1 1 0 1
-6	0 1 1 0	+2	1 1 1 0
-7	0 1 1 1	+1	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

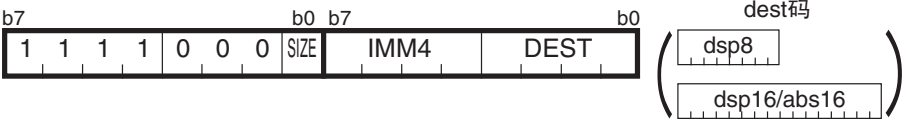
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/3	3/3	3/5	4/5	4/5	5/5	5/5	5/5

*1 在转移到label时，表中的周期数增加4个周期。

SHA

(1) SHA.size #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
+1	0000	-1	1000
+2	0001	-2	1001
+3	0010	-3	1010
+4	0011	-4	1011
+5	0100	-5	1100
+6	0101	-6	1101
+7	0110	-7	1110
+8	0111	-8	1111

dest		DEST	dest		DEST
Rn	R0L/R0	0000	dsp:8[An]	dsp:8[A0]	1000
	R0H/R1	0001		dsp:8[A1]	1001
	R1L/R2	0010	dsp:8[SB/FB]	dsp:8[SB]	1010
	R1H/R3	0011		dsp:8[FB]	1011
An	A0	0100	dsp:16[An]	dsp:16[A0]	1100
	A1	0101		dsp:16[A1]	1101
[An]	[A0]	0110	dsp:16[SB]	dsp:16[SB]	1110
	[A1]	0111	abs16	abs16	1111

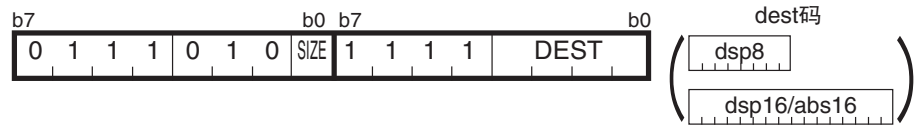
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1+m	2/1+m	2/2+m	3/2+m	3/2+m	4/2+m	4/2+m	4/2+m

*1 m为移位的位数。

SHA

(2) SHA.size R1H, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/---	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	--- /R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号 “---” 为不能选择。

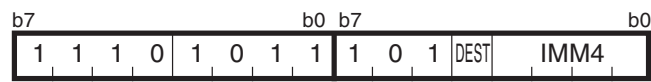
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/2+m	2/2+m	2/3+m	3/3+m	3/3+m	4/3+m	4/3+m	4/3+m

*2 m为移位的位数。

SHA

(3) SHA.L #IMM, dest



#IMM	IMM4	#IMM	IMM4
+1	0 0 0 0	-1	1 0 0 0
+2	0 0 0 1	-2	1 0 0 1
+3	0 0 1 0	-3	1 0 1 0
+4	0 0 1 1	-4	1 0 1 1
+5	0 1 0 0	-5	1 1 0 0
+6	0 1 0 1	-6	1 1 0 1
+7	0 1 1 0	-7	1 1 1 0
+8	0 1 1 1	-8	1 1 1 1

dest	DEST
R2R0	0
R3R1	1

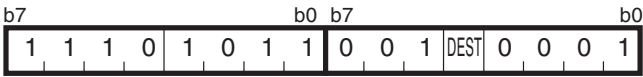
【字节数/周期数】

字节数/周期数	2/3+m
---------	-------

*2 m为移位的位数。

SHA

(4) SHA.L R1H, dest



dest	DEST
R2R0	0
R3R1	1

【字节数/周期数】

字节数/周期数	2/4+m
---------	-------

*1 m为移位的位数。

SHL

(1) SHL.size #IMM, dest



.size	SIZE
.B	0
.W	1

#IMM	IMM4	#IMM	IMM4
+1	0 0 0 0	-1	1 0 0 0
+2	0 0 0 1	-2	1 0 0 1
+3	0 0 1 0	-3	1 0 1 0
+4	0 0 1 1	-4	1 0 1 1
+5	0 1 0 0	-5	1 1 0 0
+6	0 1 0 1	-6	1 1 0 1
+7	0 1 1 0	-7	1 1 1 0
+8	0 1 1 1	-8	1 1 1 1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

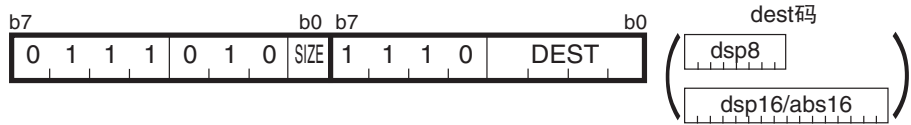
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1+m	2/1+m	2/2+m	3/2+m	3/2+m	4/2+m	4/2+m	4/2+m

*1 m为移位的位数。

SHL

(2) SHL.size R1H, dest



.size	SIZE	dest		DEST	dest		DEST
.B	0	Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
.W	1		R0H/---	0 0 0 1		dsp:8[A1]	1 0 0 1
			R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
			--- /R3	0 0 1 1		dsp:8[FB]	1 0 1 1
		An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
			A1	0 1 0 1		dsp:16[A1]	1 1 0 1
		[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
			[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号 “---” 为不能选择。

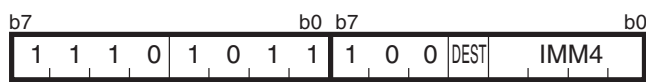
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/2+m	2/2+m	2/3+m	3/3+m	3/3+m	4/3+m	4/3+m	4/3+m

*2 m为移位的位数。

SHL

(3) SHL.L #IMM, dest



#IMM	IMM4	#IMM	IMM4	dest	DEST
+1	0 0 0 0	−1	1 0 0 0	R2R0	0
+2	0 0 0 1	−2	1 0 0 1	R3R1	1
+3	0 0 1 0	−3	1 0 1 0		
+4	0 0 1 1	−4	1 0 1 1		
+5	0 1 0 0	−5	1 1 0 0		
+6	0 1 0 1	−6	1 1 0 1		
+7	0 1 1 0	−7	1 1 1 0		
+8	0 1 1 1	−8	1 1 1 1		

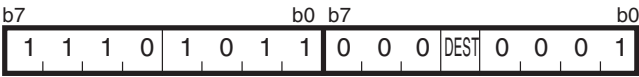
【字节数/周期数】

字节数/周期数	2/3+m
---------	-------

*2 m为移位的位数。

SHL

(4) SHL.L R1H, dest



dest	DEST
R2R0	0
R3R1	1

【字节数/周期数】

字节数/周期数	2/4+m
---------	-------

*1 m为移位的位数。

SMOVB

(1) SMOVB.size



.size	SIZE
.B	0
.W	1

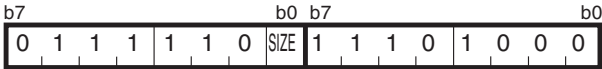
【字节数/周期数】

字节数/周期数	2/5+5×m
---------	---------

*2 m为传送次数。

SMOVF

(1) SMOVF.size



.size	SIZE
.B	0
.W	1

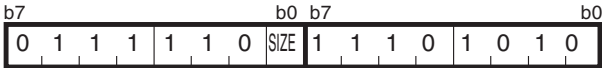
【字节数/周期数】

字节数/周期数	$2/5+5\times m$
---------	-----------------

*1 m为传送次数。

SSTR

(1) SSTR.size



.size	SIZE
.B	0
.W	1

【字节数/周期数】

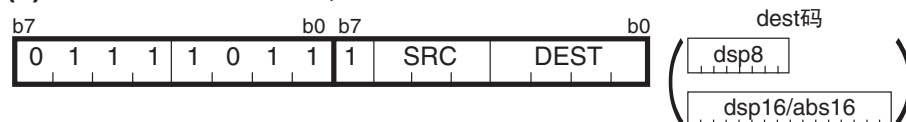
字节数/周期数	$2/3+2\times m$
---------	-----------------

*1 m为传送次数。

STC

(1) STC

src, dest



src	SRC
---	0 0 0
INTBL	0 0 1
INTBH	0 1 0
FLG	0 1 1
ISP	1 0 0
SP	1 0 1
SB	1 1 0
FB	1 1 1

dest		DEST	dest		DEST
Rn	R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

*1 记号“---”为不能选择。

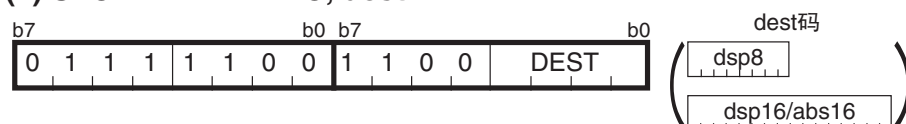
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/1	2/1	2/2	3/2	3/2	4/2	4/2	4/2

STC

(2) STC

PC, dest



dest		DEST	dest		DEST
Rn	R2R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R3R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	---	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	---	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A1A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	---	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

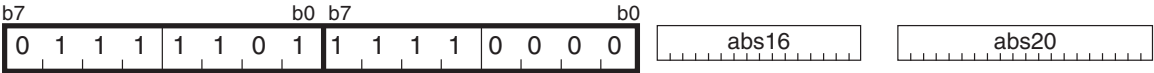
*1 记号“---”为不能选择。

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3

STCTX

(1) STCTX abs16, abs20



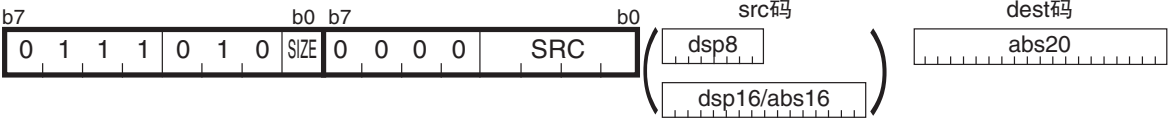
【字节数/周期数】

字节数/周期数	7/11+2×m
---------	----------

*1 m为传送次数。

STE

(1) STE.size src, abs20



.size	SIZE
.B	0
.W	1

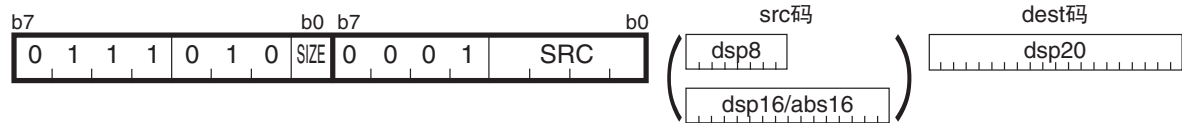
src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	5/3	5/3	5/4	6/4	6/4	7/4	7/4	7/4

STE

(2) STE.size src, dsp:20[A0]



.size	SIZE
.B	0
.W	1

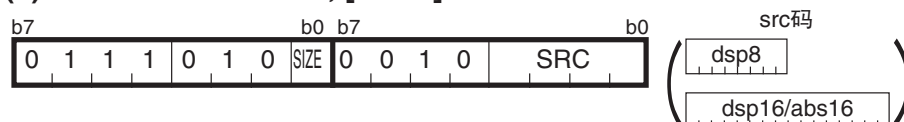
	src	SRC		src	SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	5/3	5/3	5/4	6/4	6/4	7/4	7/4	7/4

STE

(3) STE.size src, [A1A0]



.size	SIZE
.B	0
.W	1

	src	SRC		src	SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4

STNZ

(1) STNZ

#IMM8, dest

b7

1 1 0 1 0 DEST

b0

#IMM8

dest 码

(dsp8 abs16)

dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/2	4/2

*1 在Z标志为“0”时，表中的周期数增加1个周期。

STZ

(1) STZ

#IMM8, dest

b7

1 1 0 0 1 DEST

b0

#IMM8

dest 码

(dsp8 abs16)

dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/2	4/2

*2 在Z标志为“1”时，表中的周期数增加1个周期。



dest		DEST
Rn	ROH	0 1 1
	ROL	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	3/2	4/3	5/3

(1) SUB.size:G #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

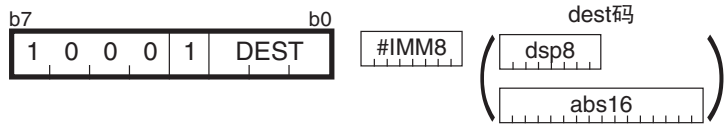
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时, 表中的字节数增加1字节。

SUB

(2) SUB.B:S #IMM8, dest



dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

dest	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	2/1	3/3	4/3

SUB

(3) SUB.size:G src, dest



.size	SIZE
.B	0
.W	1

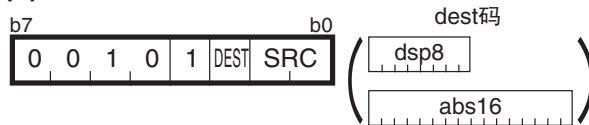
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

SUB

(4) SUB.B:S src, R0L/R0H



src		SRC
Rn	R0L/R0H	0 0
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

dest	DEST
R0L	0
R0H	1

【字节数/周期数】

src	Rn	dsp:8[SB/FB]	abs16
字节数/周期数	1/2	2/3	3/3

TST

(1) TST.size #IMM, dest



.size	SIZE
.B	0
.W	1

dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

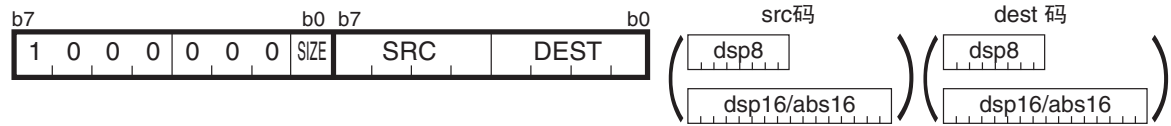
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时, 表中的字节数增加1字节。

TST

(2) TST.size src, dest



.size	SIZE
.B	0
.W	1

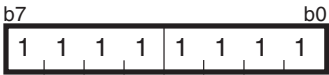
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

UND

(1) UND

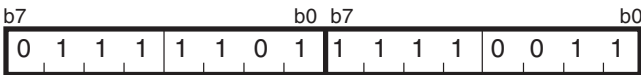


【字节数/周期数】

字节数/周期数	1/20
---------	------

WAIT

(1) WAIT



【字节数/周期数】

字节数/周期数	2/3
---------	-----

XCHG

(1) XCHG.size src, dest



.size	SIZE
.B	0
.W	1

src	SRC
R0L/R0	0 0
R0H/R1	0 1
R1L/R2	1 0
R1H/R3	1 1

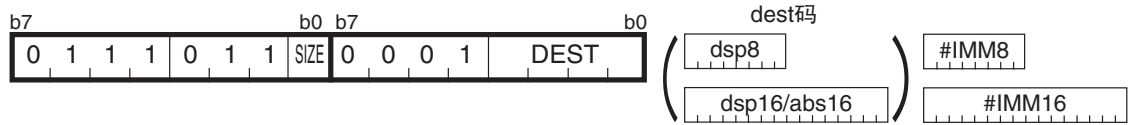
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	2/4	2/4	2/5	3/5	3/5	4/5	4/5	4/5

XOR

(1) XOR.size #IMM, dest



.size	SIZE	dest		DEST	dest		DEST
.B	0	Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
.W	1		R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
			R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
			R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
		An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
			A1	0 1 0 1		dsp:16[A1]	1 1 0 1
		[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
			[A1]	0 1 1 1	abs16	abs16	1 1 1 1

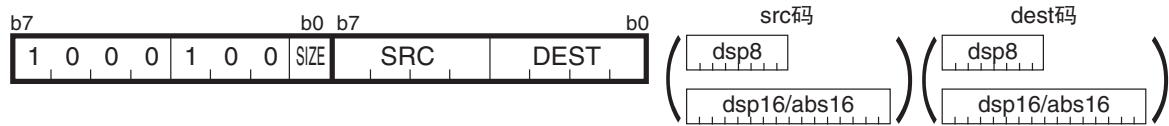
【字节数/周期数】

dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
字节数/周期数	3/2	3/2	3/4	4/4	4/4	5/4	5/4	5/4

*1 在长度说明符(.size)为(.W)时，表中的字节数增加1字节。

XOR

(2) XOR.size src, dest



.size	SIZE
.B	0
.W	1

src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1 1	abs16	abs16	1 1 1 1

【字节数/周期数】

src \ dest	Rn	An	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
Rn	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
An	2/2	2/2	2/3	3/3	3/3	4/3	4/3	4/3
[An]	2/3	2/3	2/4	3/4	3/4	4/4	4/4	4/4
dsp:8[An]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:8[SB/FB]	3/3	3/3	3/4	4/4	4/4	5/4	5/4	5/4
dsp:16[An]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
dsp:16[SB]	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4
abs16	4/3	4/3	4/4	5/4	5/4	6/4	6/4	6/4

第 5 章

中断

- 5.1 中断概要
- 5.2 中断控制
- 5.3 中断响应顺序
- 5.4 从中断程序的返回
- 5.5 中断优先权
- 5.6 多重中断
- 5.7 中断注意事项

5.1 中断概要

如果接受中断请求，就转移到中断向量表中设定的中断程序。请在各中断向量表中设定中断程序的起始地址。中断向量表的详细内容请参照“1.10 向量表”。

5.1.1 中断分类

中断分类如图5.1.1所示，中断源（非屏蔽）和固定向量表如表5.1.1所示。

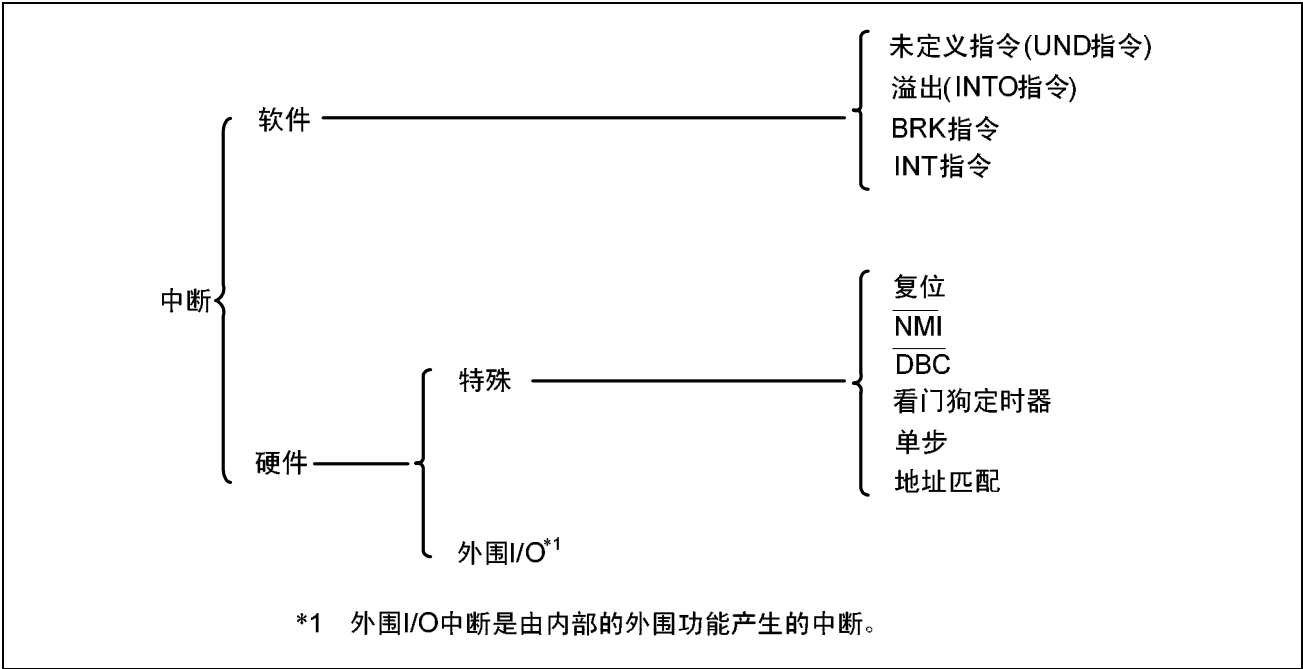


图 5.1.1 中断分类

表 5.1.1 中断源(非屏蔽)和固定向量表

中断源	向量地址 地址(L)~地址(H)	备注
未定义指令	FFFDC ₁₆ ~FFFDF ₁₆	由UND指令中断
溢出	FFFE0 ₁₆ ~FFFE3 ₁₆	由INTO指令中断
BRK指令	FFFE4 ₁₆ ~FFFE7 ₁₆	向量的内容全部为FF ₁₆ 时，从可变量表内的向量所指向的地址开始执行
地址匹配	FFFE8 ₁₆ ~FFFEB ₁₆	有地址匹配中断允许位
单步*1	FFFE _{C16} ~FFFE _{F16}	一般禁止使用
看门狗定时器	FFFF0 ₁₆ ~FFFF3 ₁₆	
DBC*1	FFFF4 ₁₆ ~FFFF7 ₁₆	一般禁止使用
NMI	FFFF8 ₁₆ ~FFFFB ₁₆	由NMI管脚输入引起的外部中断
复位	FFFFC ₁₆ ~FFFFF ₁₆	

*1 调试程序专用中断

- 可屏蔽中断：能通过中断允许标志（I 标志）控制中断的允许（禁止）或者能通过中断优先级改变中断优先权
- 非屏蔽中断：不能通过中断允许标志（I 标志）控制中断的允许（禁止）并且不能通过中断优先级改变中断优先权

5.1.2 软件中断

通过执行指令产生软件中断，软件中断是非屏蔽中断。

(1) 未定义指令中断

如果执行 UND 指令，就产生未定义指令中断。

(2) 溢出中断

在溢出标志（O 标志）为“1”时，如果执行 INTO 指令，就产生溢出中断。根据运算 O 标志变化的指令如下所示：

ABS、ADC、ADCF、ADD、CMP、DIV、DIVU、DIVX、NEG、RMPA、SBB、SHA、SUB

(3) BRK 中断

如果执行 BRK 指令，就产生 BRK 中断。

(4) INT 指令中断

如果指定的软件中断号是0~63,执行INT指令，就产生INT指令中断。另外，由于软件中断号0~31分配给外围I/O中断，因此能通过执行INT指令，执行和外围I/O中断相同的中断程序。

INT指令中断中使用的堆栈指针（SP）根据软件中断号而不同。对于软件中断号0~31，当接受中断请求时将堆栈指针指定标志（U标志）压栈，然后在将U标志清“0”，选择中断堆栈指针（ISP）后，执行中断响应顺序。在从中断程序返回时，恢复接受中断请求前的U标志。对于软件中断号32~63，不切换堆栈指针。

5.1.3 硬件中断

硬件中断有特殊中断和外围I/O中断。

(1) 特殊中断

特殊中断是非屏蔽中断。

- 复位

如果 $\overline{\text{RESET}}$ 管脚输入“L”，就产生复位。

- $\overline{\text{NMI}}$ 中断

如果 $\overline{\text{NMI}}$ 管脚输入“L”，就产生 $\overline{\text{NMI}}$ 中断。

- $\overline{\text{DBC}}$ 中断

因为是调试程序专用中断，一般请不要使用。

- 看门狗定时器中断

由看门狗定时器产生的中断。

- 单步中断

因为是调试程序专用中断，一般请不要使用。如果调试标志（D标志）为“1”时，执行一个指令后产生单步中断。

- 地址匹配中断

在地址匹配中断允许位为“1”，程序的执行地址和地址匹配寄存器的内容一致时，产生地址匹配中断。在地址匹配寄存器中设定指令起始地址以外的地址时，不产生地址匹配中断。

(2) 外围 I/O 中断

外围I/O中断是由内置的外围功能产生的中断。内置的外围功能根据产品种类而不同，因此各自的中断源也因产品种类而不同。中断向量表和INT指令中使用的软件中断号0~31相同。外围I/O中断是可屏蔽中断。有关外围I/O中断的详细内容请参照用户手册。

5.2 中断控制

说明关于可屏蔽中断的允许/禁止以及中断处理优先级的设定。在此说明的内容不适用于非屏蔽中断。

通过中断允许标志（I标志）、中断优先级选择位及处理器中断优先级（IPL）来允许、禁止可屏蔽中断。并且，有无中断请求由中断请求位来表示。中断请求位及中断优先级选择位分配在各中断的中断控制寄存器中。并且，中断允许标志（I标志）及处理器中断优先级（IPL）分配在标志寄存器（FLG）中。

中断控制寄存器的存储器分配及寄存器构成请参照用户手册。

5.2.1 中断允许标志（I标志）

中断允许标志（I标志）控制可屏蔽中断的禁止/允许。如果将该标志置“1”，就允许所有的可屏蔽中断；如果清“0”，就禁止。该标志复位解除后清“0”。

在改变I标志时，改变的内容反映到中断请求接受判定的时序如下：

- 在通过REIT指令改变时，从这一条REIT指令开始反映。
- 在通过FCLR、FSET、POPC、LDC各指令改变时，从下一条指令开始反映。

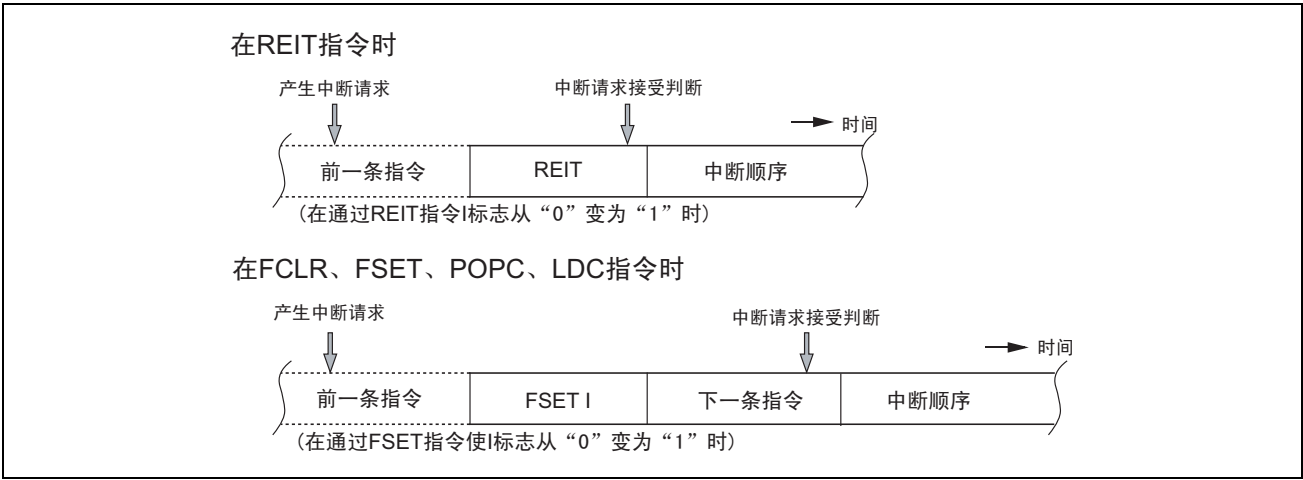


图 5.2.1 在改变 I 标志时，对中断的反映时序

5.2.2 中断请求位

如果产生中断请求，中断请求位就置“1”并在接受中断请求前一直保持。在接受中断请求后，清“0”。并且，该位能通过软件清“0”（请勿写入“1”）。

5.2.3 中断优先级选择位及处理器中断优先级（IPL）

中断优先级能通过中断控制寄存器中的中断优先级选择位来设定。在产生中断请求时，中断优先级与处理器中断优先级（IPL）比较，只有在中断的优先级比处理器中断优先级（IPL）大的时候，才允许该中断。因此，中断优先级如果设定为0级，该中断被禁止。

中断优先级的设定如表5.2.1所示，由处理器中断优先级（IPL）的内容决定的中断允许级如表5.2.2所示。

接受中断请求的条件如下所示：

- 中断允许标志（I标志） = “1”
- 中断请求位 = “1”
- 中断优先级 > 处理器中断优先级（IPL）

中断允许标志（I标志）、中断请求位、中断优先级选择位及处理器中断优先级（IPL）各自独立互不影响。

表 5.2.1 中断优先级的设定

中断优先级选择位	中断优先级	优先权
b2 b1 b0 0 0 0	0级(中断禁止)	—
0 0 1	1级	低 ↓ 高
0 1 0	2级	
0 1 1	3级	
1 0 0	4级	
1 0 1	5级	
1 1 0	6级	
1 1 1	7级	

表 5.2.2 由处理器中断优先级（IPL）的内容决定的中断允许级

IPL	允许的中断优先级
IPL ₂ IPL ₁ IPL ₀ 0 0 0	允许1级以上
0 0 1	允许2级以上
0 1 0	允许3级以上
0 1 1	允许4级以上
1 0 0	允许5级以上
1 0 1	允许6级以上
1 1 0	允许7级以上
1 1 1	禁止所有可屏蔽中断

在改变处理器中断优先级（IPL）或者各中断优先级时，改变的级被反映到中断的时序如下：

- 当通过 REIT 指令改变处理器中断优先级（IPL）时，在从 REIT 指令的最后时钟经过 2 个时钟后执行的指令开始反映。
- 当通过 POPC、LDC、LDIPL 各指令改变处理器中断优先级（IPL）时，在从使用的指令的最后时钟经过 3 个时钟后执行的指令开始反映。
- 当通过 MOV 指令等改变各中断的中断优先级时，在从使用的指令的最后时钟经过 2 个时钟或 3 个时钟后执行的指令开始反映。

M16C/60、M16C/61 群、M16C/20 系列 : 2 个时钟后
M16C/62 群以后的 M16C/60 系列、M16C/Tiny 系列 : 3 个时钟后

5.2.4 中断控制寄存器的改变

(1) 必须在对应该寄存器的中断请求不发生的位置改变中断控制寄存器。在有可能发生中断请求时，必须在禁止中断后改变中断控制寄存器。

(2) 在禁止中断后改变中断控制寄存器的情况下，请注意使用的指令。

改变 IR 位以外的位

在执行指令期间，当发生对应该寄存器的中断请求时，会有 IR 位不为“1”（有中断请求），中断被忽略的情况。如果因此出现问题，请使用以下指令改变寄存器：

对象指令…AND、OR、BCLR、BSET

改变 IR 位

在 IR 位清“0”（无中断请求）时，根据使用的指令，可能有 IR 位不变为“0”的情况。请用 MOV 指令将 IR 位清“0”。

(3) 在使用 I 标志禁止中断时，请参考以下的程序示例设定 I 标志（参考程序示例的中断控制寄存器的更改请参照(2)）。

例 1～例 3 是防止由于受内部总线和指令队列缓冲器的影响，在改变中断控制寄存器前，I 标志变为“1”（允许中断）的方法。

例 1：通过 NOP 指令，等待中断控制寄存器被改变的示例

```
INT_SWITCH1:
    FCLR    I                ; 禁止中断
    AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置“0016”
    NOP                     ; 使用 HOLD 功能时需要 4 个 NOP 指令。
    NOP                     ; NOP 指令的数目请参照硬件手册。
    FSET    I                ; 允许中断
```

例 2：通过虚读，使 FSET 指令等待的示例

```
INT_SWITCH2:
    FCLR    I                ; 禁止中断
    AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置“0016”
    MOV.W   MEM, R0         ; 虚读
    FSET    I                ; 允许中断
```

例 3：通过 POPC 指令，改变 I 标志的示例

```
INT_SWITCH3:
    PUSHC   FLG
    FCLR    I                ; 禁止中断
    AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置“0016”
    POPC    FLG             ; 允许中断
```

5.3 中断响应顺序

以下说明关于在接受中断请求后到执行中断程序为止的中断响应顺序：

如果在指令执行中发生中断请求，就在该指令执行结束后判断优先权，并且从下一个周期转移到中断响应顺序。但是，对于 SMOVB、SMOVF、SSTR 及 RMPA 各指令，如果在指令执行中发生中断请求，就暂时中断指令的运行，转移到中断响应顺序。

中断响应顺序运行如下：

- (1)通过读 00000₁₆ 地址，CPU 获取中断信息（中断号、中断请求级）。然后，该中断的请求位清“0”。
- (2)将中断响应顺序前的标志寄存器（FLG）的内容暂时保存到 CPU 内部的暂存器^{*1}。
- (3)中断允许标志（I 标志）、调试标志（D 标志）及堆栈指针指定标志（U 标志）清“0”（但是，在执行软件中断号 32～63 的 INT 指令时，U 标志不变。）
- (4)将 CPU 内部的暂存器^{*1}的内容压栈。
- (5)将程序计数器（PC）的内容压栈。
- (6)给处理器中断优先级（IPL）设定接受中断的中断优先级。

在中断响应顺序结束后，从中断程序的起始地址执行指令。

*1 用户不能使用。

5.3.1 中断响应时间

中断响应时间是指从产生中断请求开始到执行中断程序内的最初指令的时间。该时间由从中断请求产生点到当时执行的指令结束为止的时间（a）和执行中断响应顺序的时间（b）构成。中断响应时间如图 5.3.1 所示。

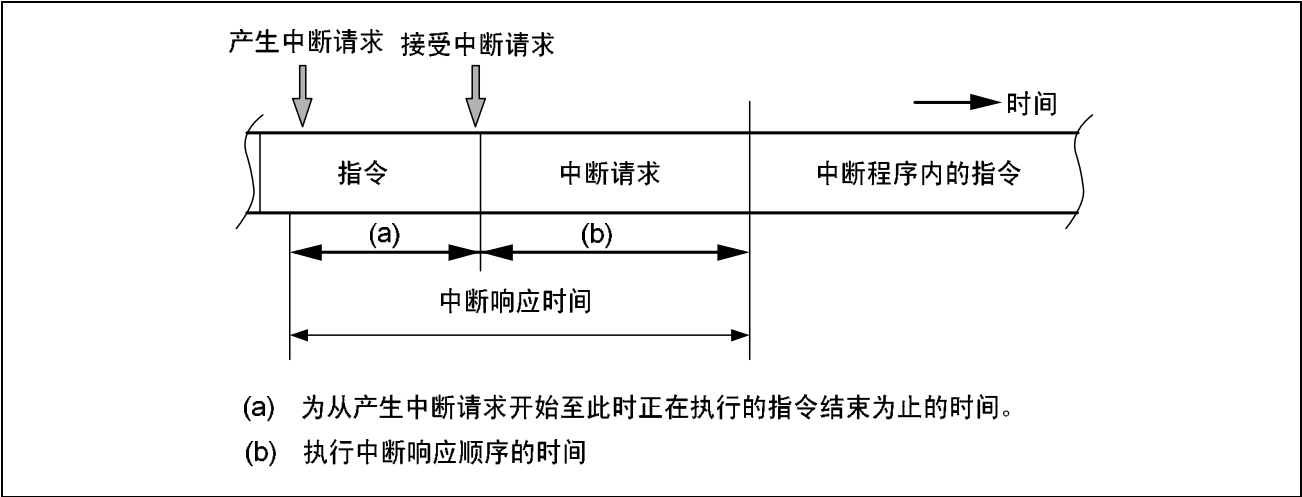


图 5.3.1 中断响应时间

(a) 的时间根据执行的指令而不同。DIVX指令最大30周期（无等待）。

(b) 的时间如下：

表 5.3.1 中断响应顺序执行时间

中断向量的地址	堆栈指针（SP）的值	16位数据总线 无等待	8位数据总线 无等待
偶数	偶数	18周期 ^{*1}	20周期 ^{*1}
偶数	奇数	19周期 ^{*1}	20周期 ^{*1}
奇数 ^{*2}	偶数	19周期 ^{*1}	20周期 ^{*1}
奇数 ^{*2}	奇数	20周期 ^{*1}	20周期 ^{*1}

^{*1} $\overline{\text{DBC}}$ 中断增加 2 个周期，地址匹配中断、单步中断增加 1 个周期。

^{*2} 中断向量的地址请尽量分配在偶数地址。

5.3.2 接受中断请求时的处理器中断优先级（IPL）的变化

如果接受中断请求，可给处理器中断优先级（IPL）设定接受的中断的中断优先级。

如果接受没有中断优先级的中断请求，设定的 IPL 值如表 5.3.2 所示。

表 5.3.2 没有中断优先级的中断和 IPL 的关系

没有中断优先级的中断源	被设定的IPL值
看门狗定时器、 $\overline{\text{NMI}}$	7
复位	0
其他	无变化

5.3.3 寄存器保存

在中断响应顺序中，只将标志寄存器（FLG）和程序计数器（PC）的内容压栈。

压栈的顺序是：首先将程序计数器的高 4 位、FLG 寄存器的高 4 位和低 8 位共计 16 位压栈，然后将程序计数器的低 16 位压栈。中断请求接受前后的堆栈状态如图 5.3.2 所示。

其它必需的寄存器请通过软件在中断程序开始时保存。如果使用 PUSHM 指令，可用 1 条指令保存除了堆栈指针（SP）以外的所有寄存器。

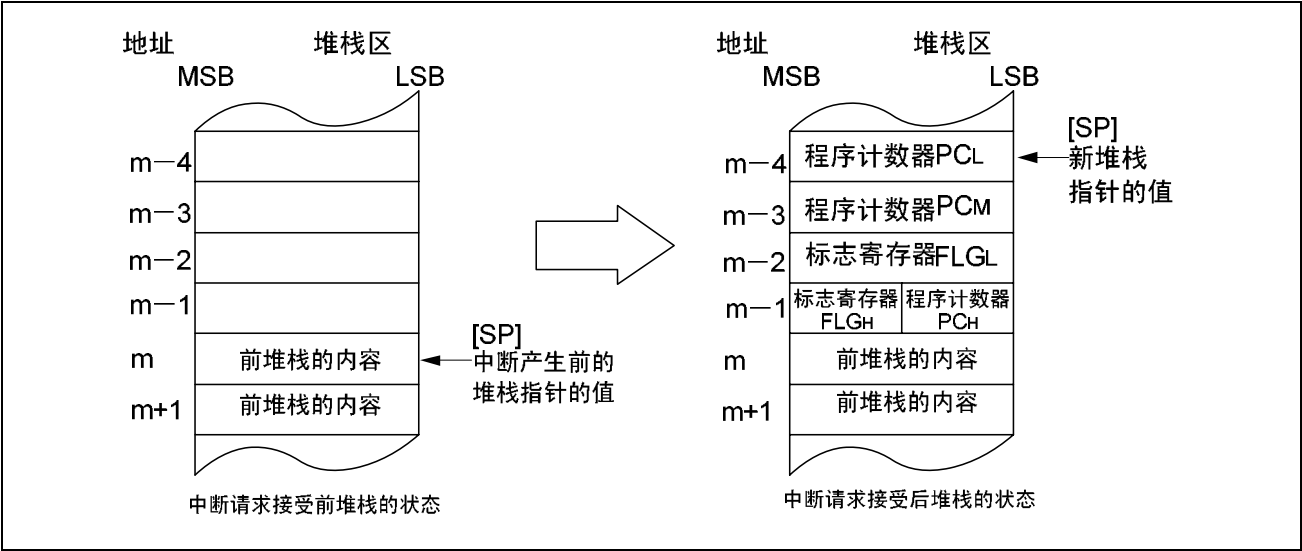


图 5.3.2 中断请求接受前后的堆栈状态

在中断响应顺序下进行的寄存器保存，接受中断请求时的堆栈指针（SP）*1 的内容会有偶数和奇数两种情况。堆栈指针（SP）*1 的内容为偶数时，标志寄存器（FLG）及程序计数器（PC）的内容各 16 位，同时保存。奇数时，分 2 次每 8 位保存。寄存器保存如图 5.3.3 所示。

*1 在执行软件号 32~63 的 INT 指令时，U 标志为指示的堆栈指针。

除此以外为中断堆栈指针（ISP）。

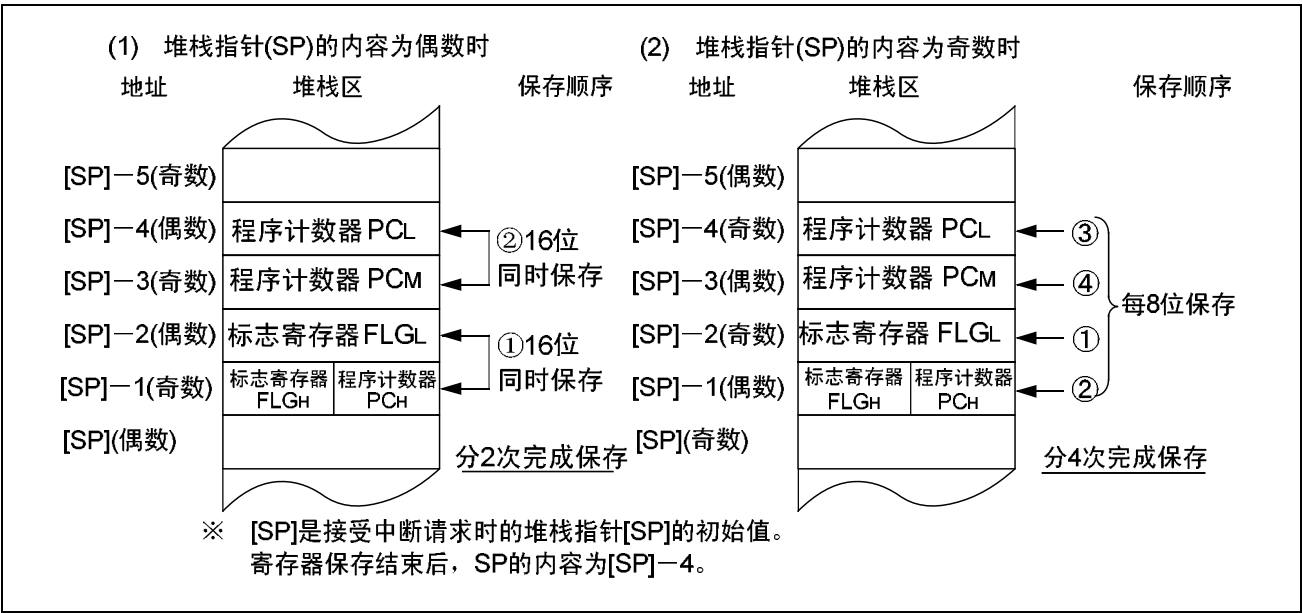


图 5.3.3 寄存器保存

5.4 从中断程序的返回

如果在中断程序的最后执行 REIT 指令，就恢复被压栈的中断响应顺序前的标志寄存器（FLG）及程序计数器（PC）的内容。然后，返回到在接受中断请求前执行的程序，继续执行被中断的处理。

在中断程序内，通过软件保存的寄存器，请在 REIT 指令执行前用 POPM 指令等恢复。

5.5 中断优先权

在同一采样时刻（调查是否存在中断请求的时序）存在 2 个以上的中断请求时，就接受优先权高的中断。

可屏蔽中断（外围 I/O 中断）的优先权可通过中断优先级选择位设定任意的优先权。但是，在中断优先级为相同设定值的情况下，接受由硬件设定的优先度*1 高的中断。

复位（复位作为优先权最高的中断来处理）、看门狗定时器中断等，非屏蔽中断的优先权由硬件设定。由硬件设定的中断优先权如图 5.5.1 所示。

软件中断不受中断优先权的影响。如果执行指令，就一定转移到中断程序。

*1 根据产品种类不同而不同，请参照用户手册。

复位>NMI>DBC>看门狗定时器>外围I/O>单步>地址匹配

图 5.5.1 由硬件设定的中断优先权

5.6 多重中断

转移到中断程序时的状态如下所下：

- 中断允许标志(I标志)为“0”(中断禁止状态)
- 接受的中断的中断请求位为“0”
- 处理中断优先级(IPL)为接受的中断的中断优先级

在中断程序中，通过将中断允许标志(I标志)置“1”，能接受高于处理中断优先级(IPL)的优先级的中断请求。多重中断如图 5.6.1 所示。

另外，保持因优先权低而不被接受的中断请求。然后，在通过 REIT 指令恢复 IPL 并且判断中断优先权时，如果为以下的状态，就接受被保持的中断请求。

被保持的中断请求的中断优先级 > 被恢复的处理中断优先级(IPL)

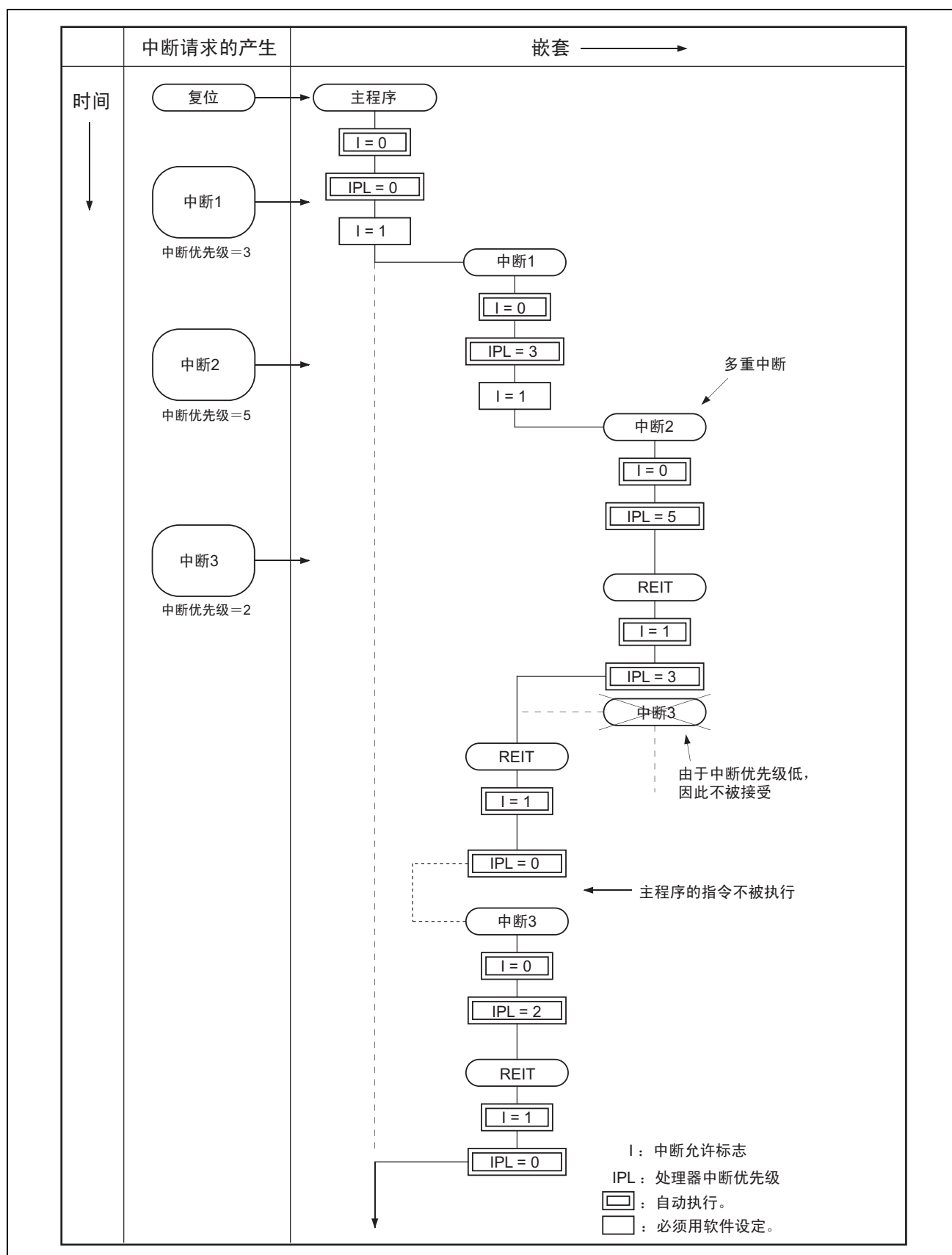


图 5.6.1 多重中断

5.7 中断注意事项

5.7.1 0000016 地址的读出

请不要通过程序读 0000016 地址。在接受可屏蔽中断的中断请求时，CPU 在中断响应顺序中从 0000016 地址读取中断信息（中断号和中断请求级）。此时，被接受的中断的 IR 位为“0”。

如果通过程序读 0000016 地址，在被允许的中断中，最高优先级的中断的 IR 位将变为“0”。因此，中断可能会被取消或者产生无法预料的中断。

5.7.2 SP 的设定

在接受中断前，给 SP（USP、ISP）设定值。复位后，SP（USP、ISP）为“000016”。因此，如果在给 SP（USP、ISP）设定值前接受中断，会成为失控的主要原因。

尤其是，使用 $\overline{\text{NMI}}$ 中断时，请在程序的起始设定 ISP 的值。仅在复位后开始的 1 个指令中，禁止包括 $\overline{\text{NMI}}$ 中断在内的所有中断。

5.7.3 中断控制寄存器的改变

(1) 请在对应该寄存器的中断请求不发生的位置改变中断控制寄存器。在有可能发生中断请求时，请在禁止中断后改变中断控制寄存器。

(2) 在禁止中断后改变中断控制寄存器时，请注意使用的指令。

IR 位以外的位的改变

在执行指令期间，当发生对应该寄存器的中断请求时，IR 位可能不为“1”（有中断请求），中断可能被忽略。如果因此出现问题，请使用以下指令改变寄存器：

对象指令…AND、OR、BCLR、BSET

改变 IR 位

在 IR 位清“0”（无中断请求）时，根据使用的指令，IR 位可能不为“0”。请用 MOV 指令将 IR 位清“0”。

(3) 在使用 I 标志禁止中断时，请参考以下的程序示例设定 I 标志（参考程序示例的中断控制寄存器改变请参照(2)）。

例 1～例 3 是防止由于受内部总线和指令队列缓冲器的影响，在改变中断控制寄存器前，I 标志为“1”（允许中断）的方法。

例 1: 通过 NOP 指令, 等待中断控制寄存器被改变的示例

INT_SWITCH1:

```
FCLR    I                ; 禁止中断
AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置 “0016”
NOP                     ; 使用 HOLD 功能时需要 4 个 NOP 指令。
NOP                     ; NOP 指令的数目请参照硬件手册。
FSET    I                ; 允许中断
```

例 2: 通过虚读, 使 FSET 指令等待的示例

INT_SWITCH2:

```
FCLR    I                ; 禁止中断
AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置 “0016”
MOV.W   MEM, R0         ; 虚读
FSET    I                ; 允许中断
```

例 3: 通过 POPC 指令, 改变 I 标志的示例

INT_SWITCH3:

```
PUSHC   FLG
FCLR    I                ; 禁止中断
AND.B   #00H, 0055H     ; 将 TA0IC 寄存器置 “0016”
POPC    FLG             ; 允许中断
```


第 6 章

周期数的计算

6.1 指令队列缓冲器

6.1 指令队列缓冲器

M16C/60、M16C/20、M16C/Tiny 系列具有 4 段（4 字节）的指令队列缓冲器。在 CPU 能使用总线的状态下，如果指令队列缓冲器为空，指令码就被取入指令队列缓冲器，将此称为预取指令。CPU 边读取存放在指令队列缓冲器中的指令码（取指令）边执行程序。

第 4 章说明的周期数是指在指令队列缓冲器中已备齐了指令码的状态下，对 16 位总线连接的存储器（包含内部存储器）从偶数地址以无软件等待或 $\overline{\text{RDY}}$ 等的无等待模式读写数据时的周期数。在下列情况下，将多于在手册中记载的周期数：

- 在指令队列缓冲器中没有备齐 CPU 所需的指令码。
到备齐执行所需的指令码为止读取指令码。在下面的情况下，读周期数还会增加：
 - (1) 当从存在软件等待、 $\overline{\text{RDY}}$ 等的等待周期的区域读入指令码时
读周期数将只增加等待周期数。
 - (2) 当从连接到 8 位总线的存储器中读入指令码时
与 16 位总线相比周期数增加。
- 对存在软件等待、 $\overline{\text{RDY}}$ 等的等待周期的区域读写数据。
周期数将只增加等待周期数。
- 对于连接 8 位总线的存储器读写 16 位数据。
对于 1 个数据的读写，读写 2 次。因此，对应 1 个数据的读写周期数，增加 1 周期。
- 对于连接 16 位总线的存储器从奇数地址读写 16 位数据。
对于 1 个数据的读写，读写 2 次。因此，对应 1 个数据的读写周期数，增加 1 周期。

另外，在同一时序中发生预取指令和数据存取时，数据存取优先。

并且，在指令队列缓冲器内存在 3 字节以上的指令码时，就判定指令队列缓冲器不为空，不预取指令。

指令队列缓冲器的工作和 CPU 的执行周期例如图 6.1.1～图 6.1.8 所示。

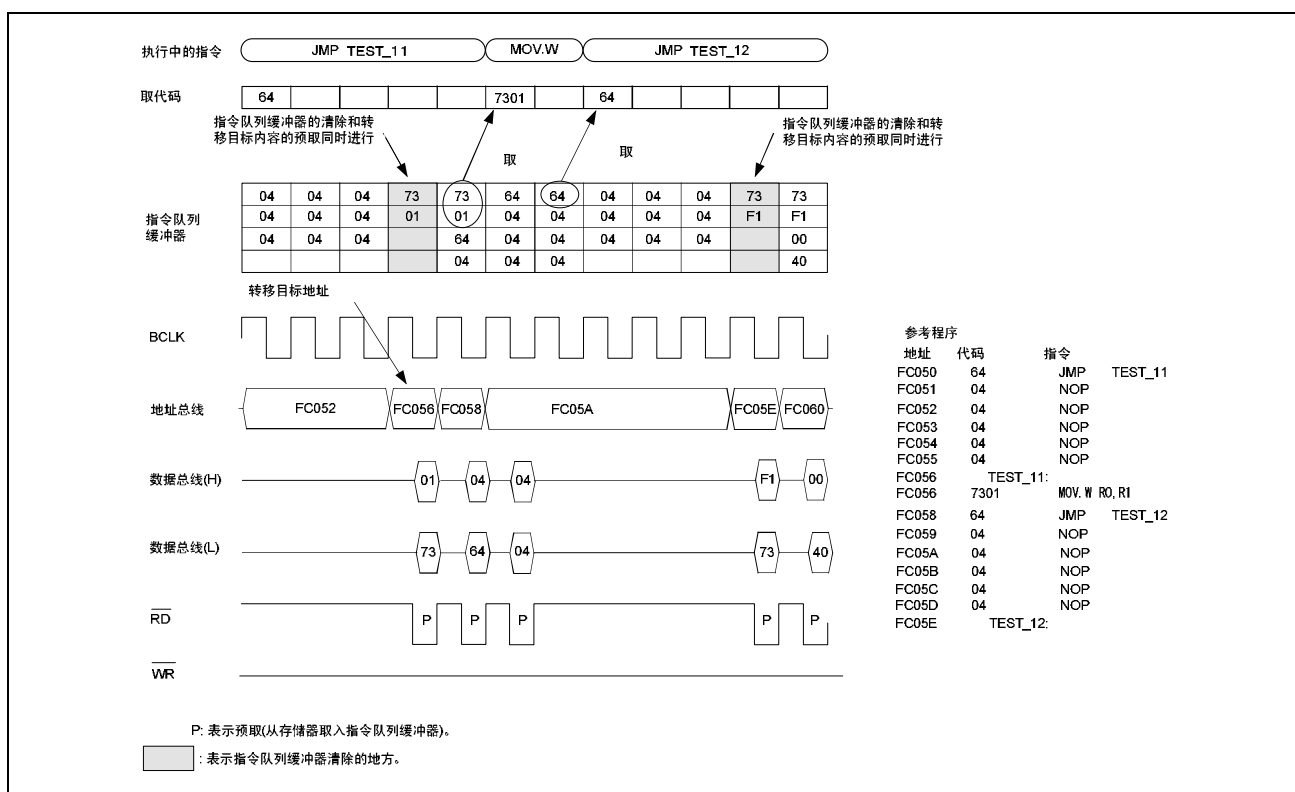


图6.1.1 寄存器从偶数地址开始传送指令的情况（程序区：16位总线无等待，数据区：16位总线无等待）

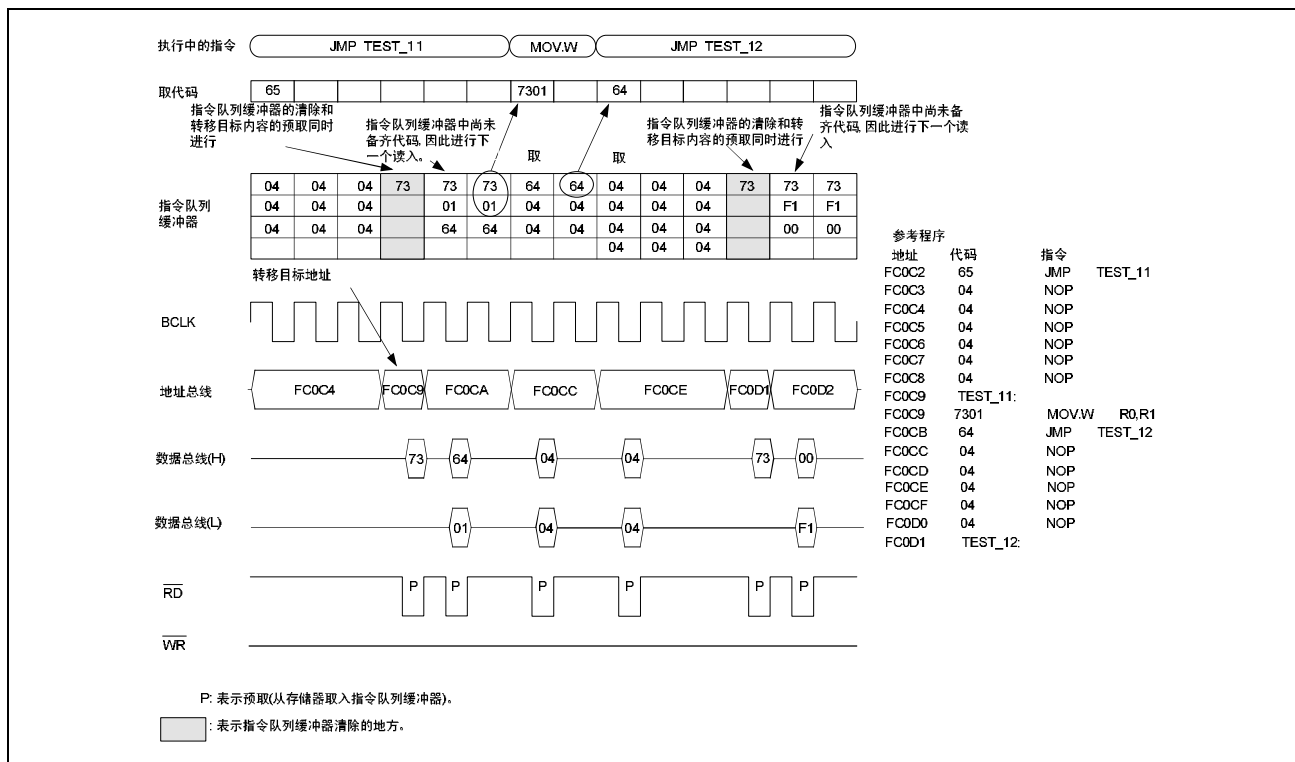


图6.1.2 寄存器从奇数地址开始传送指令的情况（程序区：16位总线无等待，数据区：16位总线无等待）

第6章 周期数的计算

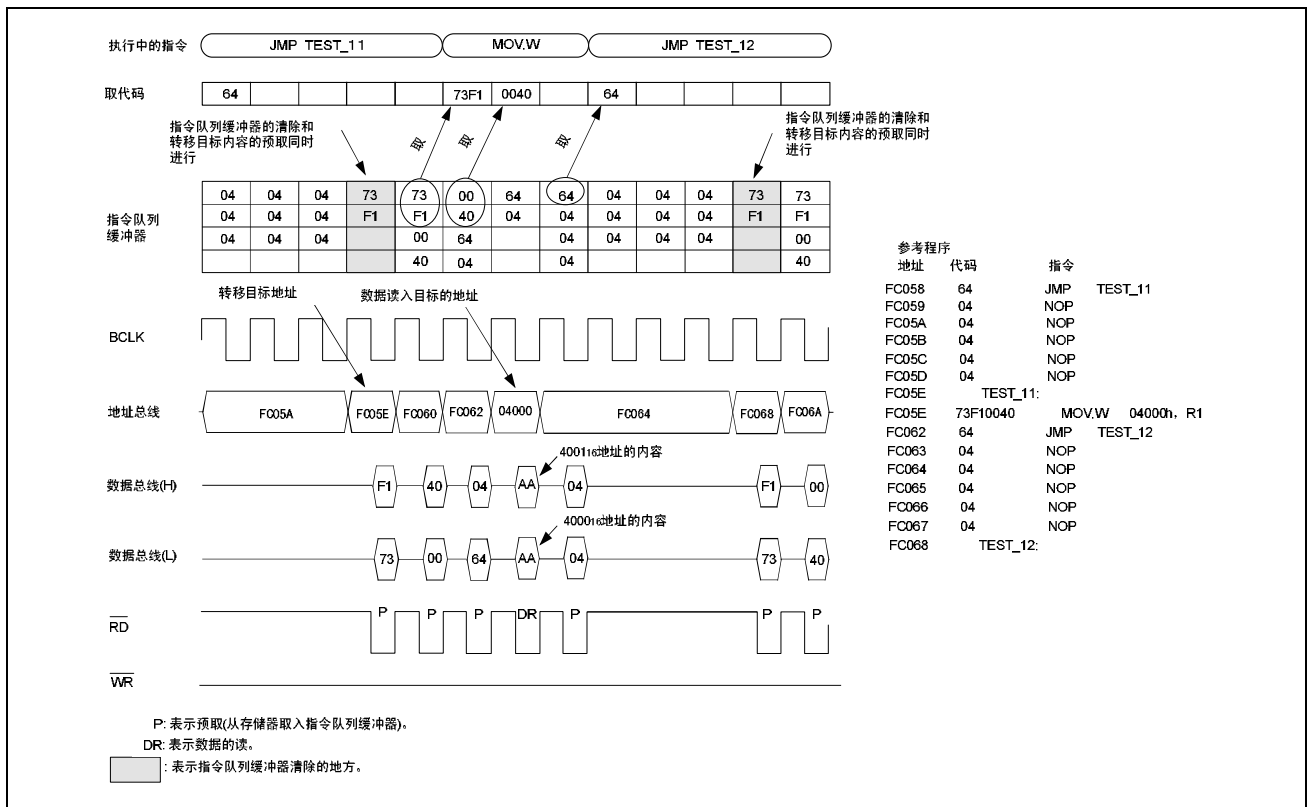


图6.1.3 从偶数地址开始读入偶数地址指令的情况（程序区：16位总线无等待，数据区：16位总线无等待）

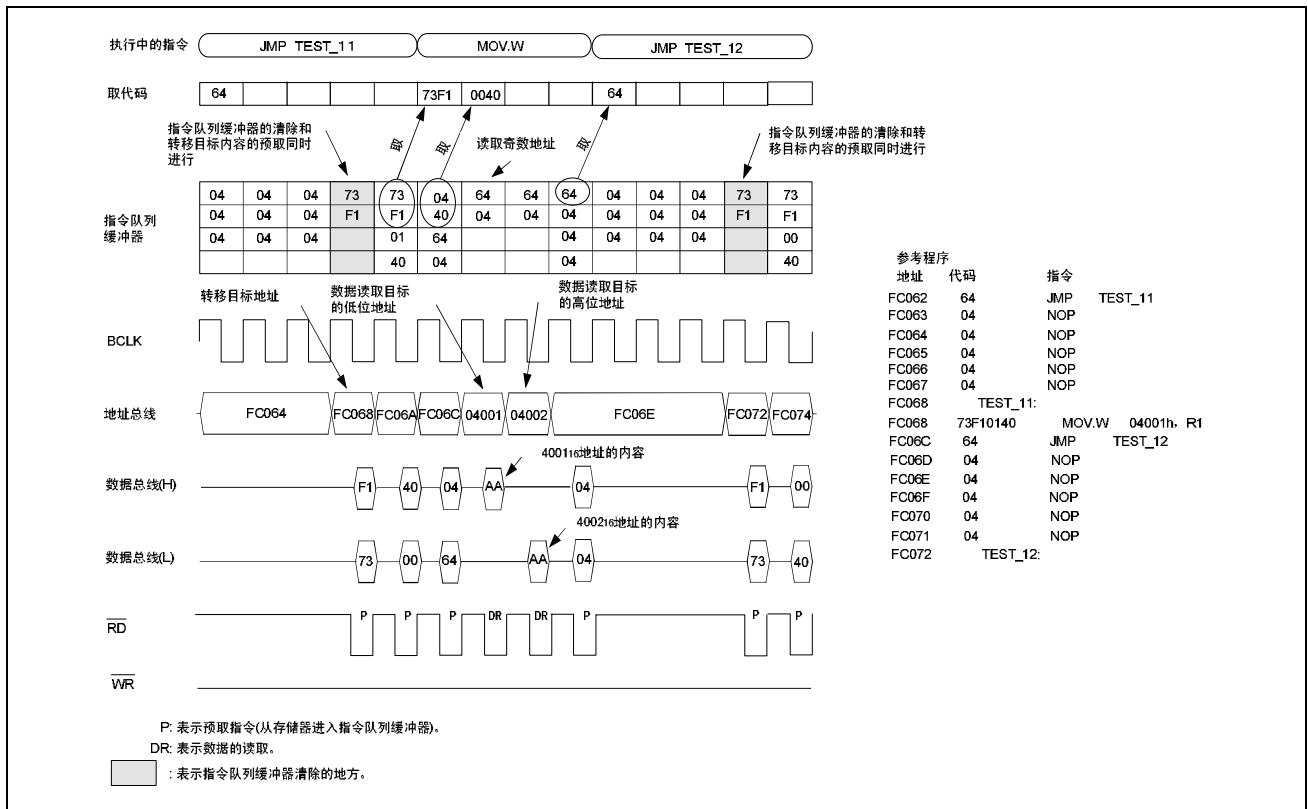


图6.1.4 从偶数地址开始读入奇数地址指令的情况（程序区：16位总线无等待，数据区：16位总线无等待）

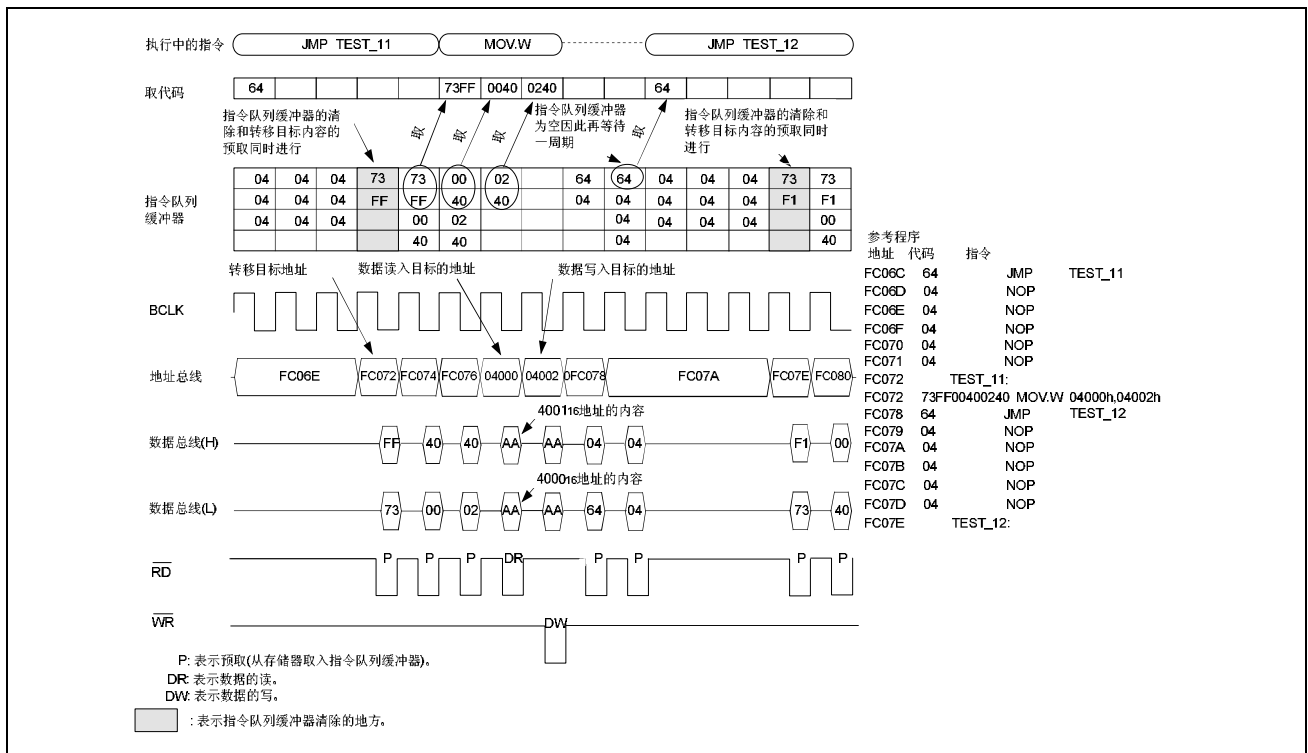


图6.1.5 从偶数地址开始偶数地址到偶数地址数据传送指令的情况（程序区：16位总线无等待，数据区：16位总线无等待）

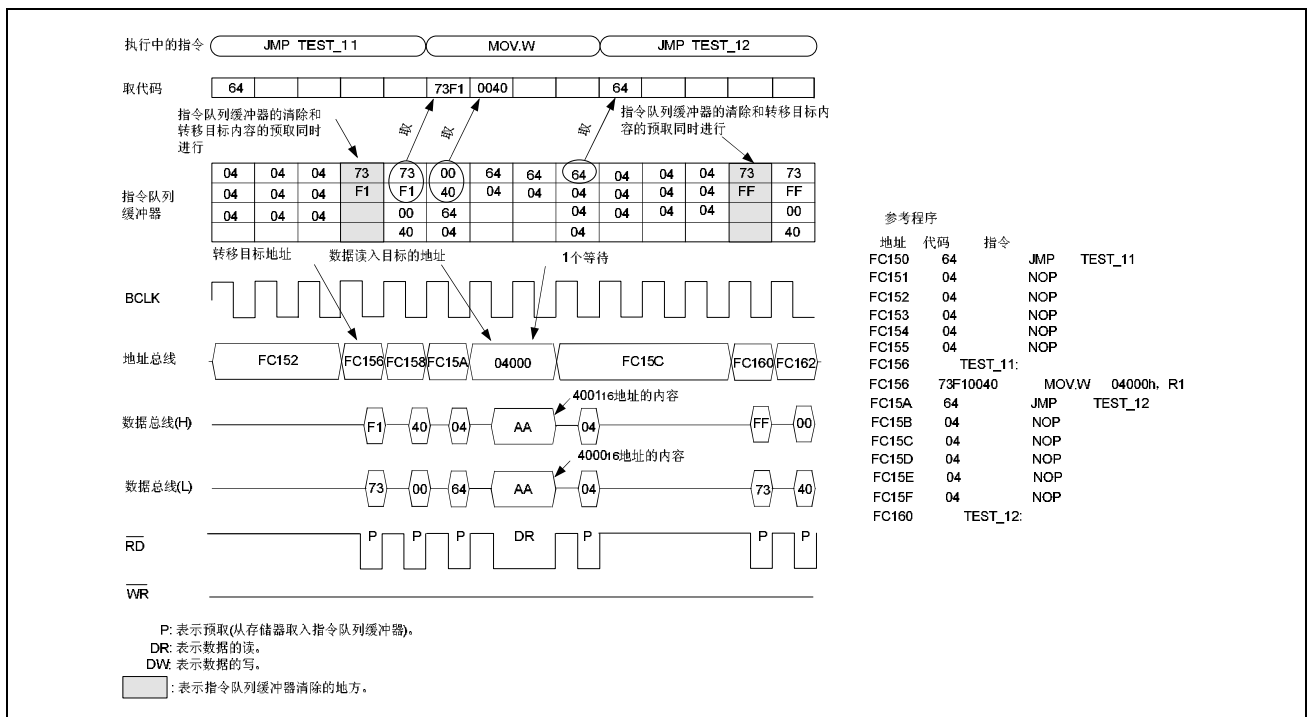


图6.1.6 从偶数地址开始读入偶数地址指令的情况（程序区：16位总线无等待，数据区：16位总线有等待）

第 6 章 周期数的计算

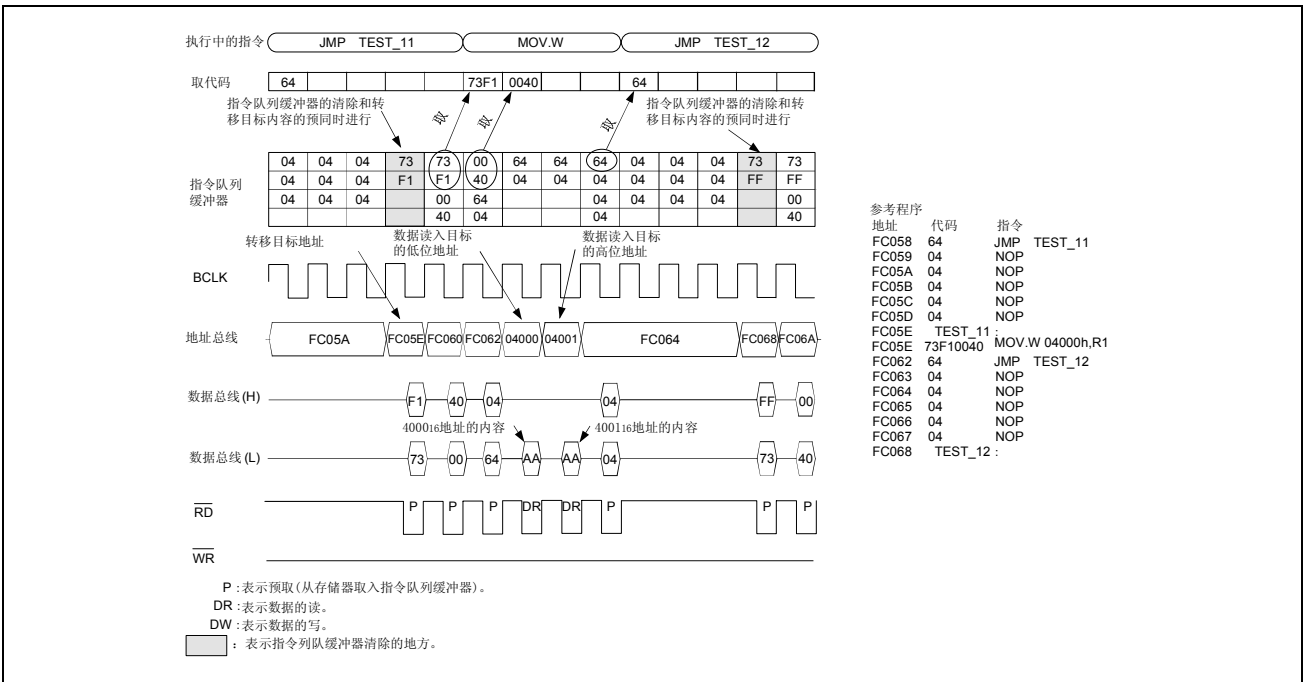


图6.1.7 对8位总线的存储器开始读入指令的情况（程序区：16位总线无等待，数据区：8位总线无等待）

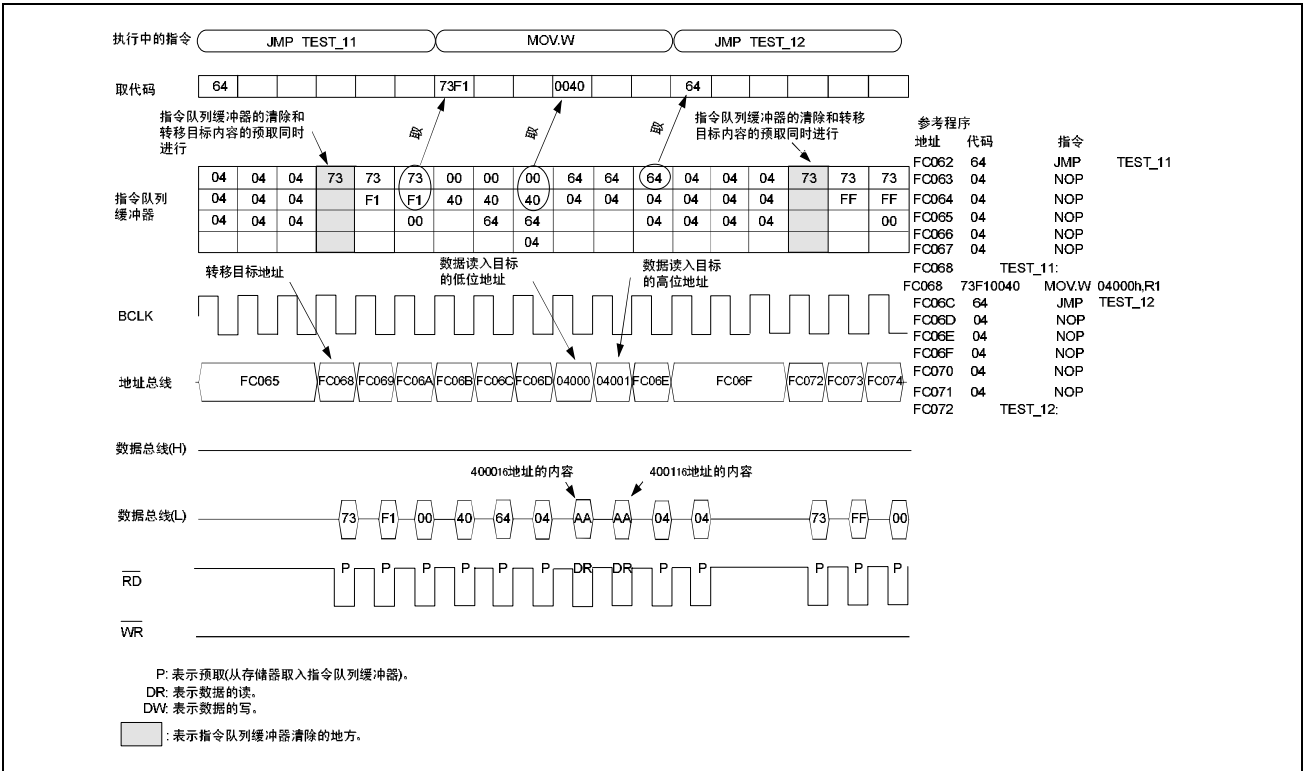


图6.1.8 对8位总线的存储器开始读入指令的情况（程序区：8位总线无等待，数据区：8位总线无等待）

Q&A

为了最大限度地正确使用 M16C 族，其信息以 Q&A 形式记载如下。

Q&A 原则上将 1 个问题和此问题的回答记载于 1 页内，各页的上段为问题、下段为此问题的回答（在 1 个问题和此问题的回答超过 2 页以上时，将在右下角记载页数）。

另外，在各页的右上角表示有关此页内容的主要功能。

Q

如何分别使用静态基址寄存器(SB)和帧基址寄存器(FB)?

A

SB和FB具有相同的功能，因此，在用汇编语言编程时，能自由使用。在用C语言编程时，FB作为栈帧基址寄存器使用。

Q

在程序执行中，能否改变中断表寄存器(INTB)的值？

A

可以。但是，如果在改变INTB的值的过程中发生中断请求，单片机就可能会失控。因此，建议不要在程序执行中频繁地改变INTB的值。

Q

用户堆栈指针(USP)和中断堆栈指针(ISP)的不同点和作用是什么？

A

在使用OS时使用USP。当存在多个任务时，OS按各任务确保保存寄存器的堆栈区。另外，由于在执行这些任务过程中会产生中断，因此必需按各任务确保由中断使用的堆栈区。在此，如果使用USP和ISP，各任务就能共享用于中断的堆栈，能有效地使用堆栈区。

Q

在以绝对寻址使用位指令时，变成什么样的指令码？

A

说明BSET bit,base:16的情况。

此指令为4字节指令。指令码的高位2个字节表示操作码，低位2个字节为寻址方式，表现bit,base:16。低位2个字节和bit,base:16的关系如下：

低位2个字节 = $\text{base:16} \times 8 + \text{bit}$

例如，在BSET 2,0AH（将000A₁₆地址的位2置1）的情况下，低位2个字节为 $A \times 8 + 2 = 52\text{H}$ 。另外，在BSET 18,8H（将从0008₁₆地址的位0开始的第18位置1）的情况下，低位2个字节为 $8 \times 8 + 18 = 52\text{H}$ ，成为和BSET 2,AH相同的指令码。

base:16 $\times$ 8+bit的最大值FFFFH表示1FFF₁₆地址的位7。这是在以绝对寻址使用位指令时能指定的最大位。

Q

DIV指令和DIVX指令的不同点是什么？

A

DIV指令和DIVX指令都为带符号的除法指令，但是余数的符号不同。

DIV指令的余数的符号和被除数的符号相同，而DIVX指令的余数的符号和除数的符号相同。

另外，一般地在商、除数、被除数以及余数之间存在以下的关系：

被除数=除数×商+余数

因为余数符号的不同结果，在正整数除以负整数时或者在负整数除以正整数时根据双方的指令商也不同。

例如，在10除以-3时，用DIV指令产生商为-3、余数为+1的结果，而用DIVX指令则产生商为-4、余数为-2的结果。

另外，在-10除以+3时，用DIV指令产生商为-3、余数为-1的结果，而用DIVX命令指令则产生商为-4、余数为+2的结果。

词汇表

有关在本软件手册中使用的词汇说明如下。此词汇表只适用本软件手册。

词汇	意义	相关语句
LSB	Least Significant Bit的略称。 表示数据的最低位。	MSB
MSB	Most Significant Bit的略称。 表示数据的最高位。	LSB
解压缩	将结合的项目或者被压缩的信息分离。 常用于表现将8位的信息分离成低4位和高4位、或者将16位的信息分离成低8位和高8位。	压缩
SFR区	SFR是Special Function Register的略称。 表示内藏在单片机中的外围电路的控制位和分配控制寄存器的区域。	
运算	为传送、比较、位处理、移位、循环移位、算术运算、逻辑运算、转移的总称。	
溢出	表示运算结果超过能表现的最大值。 (数字溢出)	
操作码	指令码中表示指令操作的码。	操作数
操作数	指令码中表示指令操作对象的码。	操作码

词汇	意义	相关语句
扩展区	在M16C/60、M16C/20、M16C/Tiny系列，表示10000 ₁₆ ～FFFFF ₁₆ 地址的区域。	
进位	向下一个高位移动一个数字。	借位
环境	指由程序使用的寄存器。	
执行地址	修改后的实际地址。	
移出	将寄存器的内容向左或向右移动而产生溢出。	
10进制加法	以10进制进行加法运算。	
栈帧	C语言的函数使用的自动变量的区域。	
字符串	字符序列。	

词汇	意义	相关语句
零扩展	在扩展数据长时，将被扩展的高位置0。 例如，当将FF ₁₆ 零扩展成16位时，变成00FF ₁₆ 。	
位移量	开始位置和最后位置的差。	
压缩	指结合数据。 常用于表现将2个4位数据结合成8位、或者将2个8位数据结合成16位。	解压缩
符号扩展	在扩展数据长时，将被扩展的高位按照符号位的状态进行扩展。 例如，当将FF ₁₆ 符号扩展时，变成FFFF ₁₆ ； 当将0F ₁₆ 符号扩展时，变成000F ₁₆ 。	
符号位	表示正或负的位（最高位）	
借位	向下一个低位移动一个数字。	进位
微指令	是由源语言编写的1条指令，在编译成机器码时，由几条机器码指令表现的指令。	

符号表

有关在本软件手册中使用的符号说明如下。此符号表只适用本软件手册。

符号	意义
←	从右侧传送到左侧
↔	右侧和左侧交换
+	加法
—	减法
×	乘法
÷	除法
∧	逻辑与
∨	逻辑或
⊕	逻辑异或
—	逻辑非
dsp16	16 位的位移量
dsp20	20 位的位移量
dsp8	8 位的位移量
EVA()	() 内表示有效地址
EXT()	() 内的符号扩展
(H)	寄存器或者存储器的高位字节
H4:	8 位寄存器或者 8 位存储器的高 4 位
	绝对值
(L)	寄存器或者存储器的低位字节
L4:	8 位寄存器或者 8 位存储器的低 4 位
LSB	Least Significant Bit 的略称
M()	() 内表示存储器的内容
(M)	寄存器或者存储器的中位字节
MSB	Most Significant Bit 的略称
PCH	程序计数器的高位字节
PCML	程序计数器的中位字节和低位字节
FLGH	标志寄存器的高 4 位
FLGL	标志寄存器的低 4 位

索引

A

A0/A1 ... 5
A1A0 ... 5

B

B标志 ... 6
半字节（4位）数据 ... 16
标志变化 ... 37
标志寄存器 ... 5

C

C标志 ... 6
操作 ... 37
操作数 ... 35, 38
长度说明符 ... 35
程序计数器 ... 5
处理器中断优先级 ... 7
存储器的位 ... 12
存储器内的数据分配 ... 17

D

dest ... 18
D标志 ... 6
地址寄存器 ... 5
地址空间 ... 3
堆栈指针 ... 5
堆栈指针指定标志 ... 6

F

FB ... 5
FLG ... 5
非屏蔽中断 ... 249
符号标志 ... 6
复位 ... 9

G

功能 ... 37
固定向量表 ... 19

I

INTB ... 5
IPL ... 7
ISP ... 5
I标志 ... 6

J

记述例 ... 37
进位标志 ... 6
静态基址寄存器 ... 5

K

可变向量表 ... 20

L

零标志 ... 6

N

能选择的src / dest (label) ... 37

O

O标志 ... 6

P

PC ... 5
屏蔽中断 ... 249

	R	整数 ... 10
R0/R1/R2/R3 ... 4		指令格式 ... 18
R0H/R1H ... 4		指令格式说明符 ... 35
R0L/R1L ... 4		指令码 ... 139
R2R0 ... 4		周期数 ... 139
R3R1 ... 4		助记符 ... 35, 38
软件中断号 ... 20		字符串 ... 15
		字节（8位）数据 ... 16
	S	专用页号 ... 19
SB ... 5		专用页向量表 ... 19
src ... 18		
S标志 ... 6		
数据寄存器... 4		
数据类型 ... 10		
	T	
调试标志 ... 6		
	U	
USP ... 5		
U标志 ... 6		
	X	
相关指令 ... 37		
寻址方式 ... 22		
	Y	
溢出标志... 6		
用户堆栈指针 ... 5		
语法... 35, 38		
	Z	
Z标志 ... 6		
帧基址寄存器 ... 5		

瑞萨 16位单片机

软件手册

M16C/60、M16C/20、M16C/Tiny系列

Publication Date: Rev.1.00, Jul. 12, 2006

Published by: Sales Strategic Planning Div.
Renesas Technology Corp.

Edited by: Customer Support Department
Global Strategic Communication Div.
Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan



RENESAS SALES OFFICES

<http://www.renesas.com>

Refer to "<http://www.renesas.com/en/network>" for the latest and detailed information.

Renesas Technology America, Inc.

450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.

Unit 204, 205, AZIACenter, No.1233 Lujiiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.

7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.

10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.

1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.

Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd

Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

M16C/60、 M16C/20、
M16C/Tiny 系列



瑞萨电子株式会社

RCJ09B0021-0100