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.

Software Manual

16

R8C/Tiny系列

瑞萨16 位单片机

瑞萨电子
www.renesas.com

Rev.1.00 2004.07

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价。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、交通车辆、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R8C/Tiny系列的软件手册。能用于具有R8C/Tiny系列CPU核心的所有产品。
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：

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。

在本手册的后面有Q&A、词汇表、符号表以及索引。

M16C族的有关文献

对M16C族，提供了如下的文献：

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
应用注意事项	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

目　录

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	138	DIVU	68	171
ADC	40	138	DIVX	69	172
ADCF	41	140	DSBB	70	173
ADD	42	140	DSUB	71	175
ADJNZ	44	146	ENTER	72	177
AND	45	147	EXITD	73	178
BAND	47	150	EXTS	74	178
BCLR	48	150	FCLR	75	179
BM*Cnd*	49	152	FSET	76	180
BMEQ/Z	49	152	INC	77	180
BMGE	49	152	INT	78	181
BMGEU/C	49	152	INTO	79	182
BMGT	49	152	J*Cnd*	80	182
BMGTU	49	152	JEQ/Z	80	182
BMLE	49	152	JGE	80	183
BMLEU	49	152	JGEU/C	80	182
BMLT	49	152	JGT	80	183
BMLTU/NC	49	152	JGTU	80	182
BMN	49	152	JLE	80	183
BMNE/NZ	49	152	JLEU	80	182
BMNO	49	152	JLT	80	183
BMO	49	152	JLTU/NC	80	182
BMPZ	49	152	JN	80	182
BNAND	50	153	JNE/NZ	80	182
BNOR	51	154	JNO	80	183
BNOT	52	154	JO	80	183
BNTST	53	155	JPZ	80	182
BNXOR	54	156	JMP	81	183
BOR	55	156	JMPI	82	185
BRK	56	157	JSR	83	187
BSET	57	157	JSRI	84	188
BTST	58	158	LDC	85	189
BTSTC	59	159	LDCTX	86	190
BTSTS	60	160	LDE	87	191
BXOR	61	160	LDINTB	88	192
CMP	62	161	LDIPL	89	193
DADC	64	165	MOV	90	193
DADD	65	167	MOVA	92	200
DEC	66	169			
DIV	67	170			

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*Dir*	93	201	ROT	112	220
MOVHH	93	201	RTS	113	221
MOVHL	93	201	SBB	114	222
MOVLH	93	201	SBJNZ	115	224
MOVLL	93	201	SHA	116	225
MUL	94	203	SHL	117	228
MULU	95	205	SMOVB	118	230
NEG	96	207	SMOVF	119	231
NOP	97	207	SSTR	120	231
NOT	98	208	STC	121	232
OR	99	209	STCTX	122	233
POP	101	211	STE	123	233
POPC	102	213	STNZ	124	235
POPM	103	213	STZ	125	235
PUSH	104	214	STZX	126	236
PUSHA	105	216	SUB	127	236
PUSHC	106	216	TST	129	239
PUSHM	107	217	UND	130	241
REIT	108	217	WAIT	131	241
RMPA	109	218	XCHG	132	242
ROLC	110	218	XOR	133	243
RORC	111	219			

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	90	193
	MOVA	有效地址的传送	92	200
	MOVDir	4 位数据的传送	93	201
	POP	寄存器/存储器的恢复	101	211
	POPM	多个寄存器的恢复	103	213
	PUSH	寄存器/存储器/立即数的保存	104	214
	PUSHA	有效地址的保存	105	216
	PUSHM	多个寄存器的保存	107	217
	LDE	从扩展数据区的传送	87	191
	STE	向扩展数据区的传送	123	233
	STNZ	条件传送	124	235
	STZ	条件传送	125	235
	STZX	条件传送	126	236
	XCHG	交换	132	242
位处理	BAND	位逻辑与	47	150
	BCLR	位清除	48	150
	BM*Cnd*	条件位传送	49	152
	BNAND	反转位的逻辑与	50	153
	BNOR	反转位的逻辑或	51	154
	BNOT	位反转	52	154
	BNTST	反转位的测试	53	155
	BNXOR	反转位的逻辑或	54	156
	BOR	位逻辑或	55	156
	BSET	置位	57	157
	BTST	位测试	58	158
	BTSTC	位测试和清除	59	159
	BTSTS	位测试和置位	60	160
	BXOR	位逻辑异或	61	160
移位	ROLC	带进位的循环左移	110	218
	RORC	带进位的循环右移	111	219
	ROT	循环移位	112	220
	SHA	算术移位	116	225
	SHL	逻辑移位	117	228
算术	ABS	绝对值	39	138
	ADC	带进位的加法运算	40	138
	ADCF	进位标志的加法运算	41	140
	ADD	无进位的加法运算	42	140
	CMP	比较	62	161
	DADC	带进位的 10 进制加法运算	64	165

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	167
	DEC	递减	66	169
	DIV	带符号的除法运算	67	170
	DIVU	无符号的除法运算	68	171
	DIVX	带符号的除法运算	69	172
	DSBB	带借位的 10 进制减法运算	70	173
	DSUB	无借位的 10 进制减法运算	71	175
	EXTS	符号扩展	74	178
	INC	递增	77	180
	MUL	带符号的乘法运算	94	203
	MULU	无符号的乘法运算	95	205
	NEG	2 的补码	96	207
	RMPA	乘加运算	109	218
	SBB	带借位的减法运算	114	222
	SUB	无借位的减法运算	127	236
逻辑	AND	逻辑与	45	147
	NOT	全位反转	98	208
	OR	逻辑或	99	209
	TST	测试	129	239
	XOR	逻辑异或	133	243
转移	ADJNZ	加法运算和条件转移	44	146
	SBJNZ	减法运算和条件转移	115	224
	J*Cnd*	条件转移	80	182
	JMP	无条件转移	81	183
	JMPI	间接转移	82	185
	JSR	子程序调用	83	187
	JSRI	间接子程序调用	84	188
	RTS	从子程序的返回	113	221
字符串	SMOVB	反向字符串传送	118	230
	SMOVF	正向字符串传送	119	231
	SSTR	字符串保存	120	231
其它	BRK	调试中断	56	157
	ENTER	堆栈帧的建立	72	177
	EXITD	堆栈帧的释放	73	178
	FCLR	标志寄存器的位清除	75	179
	FSET	标志寄存器的置位	76	180
	INT	软件中断	78	181
	INTO	溢出中断	79	182
	LDC	向专用寄存器的传送	85	189

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	86	190
	LDINTB	向 INTB 寄存器的传送	88	192
	LDIPL	中断允许级的设定	89	193
	NOP	空操作	97	207
	POPC	专用寄存器的恢复	102	213
	PUSHC	专用寄存器的保存	106	216
	REIT	从中断的返回	108	217
	STC	从专用寄存器的传送	121	232
	STCTX	环境保存	122	233
	UND	未定义指令中断	130	241
	WAIT	等待	131	241

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	138
ADC	○	○	○	○	○	○	○	○	○	○	○	○	○			40	138
ADCF	○	○	○	○	○	○	○	○	○	○	○					41	140
ADD*1	○	○	○	○	○	○	○	○	○	○	○	○	○			42	140
ADJNZ*1	○	○	○	○	○	○	○	○	○	○	○				○	44	146
AND	○	○	○	○	○	○	○	○	○	○	○	○	○			45	147
CMP	○	○	○	○	○	○	○	○	○	○	○	○	○			62	161
DADC	○	○										○	○			64	165
DADD	○	○										○	○			65	167
DEC	○	○			○			○			○					66	169
DIV	○	○	○	○	○	○	○	○	○	○	○	○	○			67	170
DIVU	○	○	○	○	○	○	○	○	○	○	○	○	○			68	171
DIVX	○	○	○	○	○	○	○	○	○	○	○	○	○			69	172
DSBB	○	○										○	○			70	173
DSUB	○	○										○	○			71	175
ENTER												○				72	177
EXTS	○		○*2			○	○	○	○	○	○					74	178
INC	○*3	○*4			○			○			○					77	180
INT															○	78	181
JMPI*1	○	○	○	○	○	○	○	○		○	○					82	185
JSRI*1	○	○	○	○	○	○	○	○		○	○					84	188
LDC*1	○	○	○	○	○	○	○	○	○	○	○		○			85	189
LDE*1	○	○	○	○	○	○	○	○	○	○	○					87	191
LDINTB														○		88	192
LDIPL															○	89	193

*1 有特殊的指令寻址。

*2 只能选择R1L。

*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	#IMM8	#IMM16	#IMM20	#IMM		
MOV[*1]	○	○	○	○	○	○	○	○	○	○	○	○	○			90	193
MOVA	○	○	○	○	○		○	○	○	○	○					92	200
MOV*Dir*	○	○	○	○		○	○	○	○	○	○					93	201
MUL	○	○	○	○	○	○	○	○	○	○	○	○	○			94	203
MULU	○	○	○	○	○	○	○	○	○	○	○	○	○			95	205
NEG	○	○	○	○	○	○	○	○	○	○	○					96	207
NOT	○	○	○	○	○	○	○	○	○	○	○					98	208
OR	○	○	○	○	○	○	○	○	○	○	○	○	○			99	209
POP	○	○	○	○	○	○	○	○	○	○	○					101	211
POPM[*1]	○	○	○	○	○											103	213
PUSH	○	○	○	○	○	○	○	○	○	○	○					104	214
PUSHA							○	○	○	○	○					105	216
PUSHM[*1]	○	○	○	○	○											107	217
ROLC	○	○	○	○	○	○	○	○	○	○	○					110	218
RORC	○	○	○	○	○	○	○	○	○	○	○					111	219
ROT	○	○	○	○	○	○	○	○	○	○	○				○	112	220
SBB	○	○	○	○	○	○	○	○	○	○	○	○	○			114	222
SBJNZ[*1]	○	○	○	○	○	○	○	○	○	○	○				○	115	224
SHA[*1]	○	○	○	○	○	○	○	○	○	○	○				○	116	225
SHL[*1]	○	○	○	○	○	○	○	○	○	○	○				○	117	228
STC[*1]	○	○	○	○	○	○	○	○	○	○	○					121	232
STCTX[*1]											○					122	233
STE[*1]	○	○	○	○	○	○	○	○	○	○	○					123	233
STNZ	○	○						○			○	○				124	235
STZ	○	○						○			○	○				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			
STZX	○	○						○			○	○				126	236
SUB	○	○	○	○	○	○	○	○	○	○	○	○	○			127	236
TST	○	○	○	○	○	○	○	○	○	○	○	○	○			129	239
XCHG	○	○	○	○	○	○	○	○	○	○	○					132	242
XOR	○	○	○	○	○	○	○	○	○	○	○	○	○			133	243

*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	140
ADJNZ*1								○						44	146
J*Cnd*								○						80	182
JMP			○					○						81	183
JMPI*1	○	○		○	○									82	185
JSR			○					○						83	187
JSRI*1	○	○		○	○									84	188
LDC*1									○	○	○	○		85	189
LDCTX			○											86	190
LDE*1	○		○			○								87	191
LDINTB												○*2		88	192
MOV*1							○							90	193
POPC									○	○	○	○		102	213
POPM*1									○					103	213
PUSHC									○	○	○	○		106	216
PUSHM*1									○					107	217
SBJNZ*1								○						115	224
SHA*1				○										116	225
SHL*1				○										117	228
STC*1				○	○				○	○	○	○	○	121	232
STCTX*1			○											122	233
STE*1	○		○			○								123	233

*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/I/O/B/S/Z/D/C		
BAND	○	○	○	○	○	○	○	○			47	150
BCLR	○	○	○	○	○	○	○	○	○		48	150
BM*Cnd*	○	○	○	○	○	○	○	○		○	49	152
BNAND	○	○	○	○	○	○	○	○			50	153
BNOR	○	○	○	○	○	○	○	○			51	154
BNOT	○	○	○	○	○	○	○	○	○		52	154
BNTST	○	○	○	○	○	○	○	○			53	155
BNXOR	○	○	○	○	○	○	○	○			54	156
BOR	○	○	○	○	○	○	○	○			55	156
BSET	○	○	○	○	○	○	○	○	○		57	157
BTST	○	○	○	○	○	○	○	○	○		58	158
BTSTC	○	○	○	○	○	○	○	○			59	159
BTSTS	○	○	○	○	○	○	○	○			60	160
BXOR	○	○	○	○	○	○	○	○			61	160
FCLR										○	75	179
FSET										○	76	180

第 1 章

概要

1.1　R8C/Tiny 系列的特点

R8C/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89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　R8C/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

●高速的指令执行时间

最短1个周期指令：在89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	0.1μs
寄存器和存储器之间的传送	0.1μs
寄存器之间的加减运算	0.1μs
8位×8位寄存器之间的运算	0.2μs
16位×16位寄存器之间的运算	0.250μs
16位÷8位寄存器之间的运算	0.904μs
32位÷16位寄存器之间的运算	1.248μs

［条件］

- 具有内部乘法器产品
- 时钟频率　20MHz

1.2　地址空间

地址空間如图1.2.1所示。

00000$_{16}$地址～002FF$_{16}$地址为SFR（专用功能寄存器）区。根据产品种类，从002FF$_{16}$地址开始向低地址方向扩展SFR区。

00400$_{16}$地址以后的区域为存储器区。根据产品种类，从00400$_{16}$地址开始向高地址方向扩展RAM区，从0FFFF$_{16}$地址开始向低地址方向扩展ROM区。但是，0FFDC$_{16}$地址～0FFFF$_{16}$地址为固定向量区。

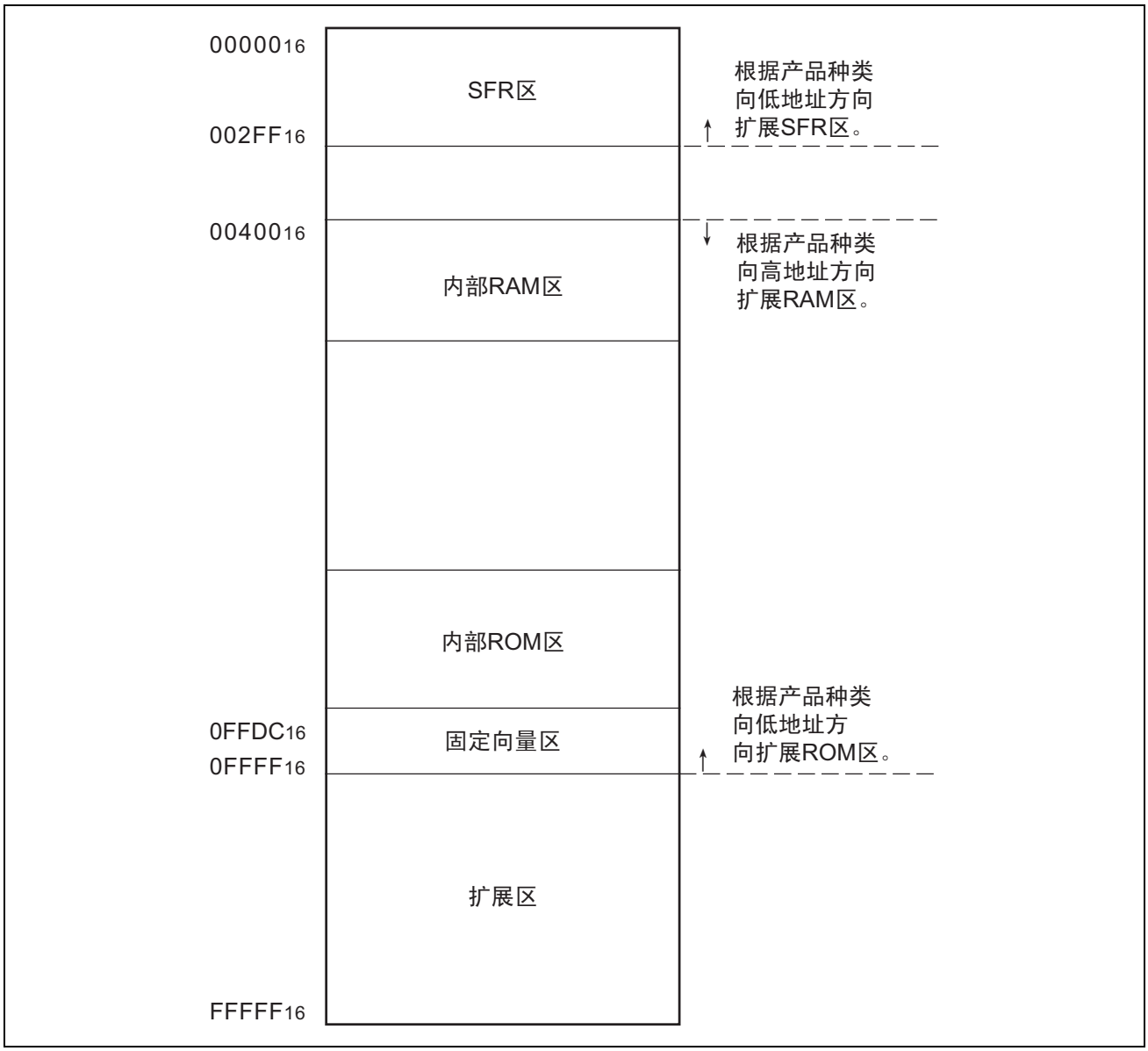


图1.2.1 地址空间

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。

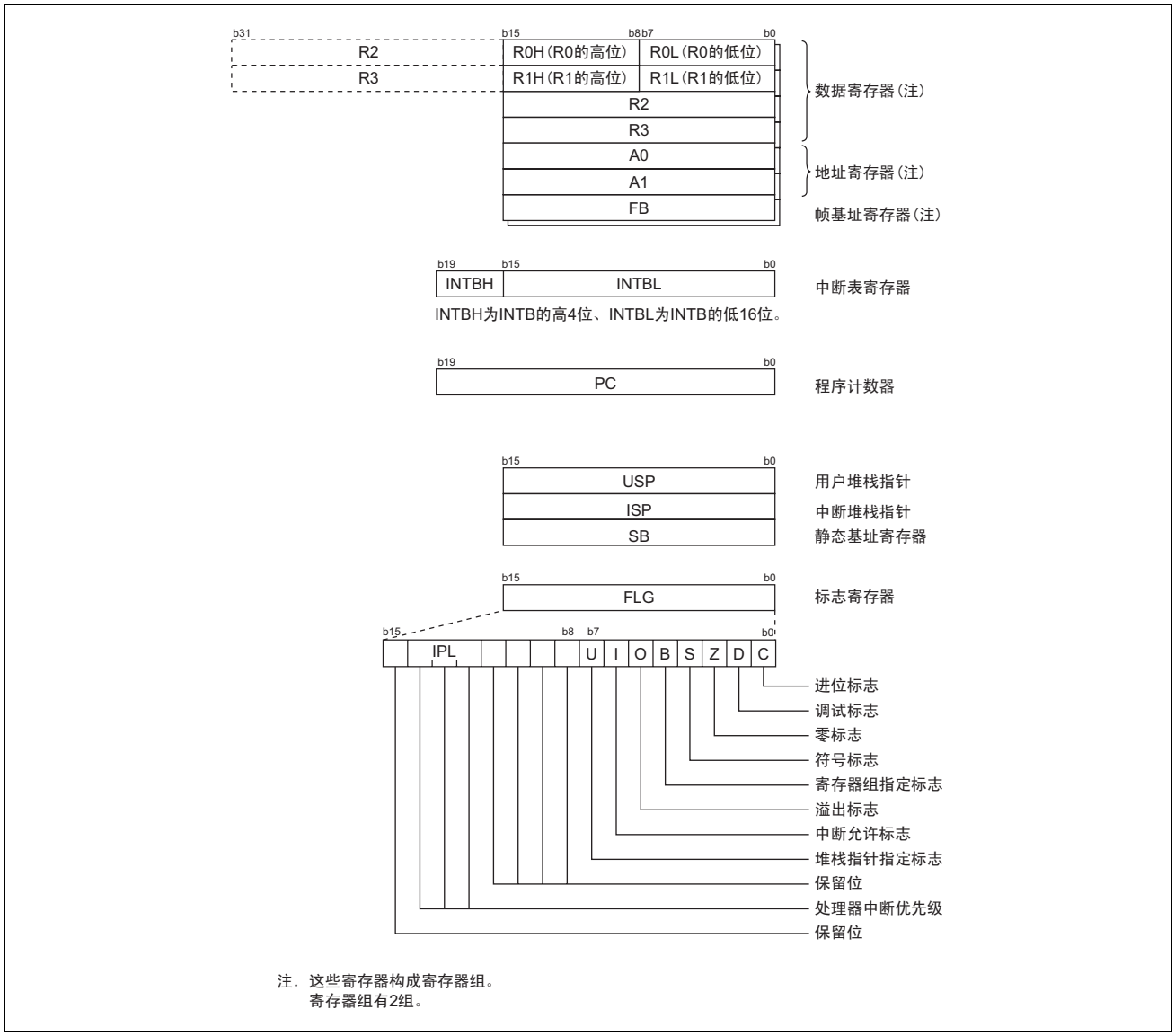


图1.3.1 中央处理器的寄存器构成

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）

它是允许可屏蔽中断的标志。

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

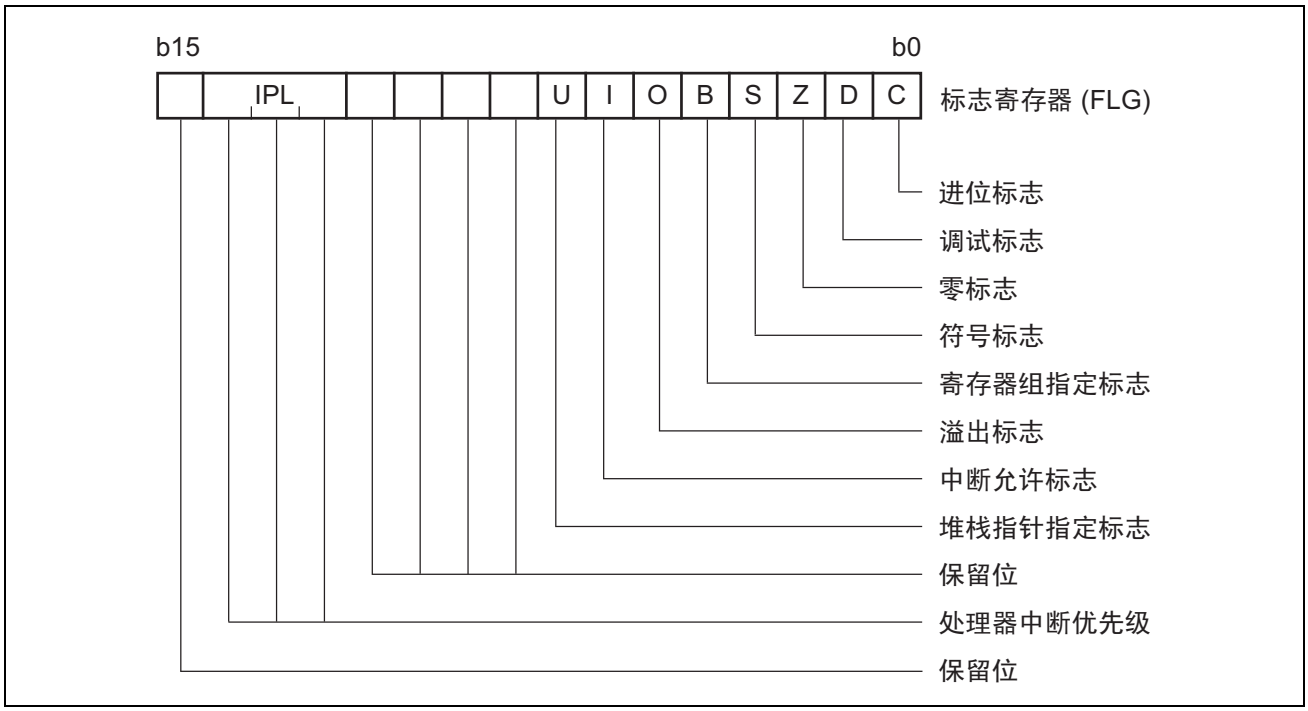


图1.4.1 标志寄存器（FLG）的构成

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。

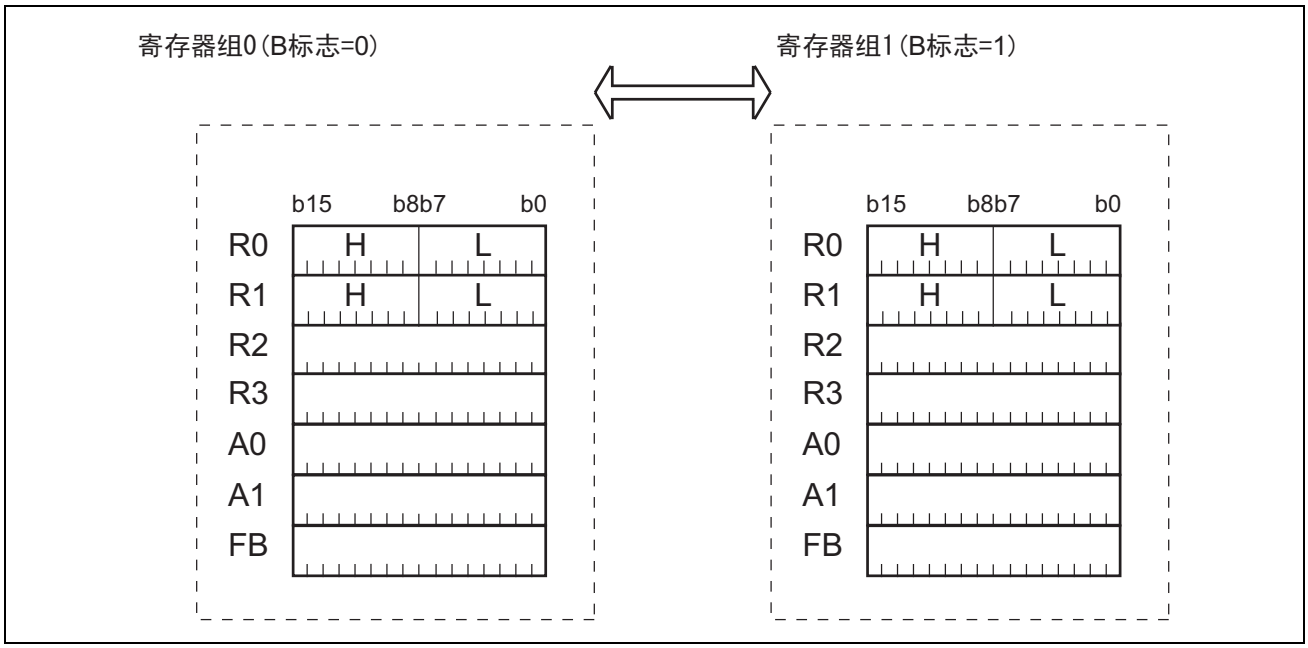


图1.5.1 寄存器组的构成

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）　：0000_{16}
- 地址寄存器（A0/A1）　：0000_{16}
- 帧基址寄存器（FB）　：0000_{16}
- 中断表寄存器（INTB）　：00000_{16}
- 用户堆栈指针（USP）　：0000_{16}
- 中断堆栈指针（ISP）　：0000_{16}
- 静态基址寄存器（SB）　：0000_{16}
- 标志寄存器（FLG）　：0000_{16}

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。

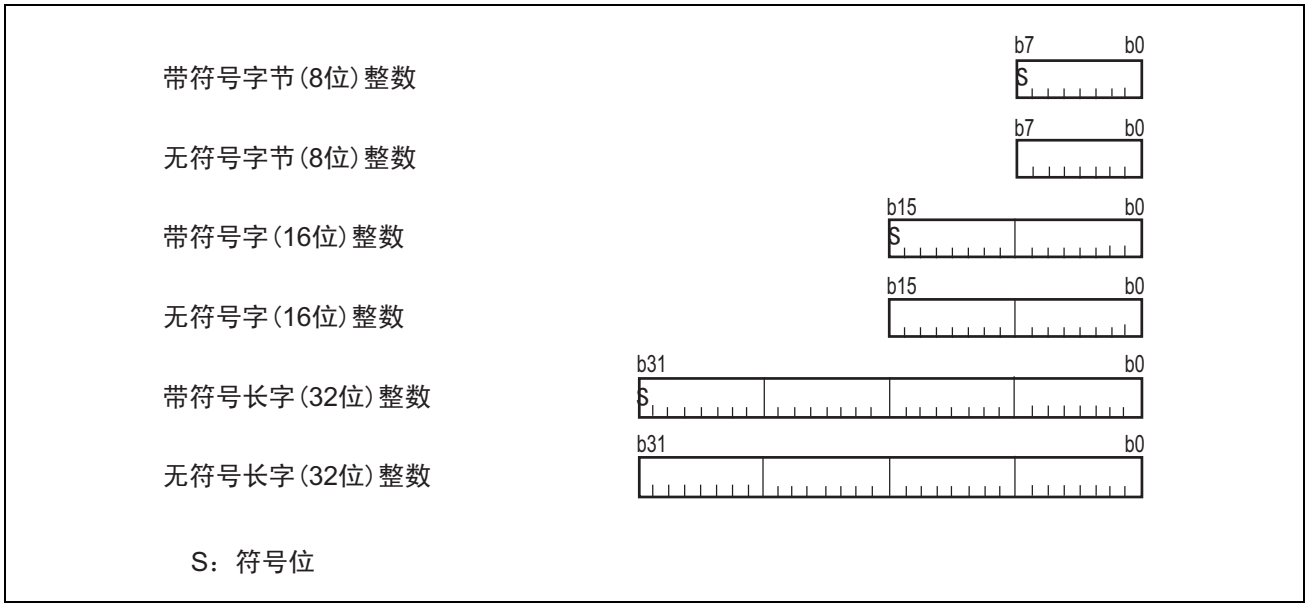


图1.7.1 整数数据

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。

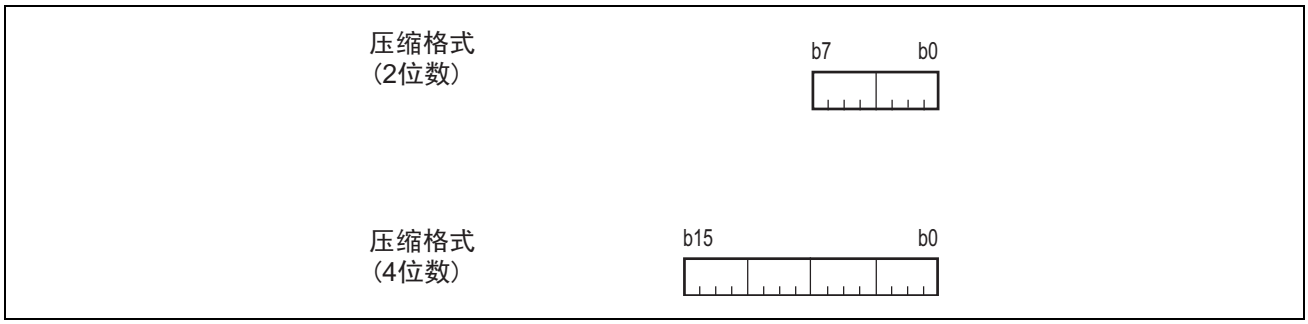


图1.7.2 10进制数据

1.7.3　位

●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。

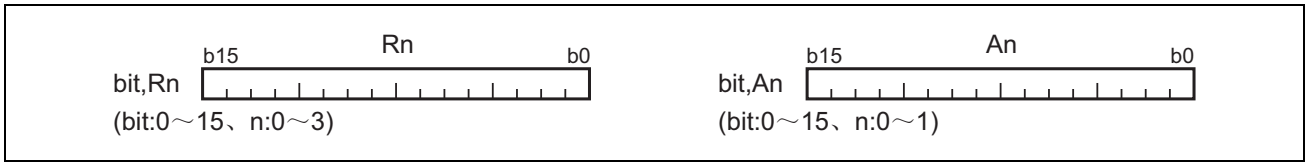


图1.7.3 寄存器的位指定

●存储器的位

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。

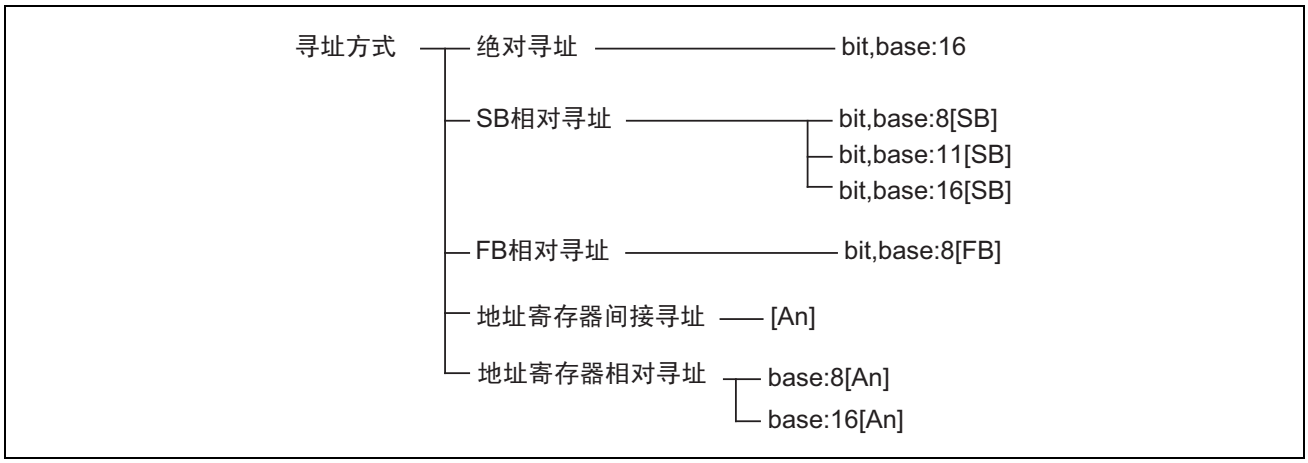


图1.7.4 在指定存储器的位时使用的寻址方式

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 考
	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}	01FFF16	
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。

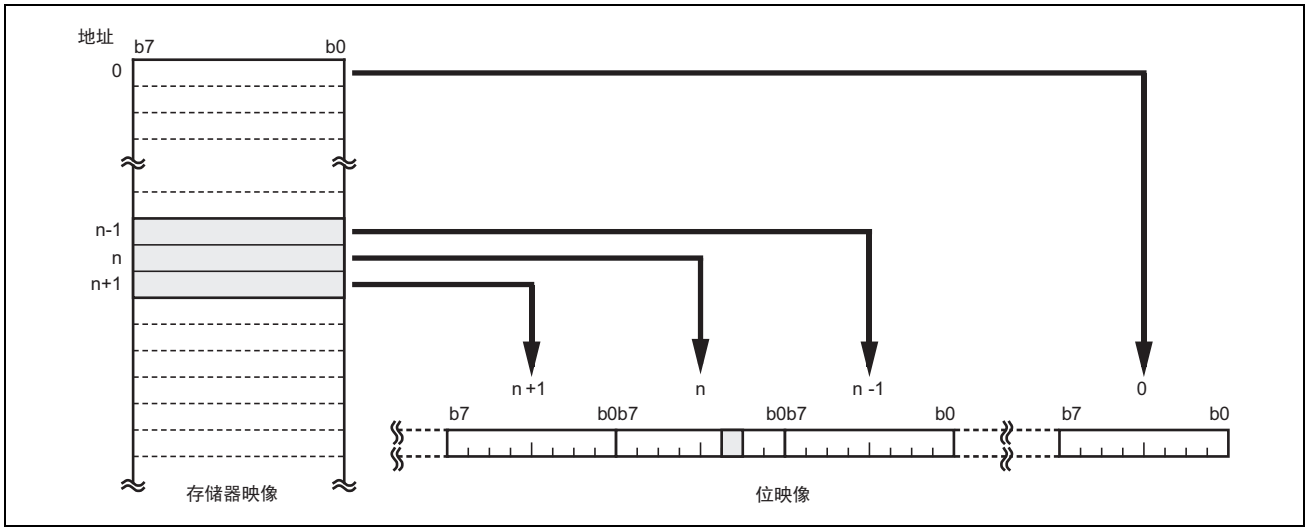


图1.7.5 存储器映像和位映像的关系

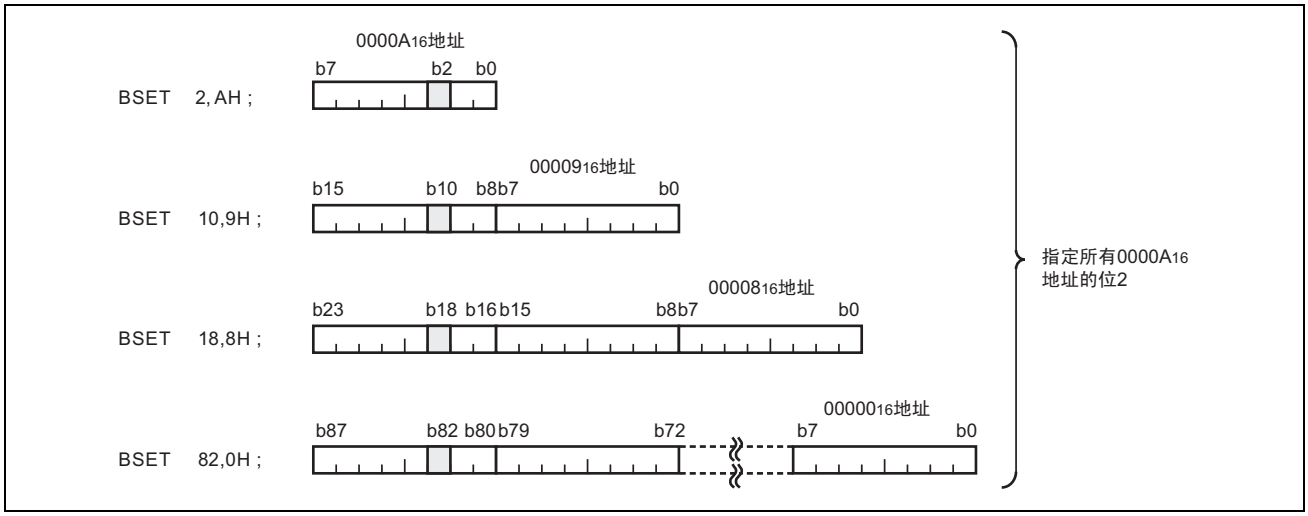


图1.7.6 0000A16地址的位2的指定例子

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00000_{16}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文字列反向传送（SMOVB指令）、文字列正向传送（SMOVF指令）以及指定区域的初始化（SSTR指令）的3种指令。

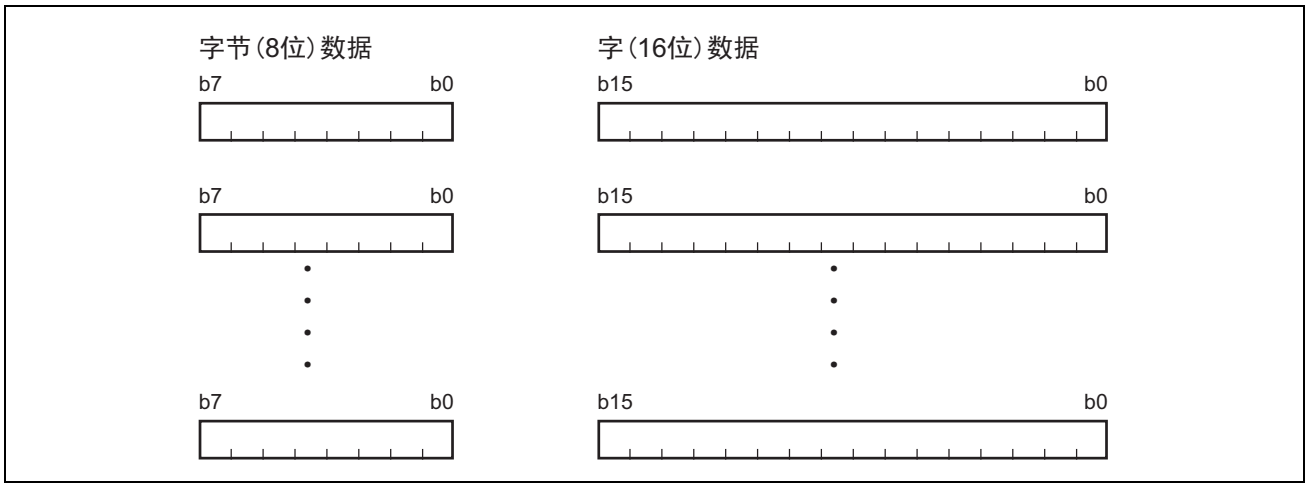


图1.7.7 字符串数据

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。

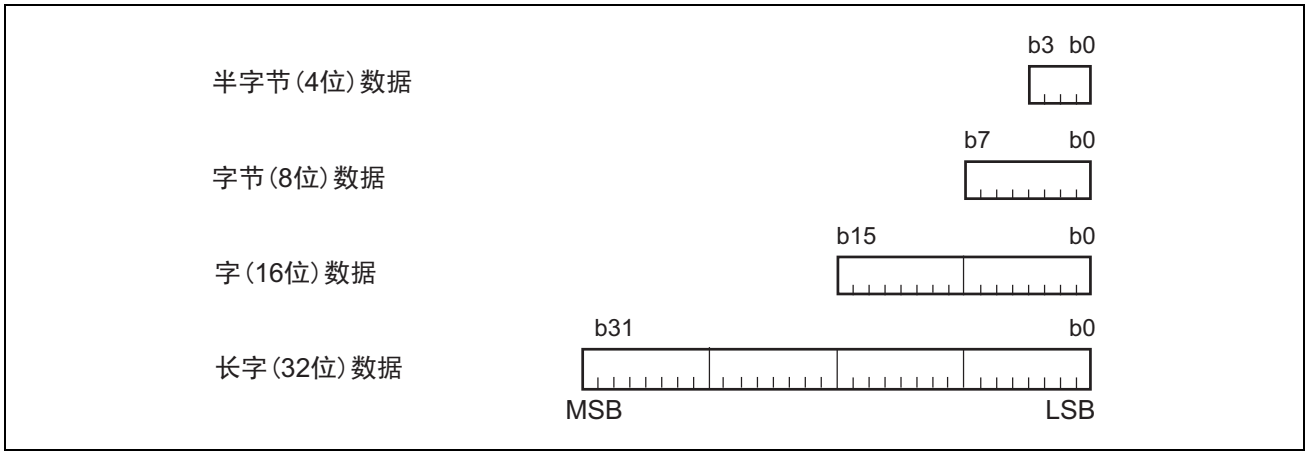


图1.8.1 寄存器内的数据分配

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。

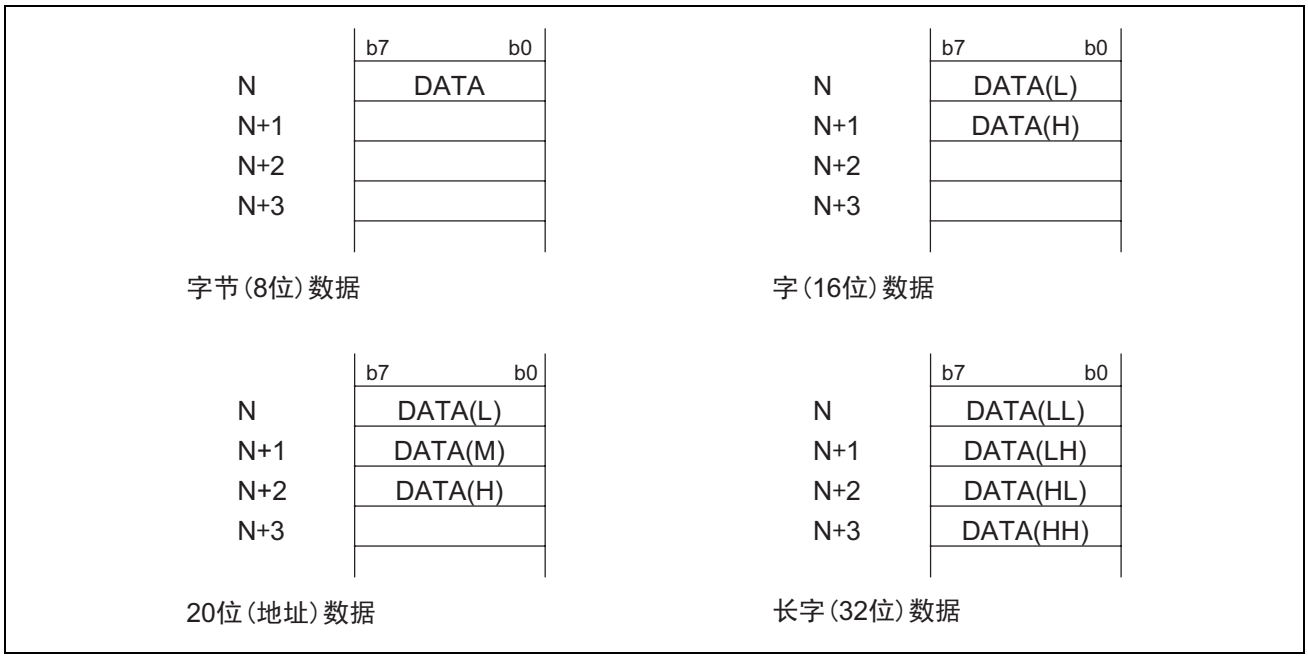


图1.8.2 存储器内的数据分配

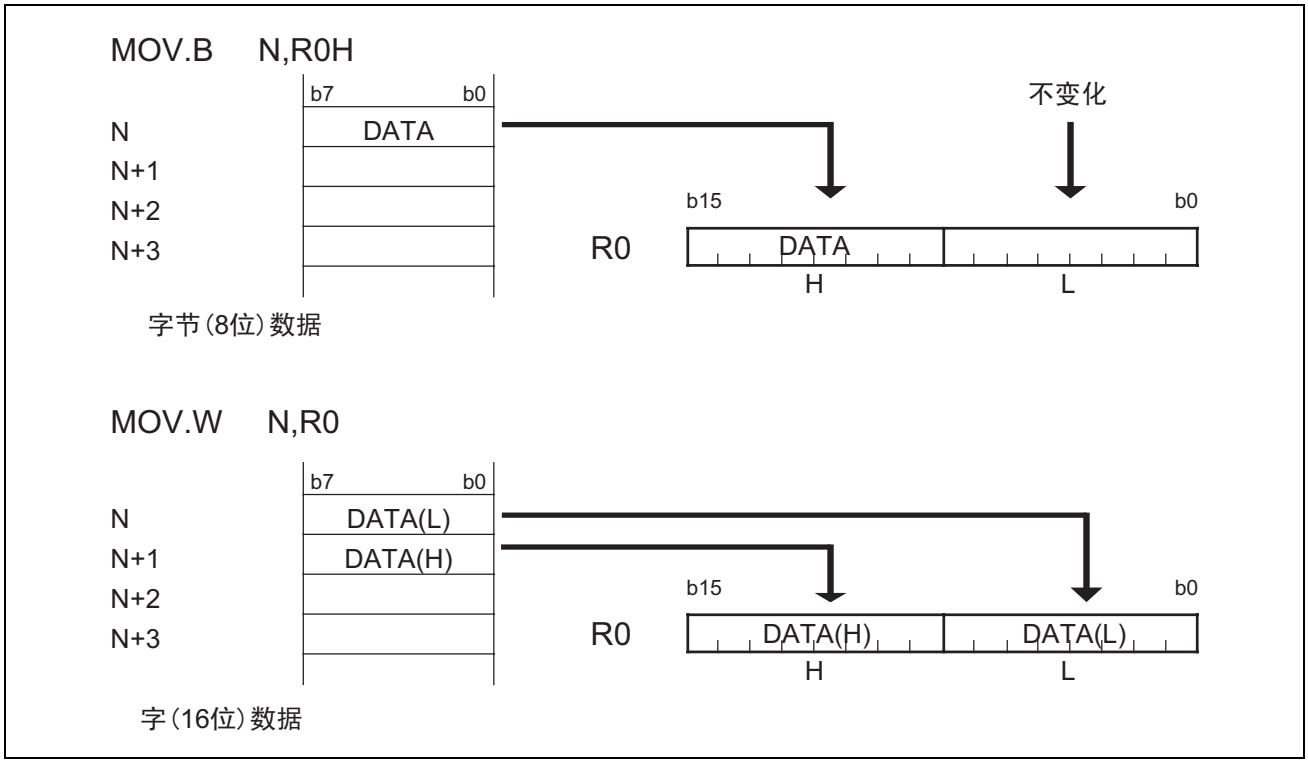


图1.8.3 运算例子

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、快速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的略称

*2 destination的略称

1.10　向量表

作为向量表，有中断向量表。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

固定向量表为固定地址的向量表。在从0FFDC16地址到0FFFF16地址中分配了中断向量表的一部分。固定向量表如图1.10.1所示。

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。

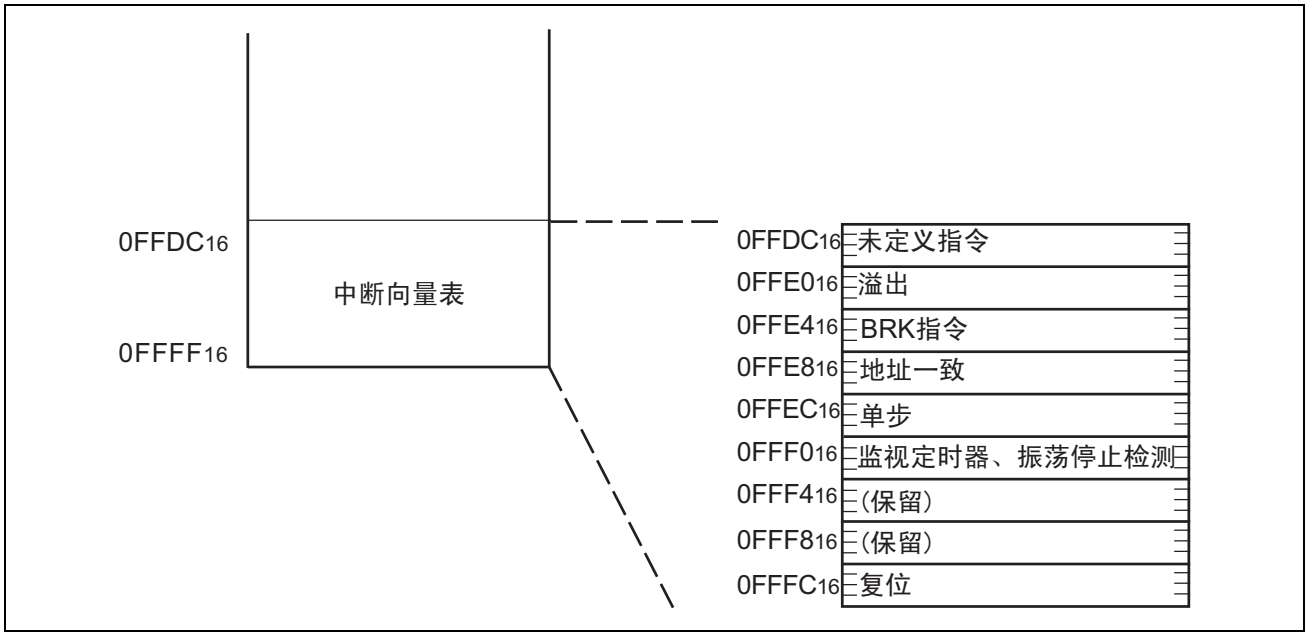


图1.10.1 固定向量表

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，内藏的外围功能中断被分配在软件中断号0～31中。

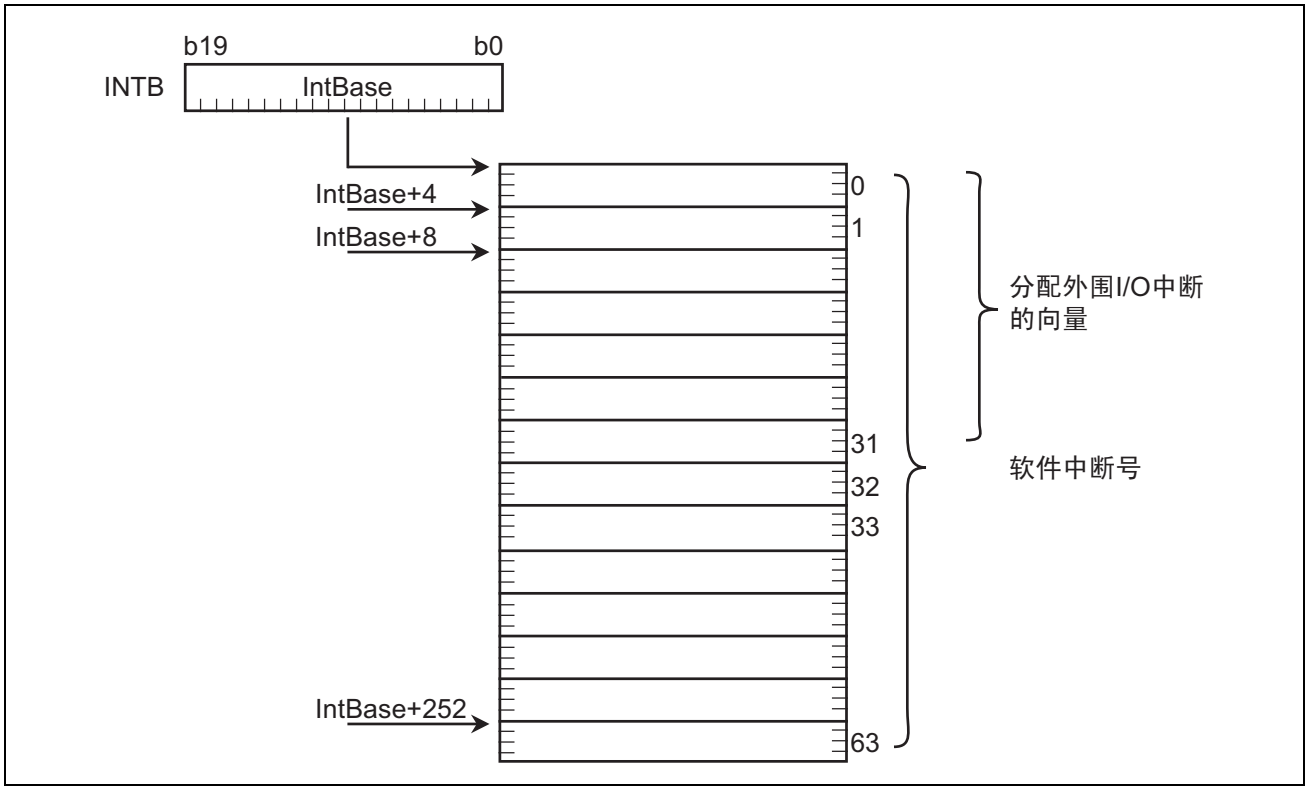


图1.10.2 可变向量表

第 2 章

寻址方式

2.1　寻址方式

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00000$_{16}$地址到0FFFF$_{16}$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00000$_{16}$地址到FFFFF$_{16}$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

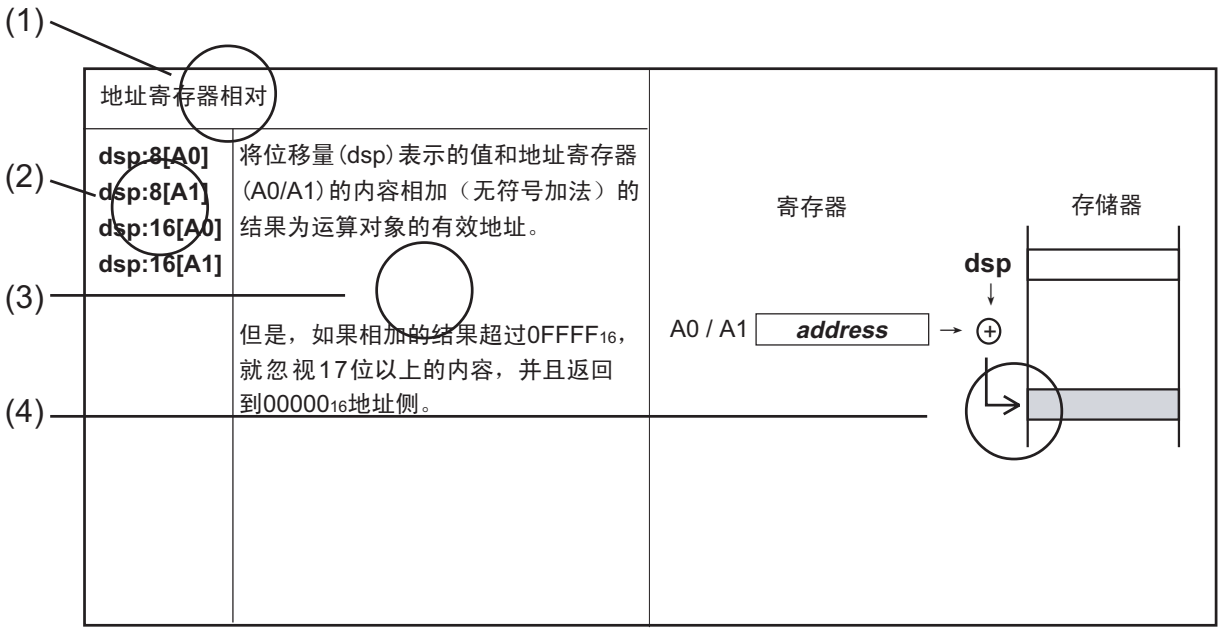
它是以位单位存取从00000$_{16}$地址到0FFFF$_{16}$地址的区域的寻址。

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。

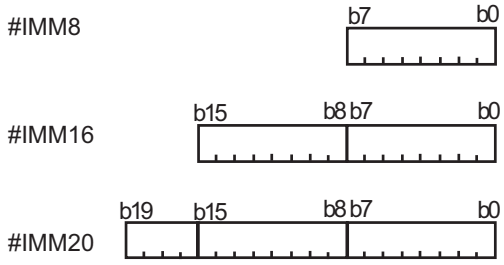
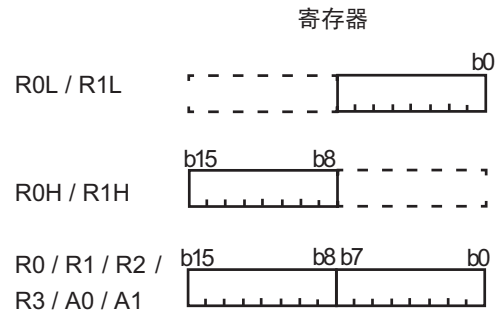
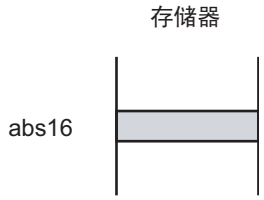
(3) 解说

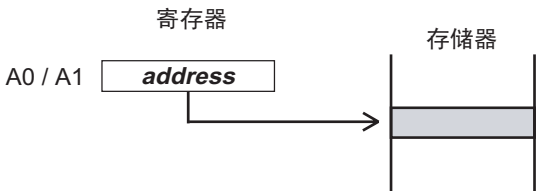
说明地址的操作和有效地址的范围。

(4) 操作图

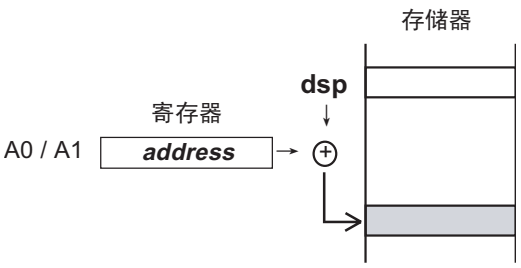
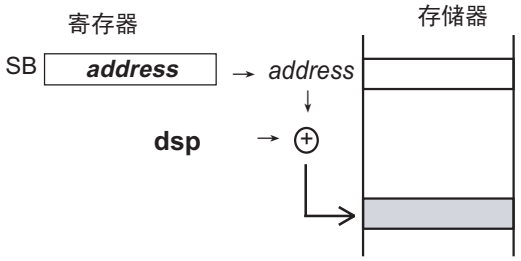
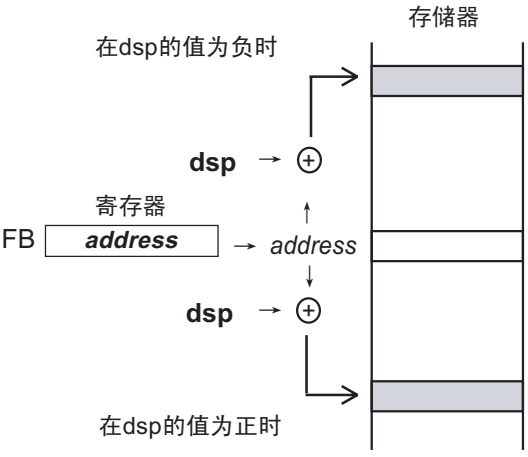
用图说明地址的操作。

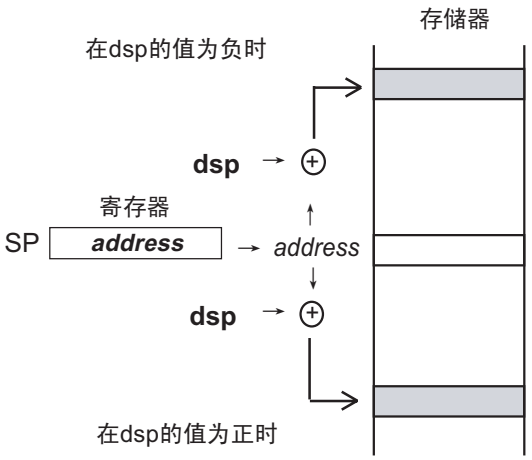
2.3　一般指令寻址

<table>
<tr><td colspan="2">立即</td><td rowspan="2"> 
</td></tr>
<tr><td>#IMM
#IMM8
#IMM16
#IMM20</td><td>以#IMM表示的立即数为运算对象。</td></tr>
<tr><td colspan="2">寄存器直接</td><td rowspan="2"> 
</td></tr>
<tr><td>R0L
R0H
R1L
R1H
R0
R1
R2
R3
A0
A1</td><td>以指定的寄存器为运算对象。</td></tr>
<tr><td colspan="2">绝对</td><td rowspan="2"> 
</td></tr>
<tr><td>abs16</td><td>以abs16表示的值为运算对象的有效地址。

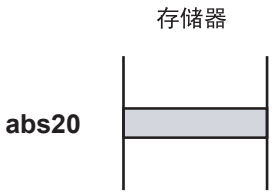
有效地址的范围为00000₁₆～0FFFF₁₆。</td></tr>
<tr><td colspan="2">地址寄存器间接</td><td rowspan="2"> 
</td></tr>
<tr><td>[A0]
[A1]</td><td>以地址寄存器（A0/A1）的内容表示的值为运算对象的有效地址。

有效地址的范围为00000₁₆～0FFFF₁₆。</td></tr>
</table>

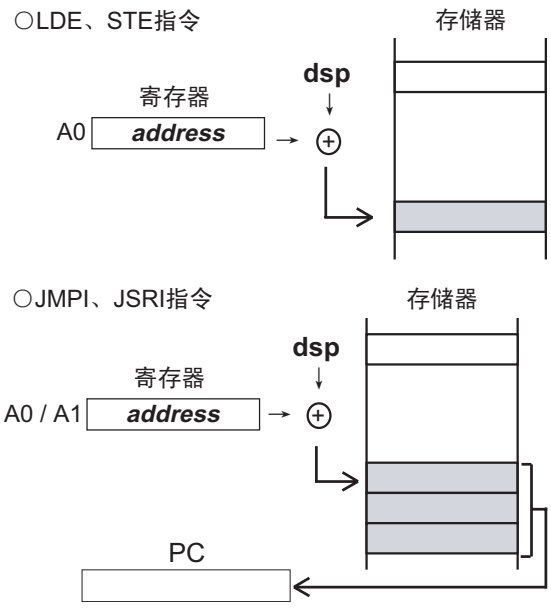
地址寄存器相对		
dsp:8[A0] dsp:8[A1] dsp:16[A0] dsp:16[A1]	将位移量（dsp）表示的值和地址寄存器（A0/A1）的内容相加（无符号加法）的结果为运算对象的有效地址。 但是，如果相加的结果超过0FFFF16，就忽视17位以上的内容，并且返回到0000016地址侧。	
SB相对		
dsp:8[SB] dsp:16[SB]	将静态基址寄存器（SB）的内容表示的地址和由位移量（dsp）表示的值相加（无符号加法）的结果为运算对象的有效地址。 但是，如果相加的结果超过0FFFF16，就忽视17位以上的内容，并且返回到0000016地址侧。	
FB相对		
dsp:8[FB]	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00000$_{16}$～0FFFF$_{16}$，就忽视17位以上的内容，并且返回到00000$_{16}$地址侧或者0FFFF$_{16}$地址侧。 此寻址能用于MOV指令。	

2.4　特殊指令寻址

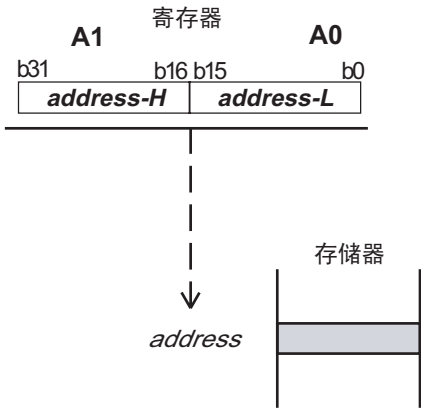
<table>
<tr><td colspan="2">20位绝对</td><td rowspan="2"> 
</td></tr>
<tr><td>abs20</td><td>以abs20表示的值为运算对象的有效地址。

有效地址的范围为00000₁₆～FFFFF₁₆。

此寻址能用于LDE、STE、JSR、JMP指令。</td></tr>
<tr><td colspan="2">20位带位移量的地址寄存器相对</td><td rowspan="2"> 
</td></tr>
<tr><td>dsp:20[A0]
dsp:20[A1]</td><td>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。

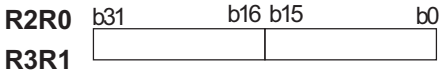
但是，如果相加的结果超过FFFFF₁₆，就忽视21位以上的内容，并且返回到00000₁₆地址侧。

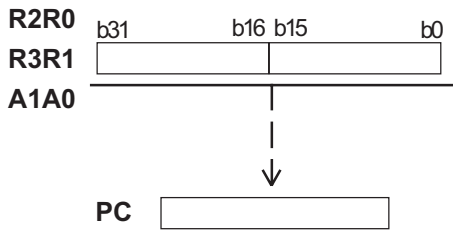
此寻址能用于LDE、STE、JMPI、JSRI指令。

能使用的寻址方式和指令的组合如下所示：
dsp:20[A0]
　→LDE、STE、JMPI、JSRI指令
dsp:20[A1]
　→JMPI、JSRI指令</td></tr>
<tr><td colspan="2">32位地址寄存器间接</td><td rowspan="2"> 
</td></tr>
<tr><td>[A1A0]</td><td>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。

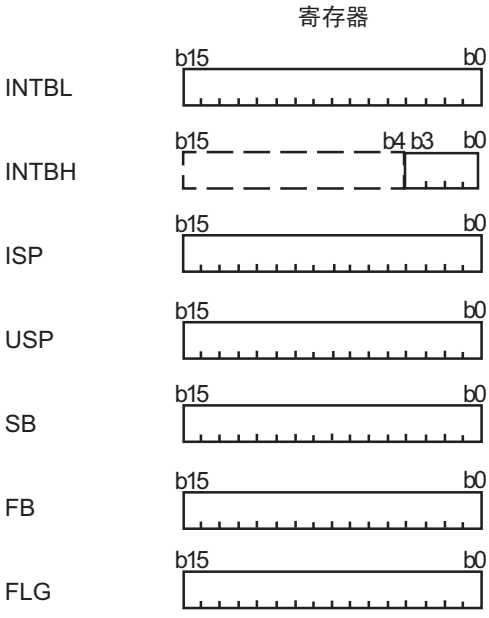
但是，如果相连后的寄存器值超过FFFFF₁₆，就忽视21位以上的内容。

此寻址能用于LDE、STE指令。</td></tr>
</table>

<table>
<tr><td colspan="2">32位寄存器直接</td><td rowspan="2">
○SHL、SHA指令
 

○JMPI、JSRI指令
 
</td></tr>
<tr><td>R2R0
R3R1
A1A0</td><td>将指定的2个寄存器相连组成的32位寄存器为运算对象。

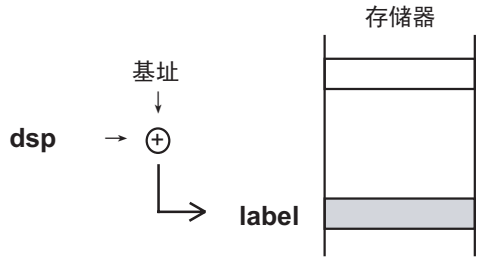
此寻址能用于SHL、SHA、JMPI、JSRI指令。

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</td></tr>
<tr><td colspan="2">专用寄存器直接</td><td rowspan="2"> 
</td></tr>
<tr><td>INTBL
INTBH
ISP
SP
SB
FB
FLG</td><td>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。

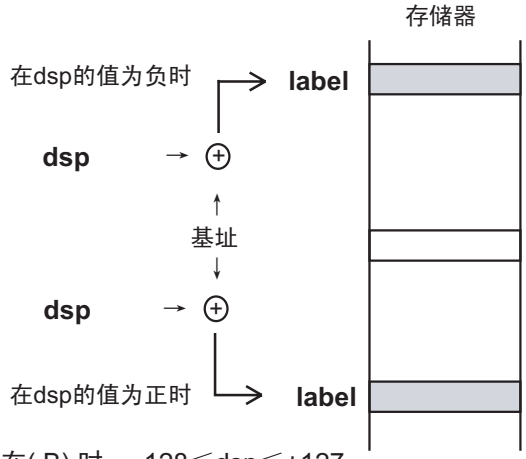
如果指定SP，就以U标志指示的堆栈指针为对象。</td></tr>
</table>

<table>
<tr><td colspan="2">程序计数器相对</td><td rowspan="2"></td></tr>
<tr><td>label</td><td>○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。</td><td> 

+0≤dsp≤+7
*1 基址为(指令的起始地址+2)。</td></tr>
<tr><td></td><td>○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。

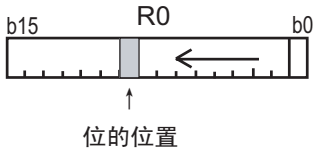
此寻址能用于JMP、JSR指令。</td><td> 

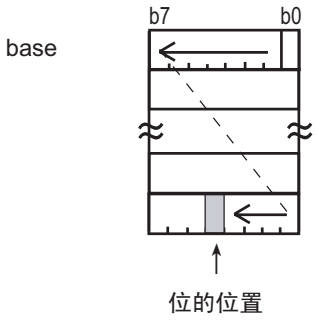
在(.B) 时，−128≤dsp≤+127
在(.W) 时，−32768≤dsp≤+32767
*2 基址根据指令的不同而不同。</td></tr>
</table>

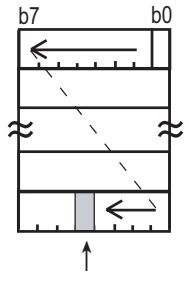
2.5　位指令寻址

此寻址能用于以下指令：

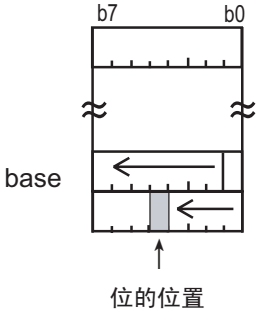
BCLR、BSET、BNOT、BTST、BNTST、BAND、BNAND、BOR、BNOR、BXOR、BNXOR、BM*Cnd*、BTSTS、BTSTC

<table>
<tr><td colspan="2">寄存器直接</td><td rowspan="2">bit , R0
 
</td></tr>
<tr><td>bit,R0
bit,R1
bit,R2
bit,R3
bit,A0
bit,A1</td><td>以指定的寄存器的位为运算对象。

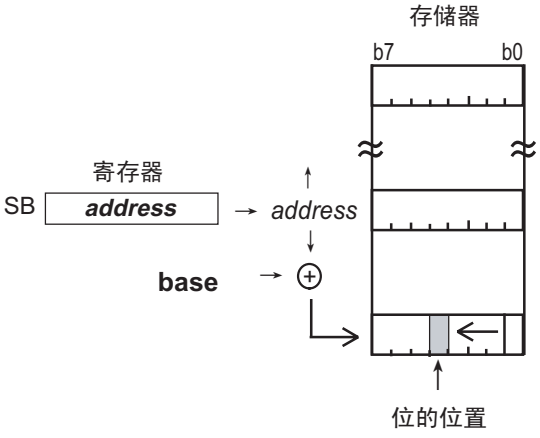
位的位置（bit）能指定0～15。</td></tr>
<tr><td colspan="2">绝对</td><td rowspan="2">base
 
</td></tr>
<tr><td>bit,base:16</td><td>以base表示的地址的位0为起点间隔bit表示的位数的位为运算对象。

以00000₁₆地址～01FFF₁₆地址的位为对象。</td></tr>
<tr><td colspan="2">地址寄存器间接</td><td rowspan="2">00000₁₆
 
</td></tr>
<tr><td>[A0]
[A1]</td><td>以00000₁₆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00000₁₆地址～01FFF₁₆地址的位为对象。</td></tr>
</table>

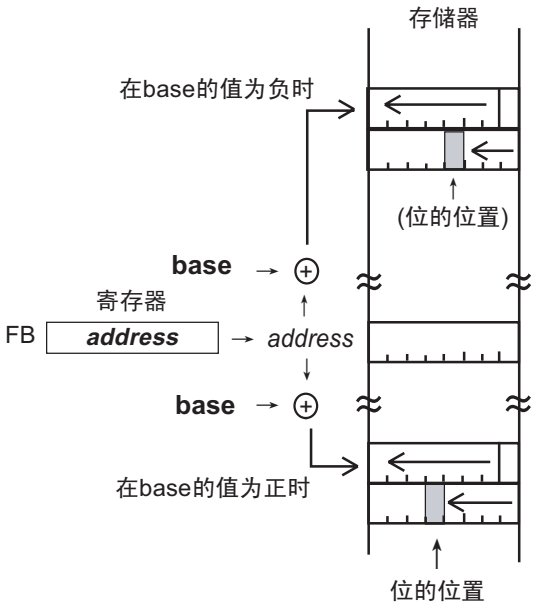
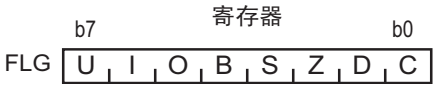
<table>
<tr><td colspan="2">地址寄存器相对</td><td rowspan="2"> 
</td></tr>
<tr><td>base:8[A0]
base:8[A1]
base:16[A0]
base:16[A1]</td><td>以base表示的地址的位0为起点间隔地址寄存器（A0/A1）表示的位数的位为运算对象。

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。</td></tr>
<tr><td colspan="2">SB相对</td><td rowspan="2"> 
</td></tr>
<tr><td>bit,base:8[SB]
bit,base:11[SB]
bit,base:16[SB]</td><td>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。</td></tr>
</table>

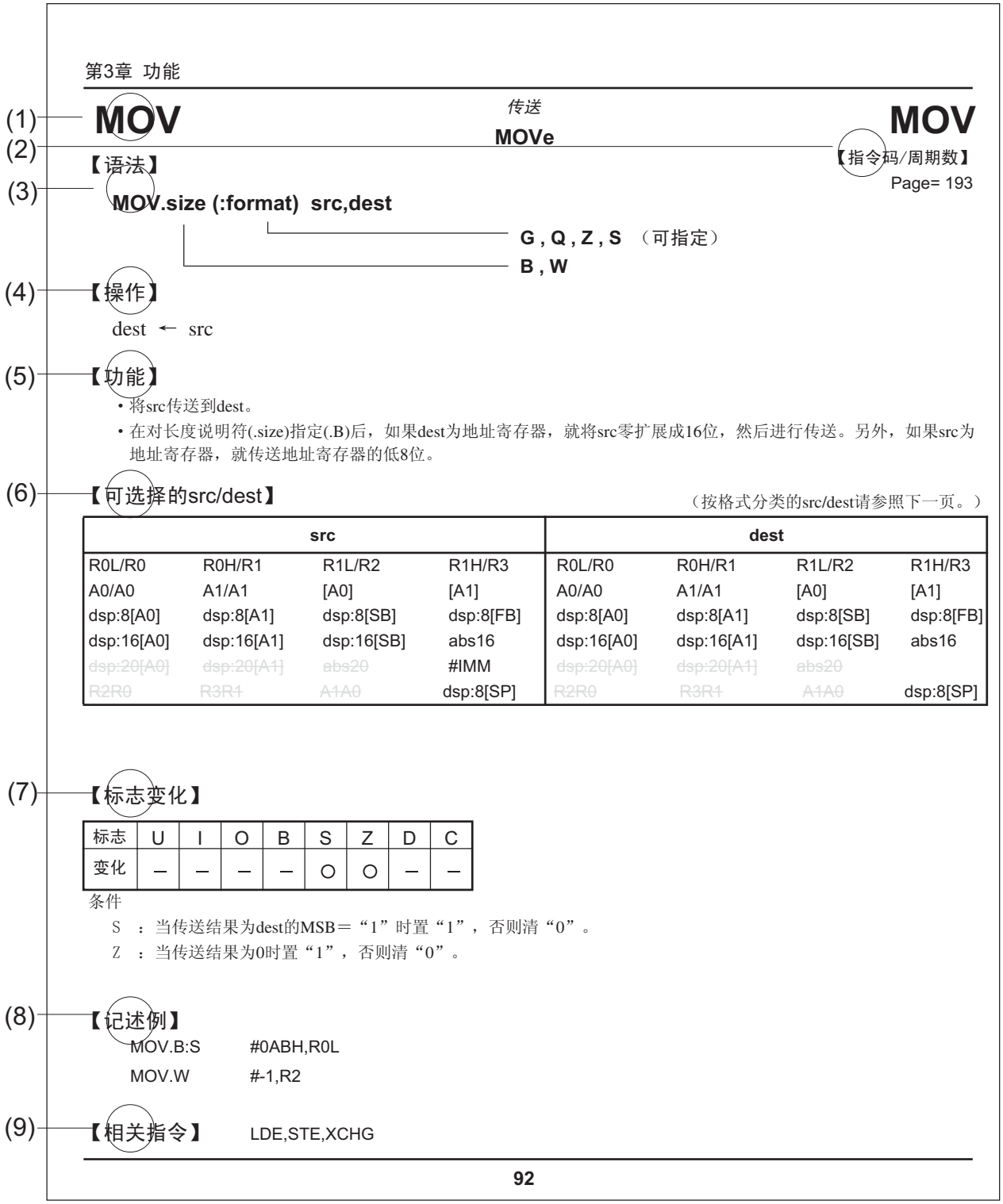
FB相对		
bit,base:8[FB]	将帧基址寄存器（FB）的内容表示的值和由base表示的值相加（带符号加法），以相加后的地址的位0为起点间隔bit表示的位数的位为运算对象。 但是，如果对象位的地址超过00000_{16}～$0FFFF_{16}$，就忽视17位以上的数据，并且返回到00000_{16}地址侧或者$0FFFF_{16}$地址侧。 能用bit,base:8指定的地址范围是以帧基址寄存器（FB）的值为中心的32字节。	
FLG直接		
U I O B S Z D C	以指定的标志为运算对象。 此寻址能用于FCLR、FSET指令。	

第 3 章

功能

3.1　本章的阅读方法

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*　　**MOV**

(2) **MOVe**　　【指令码/周期数】

Page= 193

【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。

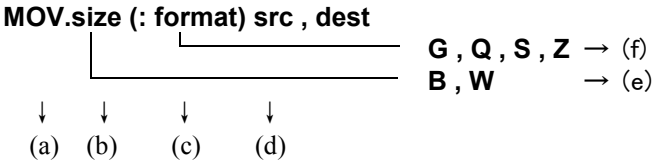
(2) 指令码/周期数

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。

(1) **MOV**　　*传送*　　**MOV**

(2) **MOVe**　　【指令码/周期数】 Page= 193

【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

(4) 操作

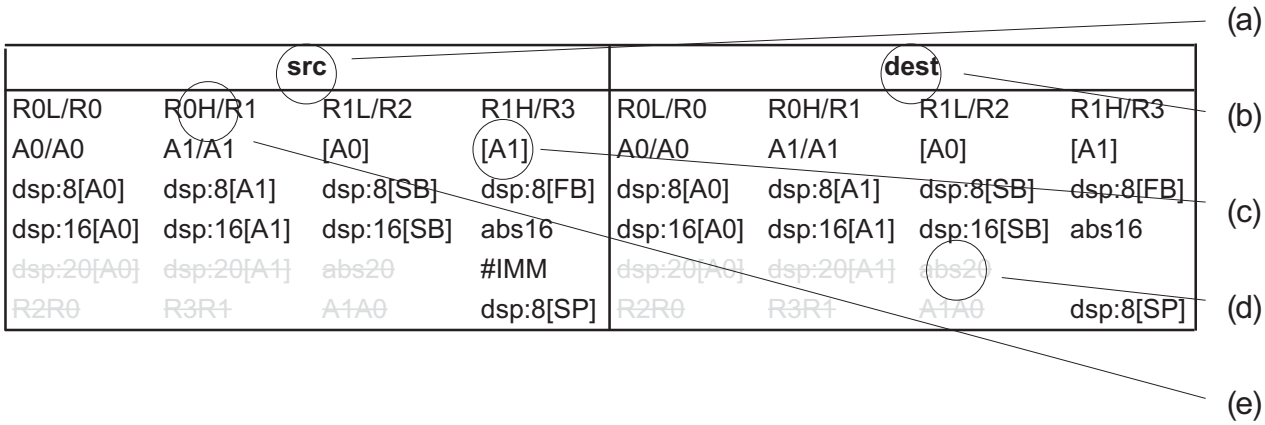
用符号说明指令的操作。

(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~~	
~~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。

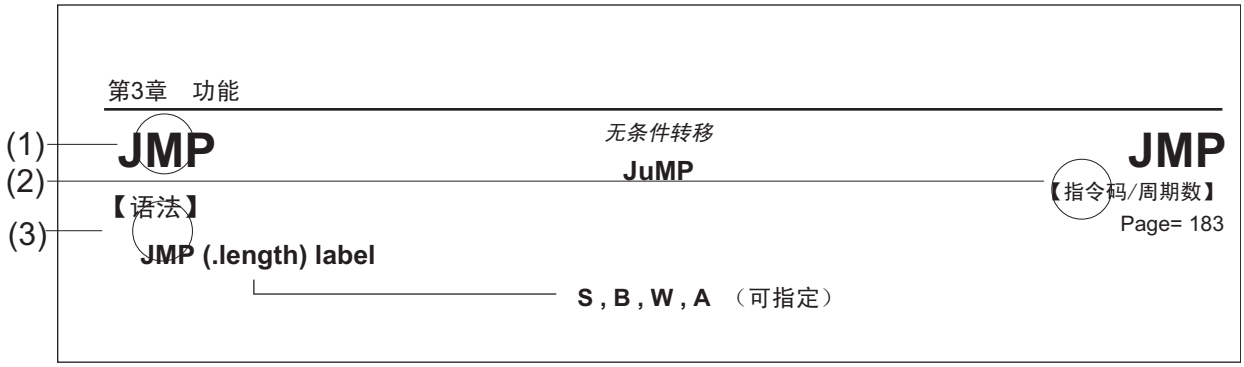
(8) 记述例

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。

JMP (.length) label

S , B , W , A →（d）

↓　↓　↓

(a)　(b)　(c)

(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

绝对值
ABSolute

ABS

【语法】
ABS.size　dest
B, W

【指令码/周期数】
Page=138

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

【记述例】
```
ABS.B    R0L
ABS.W    A0
```

ADC

带进位的加法运算
ADdition with Carry

ADC

【语法】
ADC.size　src,dest
B, W

【指令码/周期数】
Page=138

【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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当带符号运算的结果超过+32767(.W)～－32768(.W)或者+127(.B)～－128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号运算的结果超过+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进行运算。
ADC.B   R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
```

【相关指令】　ADCF,ADD,SBB,SUB

ADCF

进位标志的加法运算
ADdition Carry Flag

ADCF

【语法】
ADCF.size　dest
B, W

【指令码/周期数】
Page=140

【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 ：当带符号运算的结果超过+32767(.W)～－32768(.W)或者+127(.B)～－128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号运算的结果超过+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

无进位的加法运算
ADDition

ADD

【语法】

ADD.size (:format)　src,dest

G, Q, S（可指定）
B, W

【指令码/周期数】

Page=140

【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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。
- 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，然后进行运算。

【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 运算对象为由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 ：当带符号运算的结果超过+32767(.W)～－32768(.W)或者+127(.B)～－128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号运算的结果超过+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进行运算。
ADD.B   R0L,A0              ; 在将R0L零扩展后，和A0进行运算。
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 运算对象为由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 运算对象为由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≤#IMM≤+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

加法运算和条件转移

ADdition then Jump on Not Zero

ADJNZ

【语法】

ADJNZ.size　src,dest,label

B, W

【指令码/周期数】

Page=146

【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 R1H/R3 [A0] dsp:8[A1] dsp:16[A0] abs16	R0H/R1 A0/A0 [A1] dsp:8[SB] dsp:16[A1]	R1L/R2 A1/A1 dsp:8[A0] dsp:8[FB] dsp:16[SB]	PC*2 –126≤label≤PC*2 +129

*1 能取得的范围为－8≤#IMM≤+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=147

【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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为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进行运算。
AND.B:G   R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
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

位逻辑与
Bit AND carry flag

BAND

【语法】
BAND　src

【指令码/周期数】
Page=150

【操作】
C ← src ∧ 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 ：当运算结果为“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

BCLR

位清除
Bit CLeaR

BCLR

【语法】
BCLR (:format)　dest
G, S（可指定）

【指令码/周期数】
Page=150

【操作】
dest ← 0

【功能】
・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]*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】
```
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*

条件位传送
Bit Move Condition

BM*Cnd*

【语法】
BM*Cnd*　**dest**

【指令码/周期数】
Page=152

【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。

Cnd	条件		式	*Cnd*	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∧$\overline{Z}$=1	大于	＜	LEU	C∧$\overline{Z}$=0	小于等于	≥
PZ	S=0	正或者零	0≤	N	S=1	负	0＞
GE	S∀O=0	等于或者带符号大于	≤	LE	(S∀O)∨Z=1	等于或者带符号小于	≥
GT	(S∀O)∨Z=0	带符号大于	＜	LT	S∀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*

BNAND

反转位的逻辑与
Bit Not AND carry flag

BNAND

【语法】
BNAND　src

【指令码/周期数】
Page=153

【操作】
C ← $\overline{\text{src}}$ ∧ C

【功能】
・将 C 标志和src的非进行逻辑与运算，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 ：当运算结果为“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

BNOR

反转位的逻辑或
Bit Not OR carry flag

BNOR

【语法】
BNOR　src

【指令码/周期数】
Page=154

【操作】
C ← $\overline{\text{src}}$ ∨ C

【功能】
・将C标志和src的非进行逻辑或运算，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 ：当运算结果为“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

BNOT

位反转
Bit NOT

BNOT

【语法】
BNOT(:format)　dest
G, S（可指定）

【指令码/周期数】
Page=154

【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
	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

反转位的测试
Bit Not TeST

BNTST

【语法】
BNTST　src

【指令码/周期数】
Page=155

【操作】
Z　←　$\overline{src}$
C　←　$\overline{src}$

【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

BNXOR

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

BNXOR

【语法】
BNXOR　src

【指令码/周期数】
Page=156

【操作】
C ← $\overline{src}$ ∀ 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 ：当运算结果为“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

BOR

位逻辑或
Bit OR carry flag

BOR

【语法】
BOR　src

【指令码/周期数】
Page=156

【操作】
C ← src ∨ C

【功能】
・将 C 标志和src进行逻辑或运算，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 ：当运算结果为“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

BRK

调试中断
BReaK

BRK

【语法】
BRK

【指令码/周期数】
Page=157

【操作】

SP ← SP － 2
M(SP) ← (PC +1)H,FLG
SP ← SP － 2
M(SP) ← (PC +1)ML
PC ← M(FFFE4$_{16}$)

【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”。

*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。

【记述例】
BRK

【相关指令】 INT,INTO

BSET

置位
Bit SET

BSET

【语法】
BSET (:format)　dest
　　└──────── **G, S**（可指定）

【指令码/周期数】
Page=157

【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
C	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

位测试
Bit TeST

BTST

【语法】
BTST (:format)　src
　　└──────── **G, S**（可指定）

【指令码/周期数】
Page=158

【操作】
Z　←　$\overline{\text{src}}$
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[SE	bit, base:16
○	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

BTSTC

位测试和清除
Bit TeST & Clear

BTSTC

【语法】
BTSTC　dest

【指令码/周期数】
Page=159

【操作】
Z　← $\overline{\text{dest}}$
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

BTSTS

位测试和置位
Bit TeST & Set

BTSTS

【语法】
BTSTS　dest

【指令码/周期数】
Page=160

【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

BXOR

位逻辑异或
Bit eXclusive OR carry flag

BXOR

【语法】
BXOR　src

【指令码/周期数】
Page=160

【操作】
C ← src ∀ C

【功能】
・将 C 标志和src进行逻辑异或运算，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 ：当运算结果为“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

CMP

比较
CoMPare

CMP

【语法】

CMP.size (:format)　src,dest

G, Q, S（可指定）
B, W

【指令码/周期数】
Page=161

【操作】

dest － src

【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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当带符号的运算结果超过+32767(.W)～－32768(.W)或者+127(.B)～－128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为“0”时置“1”，否则清“0”。
C ：当无符号的运算结果大于等于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≤#IMM≤+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

带进位的10进制加法运算
Decimal ADdition with Carry

DADC

【语法】
DADC.size　src,dest
B, W

【指令码/周期数】
Page=165

【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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号的运算结果超过+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

DADD

无进位的10进制加法运算
Decimal ADDition

DADD

【语法】
DADD.size　src,dest
B, W

【指令码/周期数】
Page=167

【操作】
dest ← src + dest

【功能】
・将dest和src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 ： 当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ： 当运算结果为0时置“1”，否则清“0”。
C ： 当无符号的运算结果超过+9999(.W)或者+99(.B)时置“1”，否则清“0”。

【记述例】
```
DADD.B    #3,R0L
DADD.W    R1,R0
```

【相关指令】　DADC,DSUB,DSBB

DEC

递减
DECrement

DEC

【语法】
DEC.size　dest

B, W

【指令码/周期数】
Page=169

【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 ： 当运算结果的MSB为“1”时置“1”， 否则清“0”。。
Z ： 当运算结果为0时置“1”， 否则清“0”。

【记述例】
```
DEC.W   A0
DEC.B   R0L
```

【相关指令】　INC

DIV

带符号的除法运算
DIVide

DIV

【语法】
DIV.size　src
B, W

【指令码/周期数】
Page=170

【操作】
在长度说明符(.size)为(.B)时
R0L(商)、R0H(余数)←R0÷src
在长度说明符(.size)为(.W)时
R0(商)、R2(余数)←R2R0÷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。()[*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位作为运算对象。
- 在对长度说明符(.size)指定(.B)后，如果运算结果的商超过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，如果运算结果的商超过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 ：当运算结果的商超过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

无符号的除法运算
DIVide Unsigned

DIVU

【语法】
DIVU.size　src
B, W

【指令码/周期数】
Page=171

【操作】
在长度说明符(.size)为(.B)时
R0L(商)、R0H(余数)←R0÷src
在长度说明符(.size)为(.W)时
R0(商)、R2(余数)←R2R0÷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位作为运算对象。
- 在对长度说明符(.size)指定(.B)后，如果运算结果的商超过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，如果运算结果的商超过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 ：当运算结果的商超过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

带符号的除法运算
DIVide eXtension

DIVX

【语法】

DIVX.size　src

B, W

【指令码/周期数】

Page=172

【操作】

在长度说明符(.size)为(.B)时
R0L(商)、R0H(余数)←R0÷src
在长度说明符(.size)为(.W)时
R0(商)、R2(余数)←R2R0÷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位作为运算对象。
- 在对长度说明符(.size)指定(.B)后，如果运算结果的商超过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，如果运算结果的商超过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 ：当运算结果的商超过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

带借位的10进制减法运算

Decimal SuBtract with Borrow

DSBB

【语法】

DSBB.size　src,dest

B, W

【指令码/周期数】

Page=173

【操作】

dest ← dest – 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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。

Z ：当运算结果为0时置“1”，否则清“0”。

C ：当运算结果大于等于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

DSUB

无借位的10进制减法运算
Decimal SUBtract

DSUB

【语法】
DSUB.size　src,dest
B, W

【指令码/周期数】
Page=175

【操作】
dest ← dest – src

【功能】
・以10进制方式从dest减去src，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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当运算结果大于等于0时置“1”，否则清“0”。

【记述例】
```
DSUB.B    #3,R0L
DSUB.W    R1,R0
```

【相关指令】　DADC,DADD,DSBB

ENTER

栈帧的建立
ENTER function

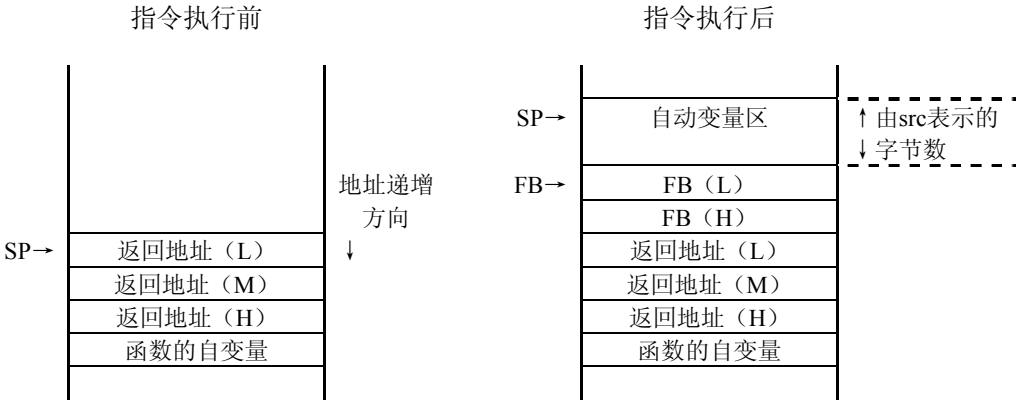
ENTER

【语法】
ENTER　src

【指令码/周期数】
Page=177

【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

EXITD

栈帧的释放
EXIT and Deallocate stack frame

EXITD

【语法】
EXITD

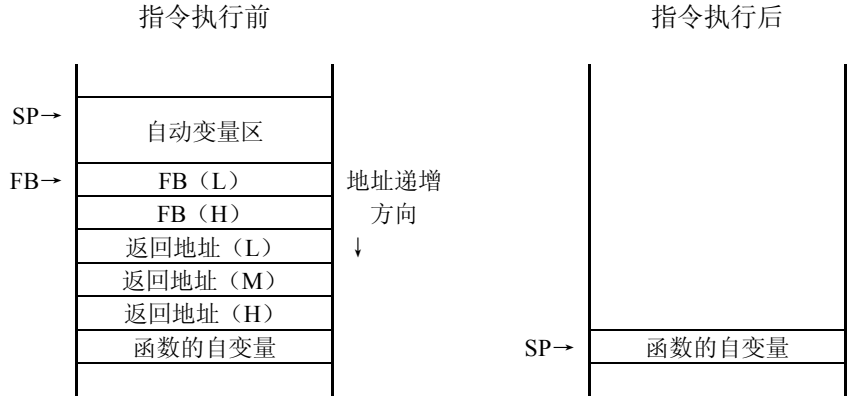
【指令码/周期数】
Page=178

【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

符号扩展
EXTend Sign

EXTS

【语法】
EXTS.size　dest
B, W

【指令码/周期数】
Page=178

【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，如果运算结果的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，如果运算结果为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
```

FCLR

标志寄存器的位清除
Flag register CLeaR

FCLR

【语法】
FCLR　dest

【指令码/周期数】
Page=179

【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

FSET

标志寄存器的置位
Flag register SET

FSET

【语法】
FSET　dest

【指令码/周期数】
Page=180

【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

INC

递增
INCrement

INC

【语法】
INC.size　dest

B, W

【指令码/周期数】
Page=180

【操作】
dest ← dest + 1

【功能】
・给dest加 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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。

【记述例】
```
INC.W   A0
INC.B   R0L
```

【相关指令】 DEC

INT

INT指令中断
INTerrupt

INT

【语法】
INT　　src

【指令码/周期数】
Page=181

【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为31以下时，U标志变为“0”，使用ISP。
- 在src为32以上时，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	○	○	－	－	－	－	○	－

*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。

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”。

【记述例】
```
INT　　#0
```

【相关指令】　BRK,INTO

INTO

溢出中断
INTerrupt on Overflow

INTO

【语法】
INTO

【指令码/周期数】
Page=182

【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】
J*Cnd*　**label**

【指令码/周期数】
Page=182

【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：

Cnd	条件		式	*Cnd*	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∧$\overline{Z}$=1	大于	＜	LEU	C∧$\overline{Z}$=0	小于等于	≥
PZ	S=0	正或者零	0≤	N	S=1	负	0＞
GE	S∀O=0	等于或者带符号大于	≤	LE	(S∀O)∨Z=1	等于或者带符号小于	≥
GT	(S∀O)∨Z=0	带符号大于	＜	LT	S∀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≤label≤PC*1+128	GEU/C,GTU,EQ/Z,N,LTU/NC,LEU,NE/NZ,PZ
PC*1–126≤label≤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

S, B, W, A（可指定）

【指令码/周期数】

Page=183

【操作】

PC ← label

【功能】

• 转移到label。

【可选择的label】

.length	label
.S	PC*1-2≤label≤PC*1+9
.B	PC*1-127≤label≤PC*1+128
.W	PC*1-32767≤label≤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】　JMPI

JMPI

间接转移
JuMP Indirect

JMPI

【语法】
JMPI.length　src

W, A

【指令码/周期数】
Page=185

【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

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=187

【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≤label≤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

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=188

【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

LDC

向专用寄存器的传送
LoaD Control register

LDC

【语法】
LDC　src, dest

【指令码/周期数】
Page=188

【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

LDCTX

环境恢复
LoaD ConTeXt

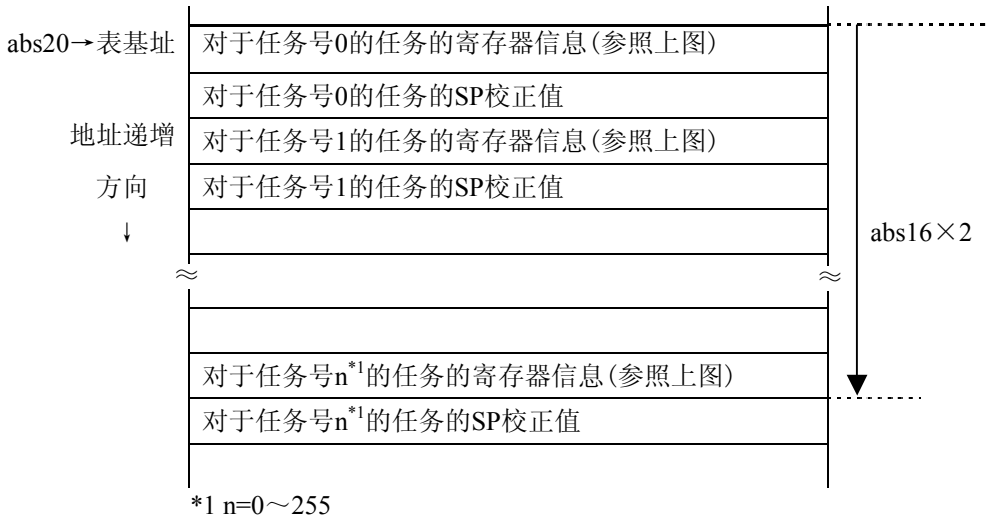
LDCTX

【语法】
LDCTX　abs16,abs20

【指令码/周期数】
Page=190

【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。



【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

LDE

从扩展数据区的传送

LoaD from EXtra far data area

LDE

【语法】

LDE.size　src,dest

B, W

【指令码/周期数】

Page=191

【操作】

dest ← src

【功能】

- 将扩展区中的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。

【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	—	—	—	—	○	○	—	—

条件

S　：当传送结果的dest的MSB为“1”时置“1”，否则清“0”。

Z　：当传送结果的dest为0时置“1”，否则清“0”。

【记述例】

```
LDE.W   [A1A0],R0
LDE.B   Rom_TBL,A0
```

【相关指令】　STE,MOV,XCHG

LDINTB

向INTB寄存器的传送
LoaD INTB register

LDINTB

【语法】
LDINTB　src

【指令码/周期数】
Page=192

【操作】
INTBHL ← src

【功能】
- 将src传送到INTB。
- LDINTB指令是由以下指令构成的微指令。

```
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

LDIPL

中断允许级的设定
LoaD Interrupt Permission Level

LDIPL

【语法】
LDIPL　src

【指令码/周期数】
Page=193

【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≤#IMM≤7。

【标志变化】

标志	U	I	O	B	S	Z	D	C
变化	—	—	—	—	—	—	—	—

【记述例】
LDIPL　#2

MOV

传送
MOVe

MOV

【语法】

MOV.size (:format)　src,dest

G, Q, Z, S（可指定）
B, W

【指令码/周期数】
Page=193

【操作】

dest ← src

【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 运算对象为由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 运算对象为由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≤#IMM≤+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

有效地址的传送
MOVe effective Address

MOVA

【语法】
MOVA　src,dest

【指令码/周期数】
Page=200

【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

MOV*Dir*

4位数据的传送
MOVe nibble

MOV*Dir*

【语法】
MOV*Dir*　src,dest

【指令码/周期数】
Page=201

【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

带符号的乘法运算
MULtiple

MUL

【语法】
MOVL.size　src,dest
B, W

【指令码/周期数】
Page=203

【操作】
dest ← dest × src

【功能】
- 将src和dest进行带符号乘法运算，结果保存到dest。
- 如果对长度说明符(.size)指定(.B)，src和dest都以8位运算，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，就运算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位运算，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位进行运算。
MUL.W   #3,R0
MUL.B   R0L,R1L
MUL.W   A0,Ram
```

【相关指令】　DIV,DIVU,DIVX,MULU

MULU

无符号的乘法运算
MULtiple Unsigned

MULU

【语法】
MULU.size　src,dest
　　B, W

【指令码/周期数】
Page=205

【操作】
dest ← dest × src

【功能】
- 将src和dest进行无符号乘法运算，结果保存到dest。
- 如果对长度说明符(.size)指定(.B)，src和dest都以8位运算，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，就运算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位运算，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位进行运算。
MULU.W    #3,R0
MULU.B    R0L,R1L
MULU.W    A0,Ram
```

【相关指令】　DIV,DIVU,DIVX,MUL

NEG

2的补码
NEGate

NEG

【语法】
NEG.size　dest
B, W

【指令码/周期数】
Page=207

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

【记述例】
```
NEG.B   R0L
NEG.W   A1
```

【相关指令】　NOT

NOP

空操作
No OPeration

NOP

【语法】
NOP

【指令码/周期数】
Page=207

【操作】
PC ← 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

全位反转
NOT

NOT

【语法】

NOT.size (:format)　dest

G, S（可指定）
B, W

【指令码/周期数】
Page=208

【操作】

dest ← $\overline{\text{dest}}$

【功能】

・将dest的非保存到dest。

【可选择的dest】

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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。

【记述例】

```
NOT.B   R0L
NOT.W   A1
```

【相关指令】　NEG

OR

逻辑或
OR

OR

【语法】

OR.size (:format)　src,dest

G, S（可指定）
B, W

【指令码/周期数】
Page=209

【操作】

dest ← src ∨ dest

【功能】

- 将dest和src进行逻辑或运算，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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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　：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z　：当运算结果为0时置“1”，否则清“0”。

【记述例】

```
OR.B    Ram:8[SB],R0L
OR.B:G  A0,R0L          ; 将A0的低8位和R0L进行运算。
OR.B:G  R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
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。

POP

寄存器/存储器的恢复
POP

POP

【语法】

POP.size (:format)　dest

G, S（可指定）
B, W

【指令码/周期数】
Page=211

【操作】

在长度说明符(.size)为(.B)时
dest　← M(SP)
SP　← SP + 1

在长度说明符(.size)为(.W)时
dest　← M(SP)
SP　← SP + 2

【功能】

・将dest从堆栈区恢复。

【可选择的dest】

dest			
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	
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	—	—	—	—	—	—	—	—

【记述例】

```
POP.B   R0L
POP.W   A0
```

【相关指令】　PUSH,POPM,PUSHM

POPC

专用寄存器的恢复
POP Control register

POPC

【语法】
POPC　dest

【指令码/周期数】
Page=213

【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

POPM

多个寄存器的恢复
POP Multiple

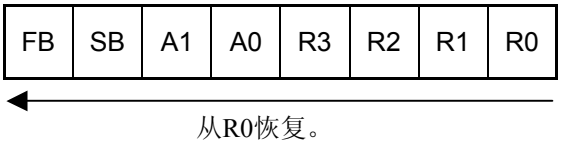
POPM

【语法】
POPM　dest

【指令码/周期数】
Page=213

【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

PUSH

寄存器/存储器/立即数的保存
PUSH

PUSH

【语法】

PUSH.size (:format)　src

G, S（可指定）
B, W

【指令码/周期数】

Page=214

【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

有效地址的保存
PUSH effective Address

PUSHA

【语法】
PUSHA　src

【指令码/周期数】
Page=216

【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

PUSHC

专用寄存器的保存
PUSH Control register

PUSHC

【语法】
PUSHC　src

【指令码/周期数】
Page=216

【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

PUSHM

多个寄存器的保存
PUSH Multiple

PUSHM

【语法】
PUSHM　src

【指令码/周期数】
Page=217

【操作】
SP　　← SP － N[*1] × 2
M(SP)　← 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

REIT

从中断的返回
REturn from InTerrupt

REIT

【语法】
REIT

【指令码/周期数】
Page=217

【操作】

PC$_{ML}$ ← M(SP)
SP ← SP + 2
PC$_{H}$,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

乘加运算
Repeat MultiPle & Addition

RMPA

【语法】
RMPA.size
B, W

【指令码/周期数】
Page=218

【操作】*1
Repeat
R2R0(R0) *2 ← R2R0(R0) *2 + M(A0) × M(A1)
A0 ← A0 + 2 (1) *2
A1 ← A1 + 2 (1) *2
R3 ← 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作为运算次数，进行乘加运算。运算以带符号进行，结果保存到R2R0(R0) *1。
- 如果在运算中溢出，O 标志就变为“1”，结束运算。在R2R0(R0) *1中保存最后的加法运算结果。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，就在乘加运算的加法运算结束后(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 ：在运算中超过+2147483647(.W)～－2147483648(.W)或者+32767(.B)～－32768(.B)时置“1”，否则清“0”。

【记述例】
RMPA.B

ROLC

带进位的循环左移
ROtate to Left with Carry

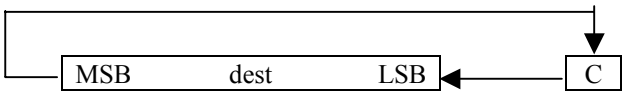
ROLC

【语法】
ROLC.size　dest

B, W

【指令码/周期数】
Page=218

【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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果的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

RORC

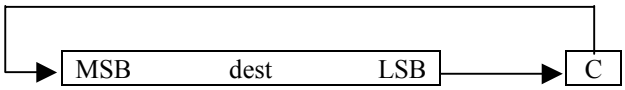
带进位的循环右移
ROtate to Right with Carry

RORC

【语法】
RORC.size　dest
B, W

【指令码/周期数】
Page=219

【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	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 ： 当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ： 当运算结果的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

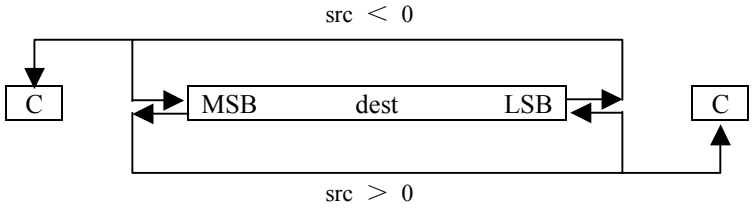
循环移位
ROTate

ROT

【语法】
ROT.size　src,dest
B, W

【指令码/周期数】
Page=220

【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	
R2R0	R3R1	A1A0		R2R0	R3R1	A1A0	

*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≤#IMM≤+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
S ：当运算结果为MSB＝“1”时置“1”，否则清“0”。
Z ：当运算结果为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

从子程序的返回
ReTurn from Subroutine

RTS

【语法】
RTS

【指令码/周期数】
Page=221

【操作】

PC_{ML}	←	M(SP)
SP	←	SP + 2
PC_{H}	←	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=222

【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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当带符号运算的结果超过+32767(.W)～−32768(.W)或者+127(.B)～−128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号运算的结果大于等于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进行运算。
SBB.B   R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
```

【相关指令】　ADC,ADCF,ADD,SUB

SBJNZ

减法运算和条件转移

SuBtract then Jump on Not Zero

SBJNZ

【语法】

SBJNZ.size　src,dest,label

B, W

【指令码/周期数】

Page=224

【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 R1H/R3 [A0] dsp:8[A1] dsp:16[A0] abs16	R0H/R1 A0/A0 [A1] dsp:8[SB] dsp:16[A1]	R1L/R2 A1/A1 dsp:8[A0] dsp:8[FB] dsp:16[SB]	PC*2 –126≤label≤PC*2 +129

*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

算术移位
SHift Arithmetic

SHA

【语法】

SHA.size　src,dest

B, W, L

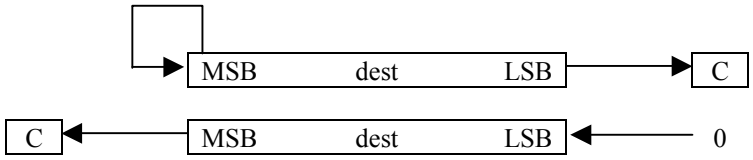
【指令码/周期数】

Page=225

【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]	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 ：当运算结果为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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为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

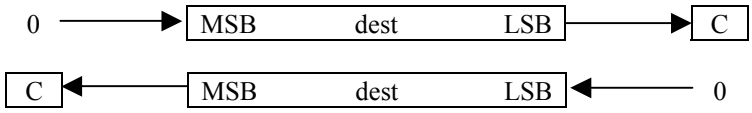
【指令码/周期数】

Page=228

【操作】

当src ＜ 0时

当src ＞ 0时



【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]	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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为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=230

【操作】

在长度说明符(.size)为(.B)时

Repeat
M(A1) ← M(2^{16} ×R1H +A0)
A0[*2] ← A0 － 1
A1 ← A1 － 1
R3 ← R3 － 1
Until R3 =0

在长度说明符(.size)为(.W)时

Repeat
M(A1) ← M(2^{16} ×R1H +A0)
A0[*2] ← A0 － 2
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。
*2 当A0下溢时，R1H的内容被减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=231

【操作】

在长度说明符(.size)为(.B)时

Repeat
M(A1) ← M(2^{16} ×R1H +A0)
A0[*2] ← A0 + 1
A1 ← A1 + 1
R3 ← R3 － 1
Until R3 =0

在长度说明符(.size)为(.W)时

Repeat
M(A1) ← M(2^{16} ×R1H +A0)
A0[*2] ← A0 + 2
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。
*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

字符串保存
String SToRe

SSTR

【语法】
SSTR.size
B, W

【指令码/周期数】
Page=231

【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

从专用寄存器的传送
STore from Control register

STC

【语法】
STC　src, dest

【指令码/周期数】
Page=232

【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

环境保存
STore ConTeXt

STCTX

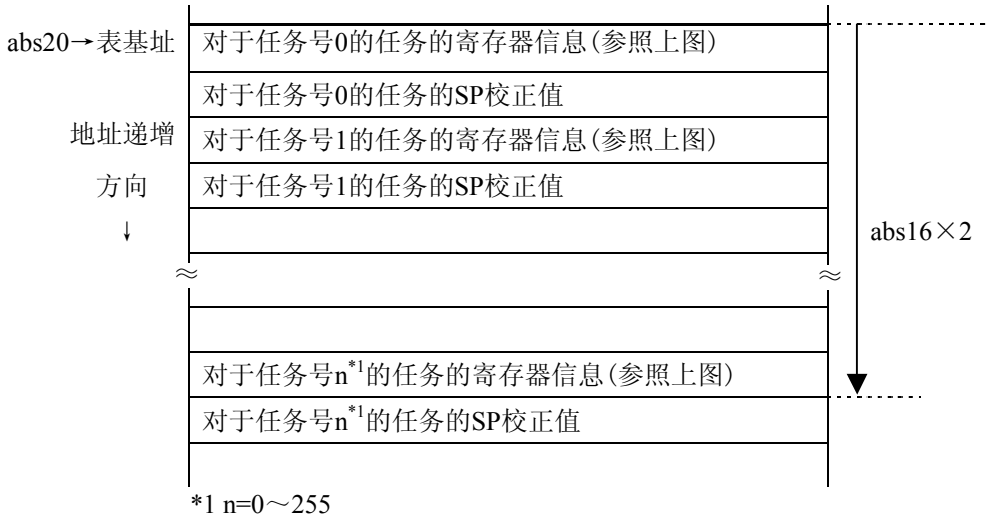
【语法】
STCTX　abs16,abs20

【指令码/周期数】
Page=233

【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

STE

向扩展数据区的传送

STore to EXtra far data area

STE

【语法】

STE.size　src,dest

B, W

【指令码/周期数】

Page=233

【操作】

dest ← src

【功能】

- 将src传送到扩展区中的dest。
- 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位作为运算对象。但是，标志以运算前的A0或者A1的状态(16位)变化。

【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　：当运算结果的MSB为“1”时置“1”，否则清“0”。

Z　：当运算结果为0时置“1”，否则清“0”。

【记述例】

```
STE.B    R0L,[A1A0]
STE.W    R0,10000H[A0]
```

【相关指令】　MOV,LDE,XCHG

STNZ

条件传送
STore on Not Zero

STNZ

【语法】
STNZ　src,dest

【指令码/周期数】
Page=235

【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	R0H	dsp:8[SB]	dsp:8[FB]
	abs16	A0	A1	

【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

STZ

条件传送
STore on Zero

STZ

【语法】
STZ　src,dest

【指令码/周期数】
Page=235

【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	R0H	dsp:8[SB]	dsp:8[FB]
	abs16	A0	A1	

【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

条件传送
STore on Zero eXtention

STZX

【语法】
STZX　src1,src2,dest

【指令码/周期数】
Page=236

【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	R0H	dsp:8[SB]	dsp:8[FB]
	abs16	A0	A1	

【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

SUB

无借位的减法运算
SUBtract

SUB

【语法】

SUB.size (:format)　src,dest

G, S（可指定）
B, W

【指令码/周期数】
Page=236

【操作】

dest ← dest − src

【功能】

- 从dest减去src，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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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	
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 ：当带符号运算的结果超过+32767(.W)～−32768(.W)或者+127(.B)～−128(.B)时置“1”，否则清“0”。
S ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为0时置“1”，否则清“0”。
C ：当无符号运算的结果大于等于0时置“1”，否则清“0”。

【记述例】

```
SUB.B   A0,R0L                  ; 将A0的低8位和R0L进行运算。
SUB.B   R0L,A0                  ; 在将R0L零扩展后，和A0进行运算。
SUB.B   Ram:8[SB],R0L
SUB.W   #2,[A0]
```

【相关指令】　ADC,ADCF,ADD,SBB

【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

测试
TeST

TST

【语法】
TST.size　src,dest
B, W

【指令码/周期数】
Page=239

【操作】
dest ∧ 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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当运算结果的MSB为“1”时置“1”，否则清“0”。
Z ：当运算结果为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R0L进行运算。
TST.B   R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
```

【相关指令】　AND,OR,XOR

UND

未定义指令中断
UNDefined instruction

UND

【语法】
UND

【指令码/周期数】
Page=241

【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

等待
WAIT

WAIT

【语法】
WAIT

【指令码/周期数】
Page=241

【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

交换
eXCHanGe

XCHG

【语法】
XCHG.size　src,dest
B, W

【指令码/周期数】
Page=242

【操作】
dest ←→ 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

XOR

逻辑异或
eXclusive OR

XOR

【语法】

XOR.size　src,dest

B, W

【指令码/周期数】

Page=243

【操作】

dest ← dest ∀ src

【功能】

- 将src和dest进行逻辑异或运算，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，然后进行运算。另外，如果src为A0或者A1，就将A0或者A1的低8位作为运算对象。

【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 ：当运算结果为MSB＝“1”时置“1”，否则清“0”。

Z ：当运算结果为0时置“1”，否则清“0”。

【记述例】

```
XOR.B   A0,R0L          ; 将A0的低8位和R0L进行运算。
XOR.B   R0L,A0          ; 在将R0L零扩展后，和A0进行运算。
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

在本章中，按操作码说明指令码和周期数。

有关本章的阅读方法，举例说明如下：

① LDIPL

② (1) LDIPL #IMM

③

b7							b0	b7							b0
0	1	1	1	1	1	0	1	1	0	1	0	IMM4			

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

b7							b0	b7					b0
0	1	1	1	0	1	0	SIZE	1	1	0	0	DEST	

dest码：dsp8 / dsp16/abs16 ⟩ #IMM8 / #IMM16

.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

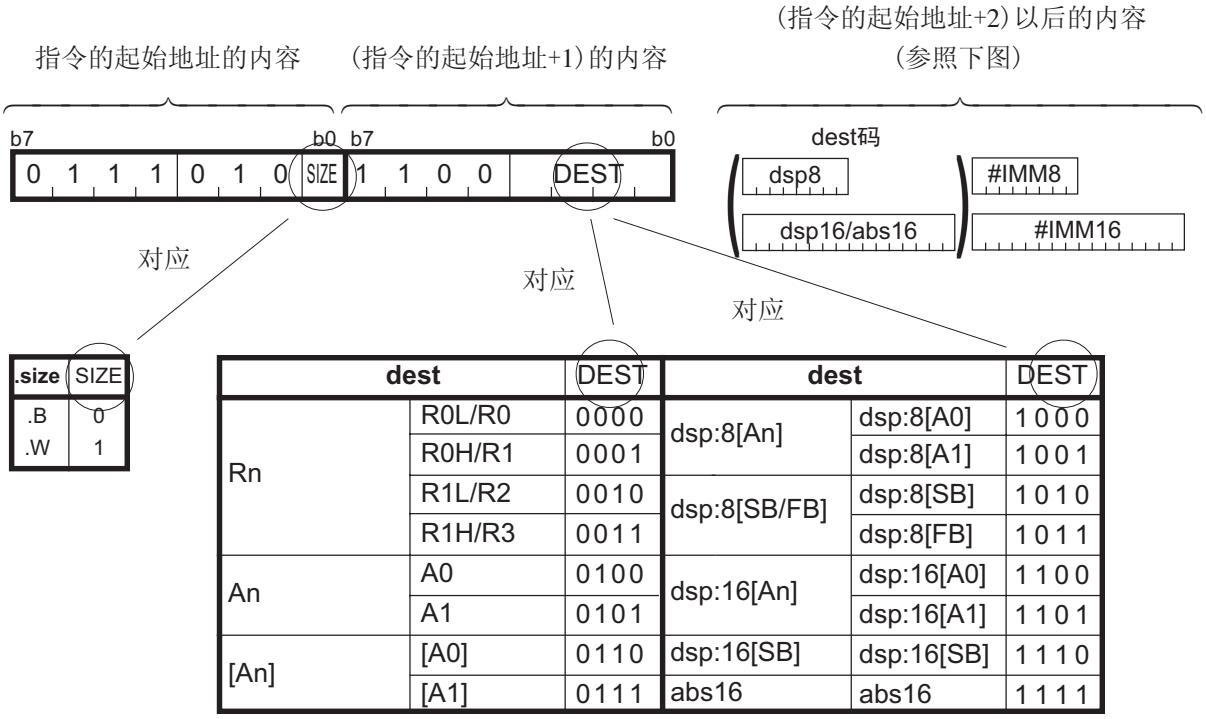
表示在本页中说明的助记符。

② 语法

用符号表示指令的语法。

③ 指令码

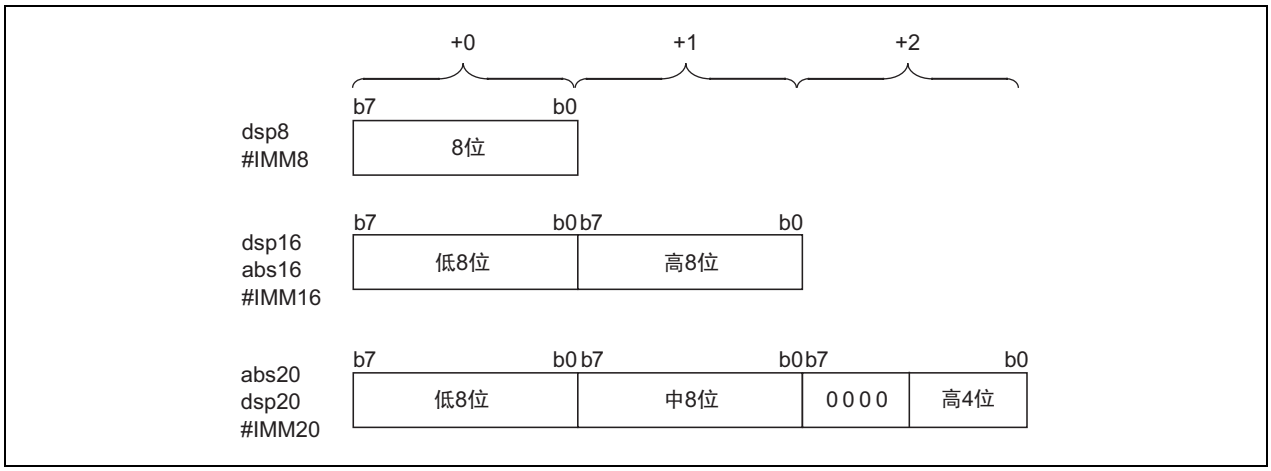
表示指令码。()中的内容根据选择的src/dest将被省略。



.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

（指令的起始地址＋2）以后的内容分配如下：



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。

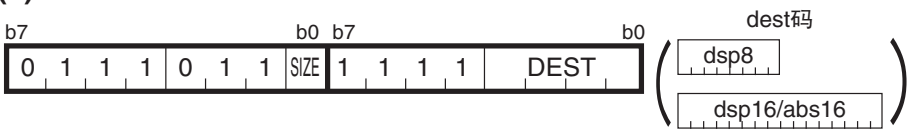
但是，周期数根据软件等待等的影响可能会增加。

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

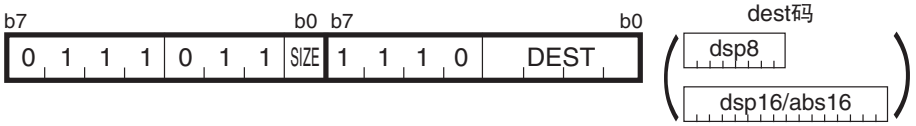
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

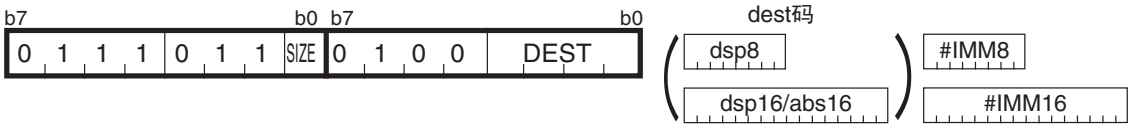
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

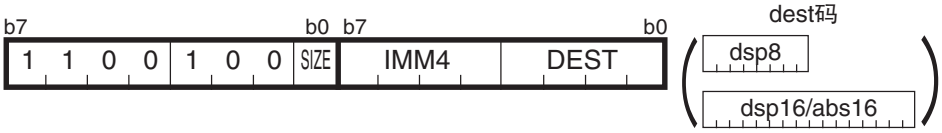
【字节数/周期数】

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

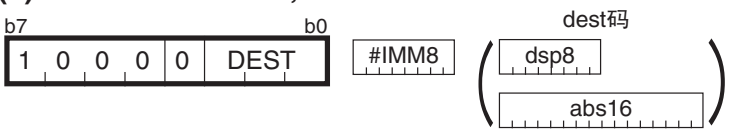
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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



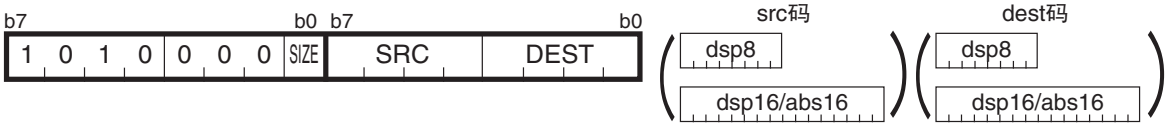
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

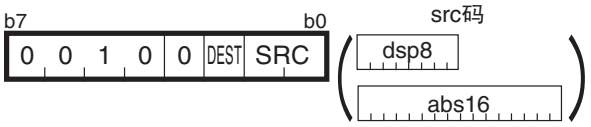
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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



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

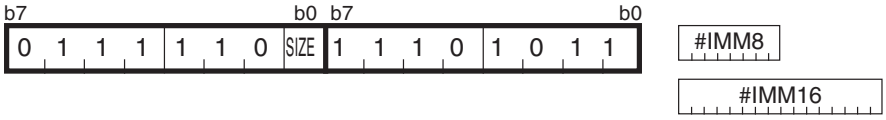
dest	DEST
R0L	0
R0H	1

【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



.size	SIZE
.B	0
.W	1

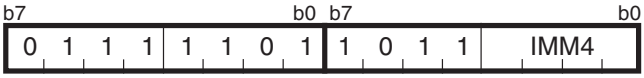
【字节数/周期数】

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。

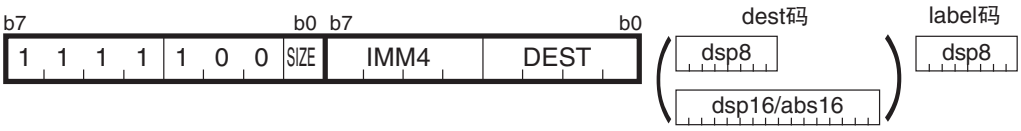
#IMM	IMM4	#IMM	IMM4
0	0 0 0 0	−8	1 0 0 0
+1	0 0 0 1	−7	1 0 0 1
+2	0 0 1 0	−6	1 0 1 0
+3	0 0 1 1	−5	1 0 1 1
+4	0 1 0 0	−4	1 1 0 0
+5	0 1 0 1	−3	1 1 0 1
+6	0 1 1 0	−2	1 1 1 0
+7	0 1 1 1	−1	1 1 1 1

【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

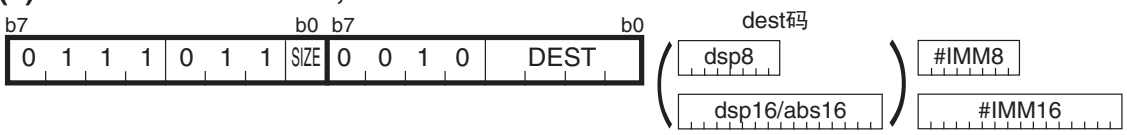
【字节数/周期数】

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

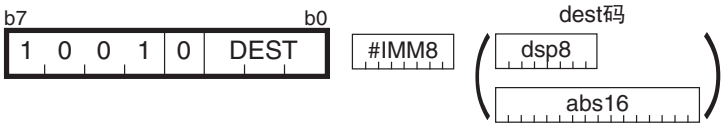
【字节数/周期数】

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



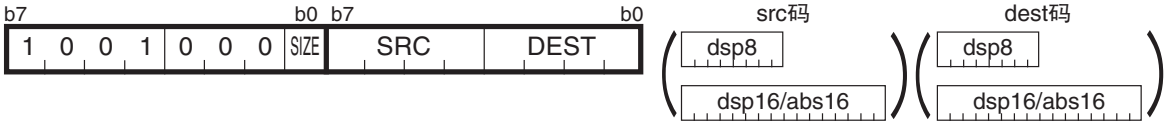
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

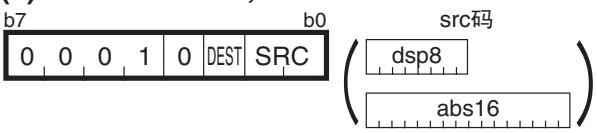
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

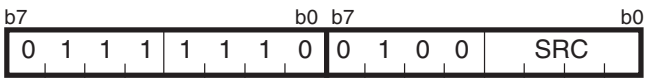
dest	DEST
R0L	0
R0H	1

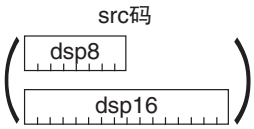
【字节数/周期数】

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

BAND

(1) BAND　　src





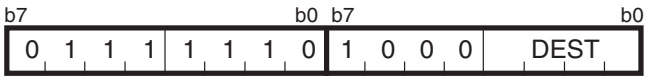
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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

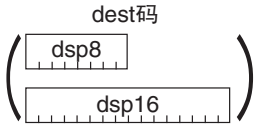
【字节数/周期数】

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

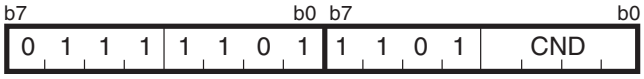
Cnd	CND	***Cnd***	CND
GEU/C	0 0 0 0 0 0 0 0	LTU/NC	1 1 1 1 1 0 0 0
GTU	0 0 0 0 0 0 0 1	LEU	1 1 1 1 1 0 0 1
EQ/Z	0 0 0 0 0 0 1 0	NE/NZ	1 1 1 1 1 0 1 0
N	0 0 0 0 0 0 1 1	PZ	1 1 1 1 1 0 1 1
LE	0 0 0 0 0 1 0 0	GT	1 1 1 1 1 1 0 0
O	0 0 0 0 0 1 0 1	NO	1 1 1 1 1 1 0 1
GE	0 0 0 0 0 1 1 0	LT	1 1 1 1 1 1 1 0

【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*

(2) BM*Cnd*　　C



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

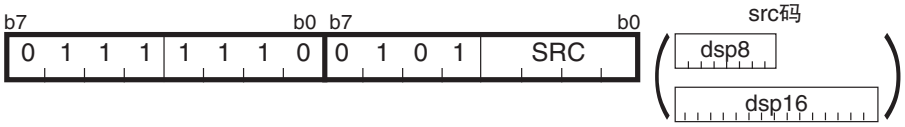
【字节数/周期数】

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

(1) BNAND　　src



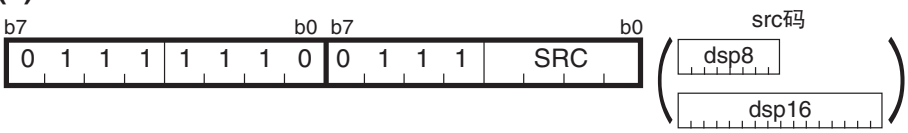
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1	[SB/FB]	bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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

BNOR

(1) BNOR　　src



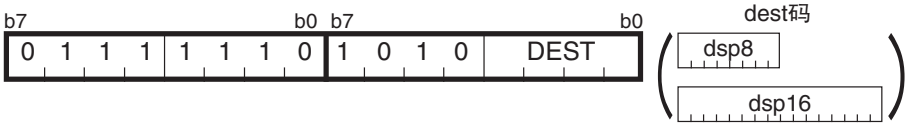
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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

BNOT

(1) BNOT: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

BNOT

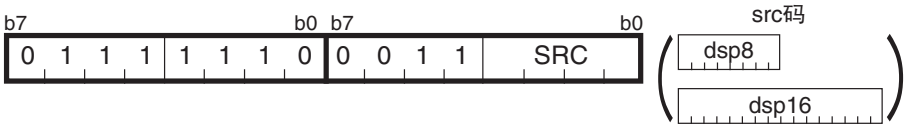
(2) BNOT:S　　bit, base:11[SB]

【字节数/周期数】

字节数/周期数	2/3

BNTST

(1) BNTST　　src



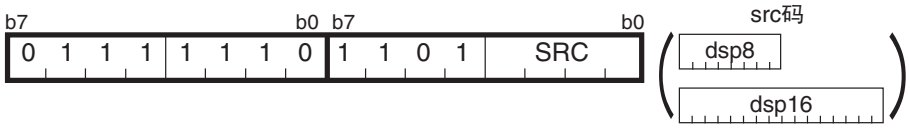
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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



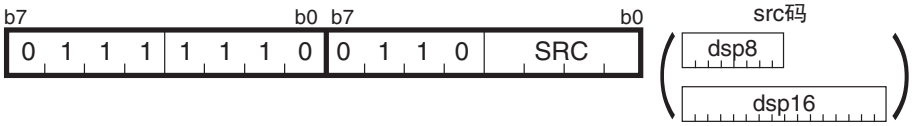
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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



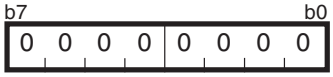
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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



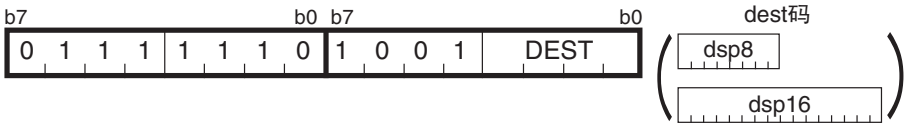
【字节数/周期数】

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FFFE4$_{16}$地址～FFFE7$_{16}$地址中设定FF$_{16}$。

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

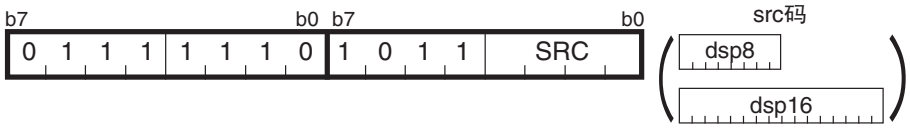
(2) BSET:S　　bit, base:11[SB]

【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

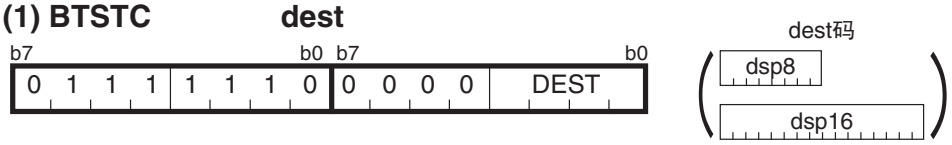
(2) BTST:S　　bit, base:11[SB]

【字节数/周期数】

字节数/周期数	2/3

BTSTC

(1) BTSTC　　dest



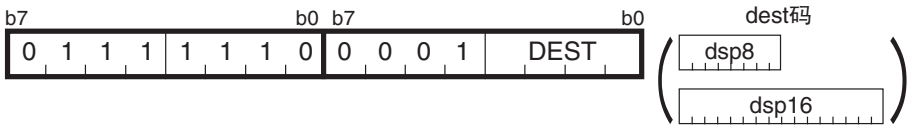
dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]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



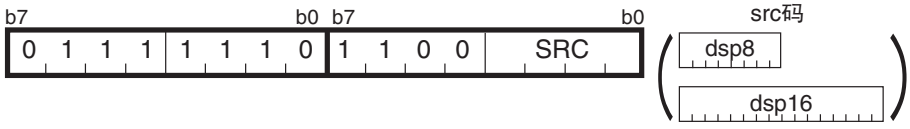
dest		DEST	dest		DEST
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]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



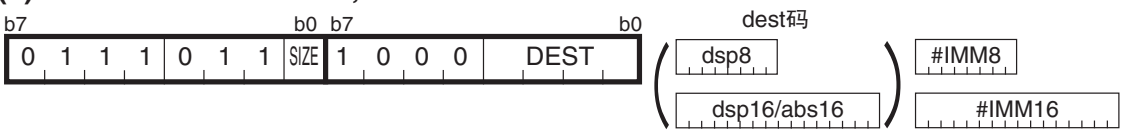
src		SRC	src		SRC
bit,Rn	bit,R0	0 0 0 0	base:8[An]	base:8[A0]	1 0 0 0
	bit,R1	0 0 0 1		base:8[A1]	1 0 0 1
	bit,R2	0 0 1 0	bit,base:8 [SB/FB]	bit,base:8[SB]	1 0 1 0
	bit,R3	0 0 1 1		bit,base:8[FB]	1 0 1 1
bit,An	bit,A0	0 1 0 0	base:16[An]	base:16[A0]	1 1 0 0
	bit,A1	0 1 0 1		base:16[A1]	1 1 0 1
[An]	[A0]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

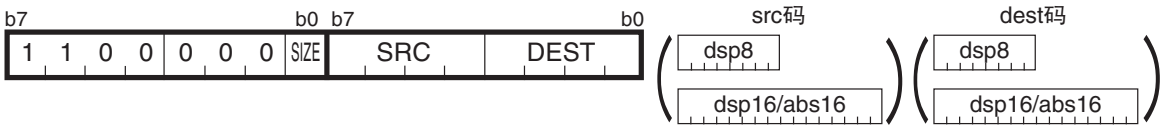
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

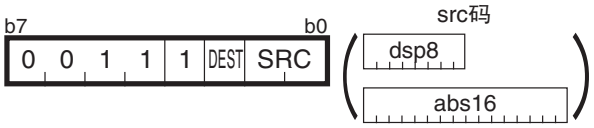
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

CMP

(5) CMP.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

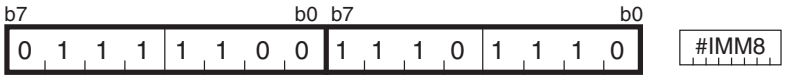
dest	DEST
R0L	0
R0H	1

【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

DADC

(1) DADC.B　　#IMM8, R0L

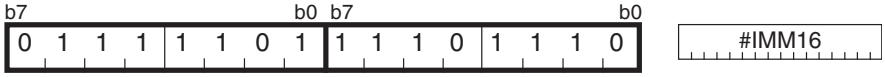


【字节数/周期数】

字节数/周期数	3/5

DADC

(2) DADC.W　　#IMM16, R0

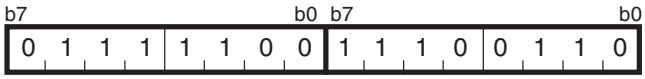


【字节数/周期数】

字节数/周期数	4/5

DADC

(3) DADC.B　　R0H, R0L

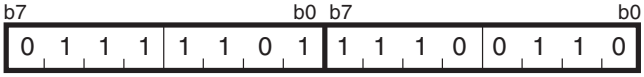


【字节数/周期数】

字节数/周期数	2/5

DADC

(4) DADC.W　　R1, R0

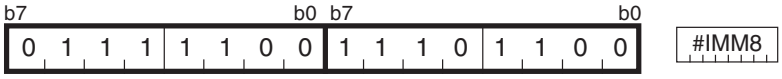


【字节数/周期数】

字节数/周期数	2/5

DADD

(1) DADD.B　　#IMM8, R0L

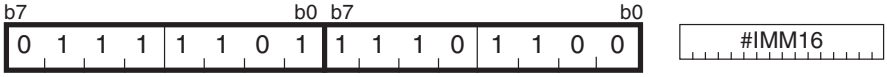


【字节数/周期数】

字节数/周期数	3/5

DADD

(2) DADD.W　　#IMM16, R0

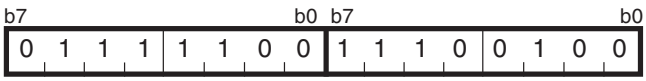


【字节数/周期数】

字节数/周期数	4/5

DADD

(3) DADD.B　　R0H, R0L

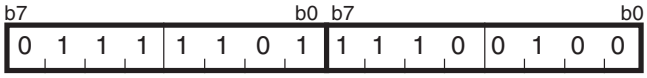


【字节数/周期数】

字节数/周期数	2/5

DADD

(4) DADD.W　　R1, R0

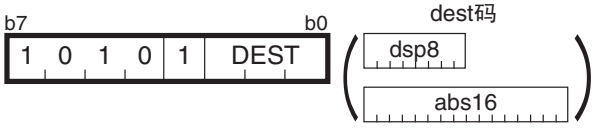


【字节数/周期数】

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

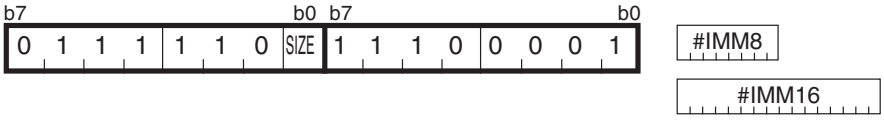
dest	DEST
A0	0
A1	1

【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

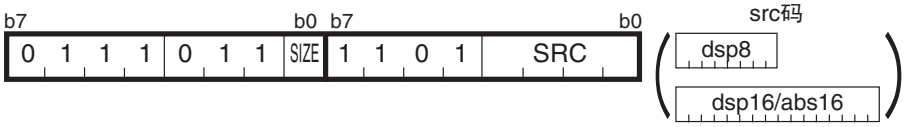
【字节数/周期数】

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
.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

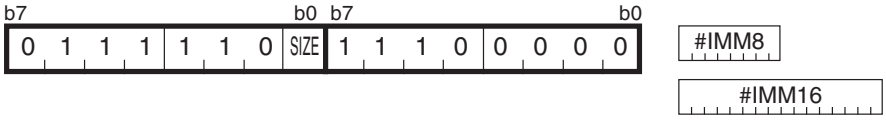
【字节数/周期数】

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

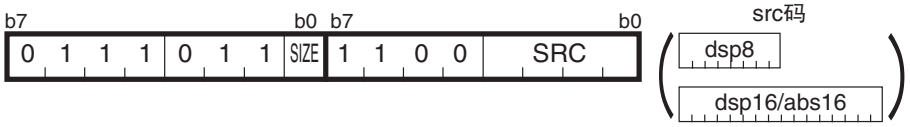
【字节数/周期数】

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
.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

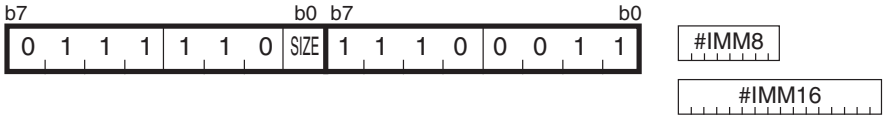
【字节数/周期数】

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

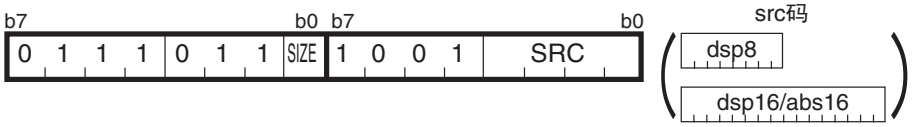
【字节数/周期数】

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[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

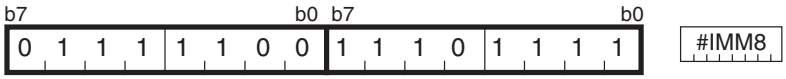
【字节数/周期数】

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。

DSBB

(1) DSBB.B　　#IMM8, R0L

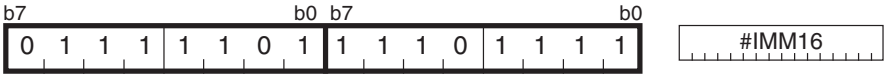


【字节数/周期数】

字节数/周期数	3/4

DSBB

(2) DSBB.W　　#IMM16, R0

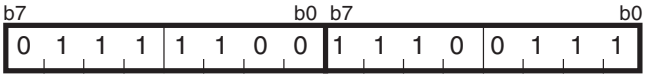


【字节数/周期数】

字节数/周期数	4/4

DSBB

(3) DSBB.B　　R0H, R0L

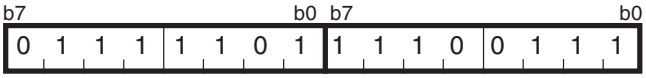


【字节数/周期数】

字节数/周期数	2/4

DSBB

(4) DSBB.W　　R1, R0

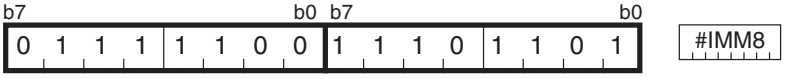


【字节数/周期数】

字节数/周期数	2/4

DSUB

(1) DSUB.B　　#IMM8, R0L

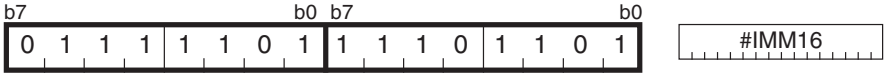


【字节数/周期数】

字节数/周期数	3/4

DSUB

(2) DSUB.W　　#IMM16, R0

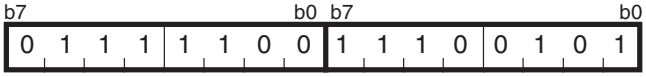


【字节数/周期数】

字节数/周期数	4/4

DSUB

(3) DSUB.B　　R0H, R0L

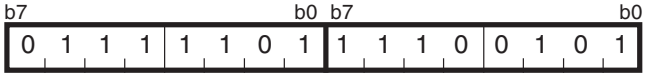


【字节数/周期数】

字节数/周期数	2/4

DSUB

(4) DSUB.W　　R1, R0

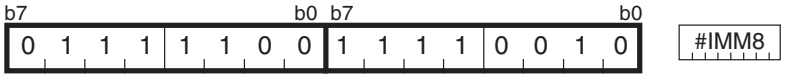


【字节数/周期数】

字节数/周期数	2/4

ENTER

(1) ENTER　　#IMM8

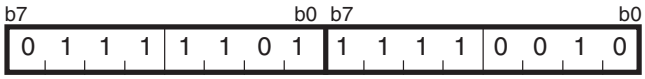


【字节数/周期数】

字节数/周期数	3/4

EXITD

(1) EXITD

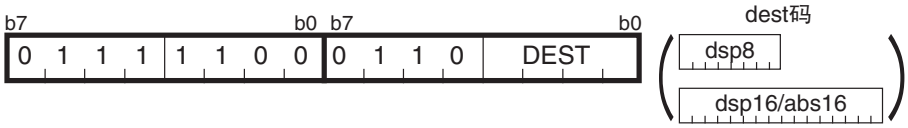


【字节数/周期数】

字节数/周期数	2/9

EXTS

(1) EXTS.B　　dest



dest		DEST	dest		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

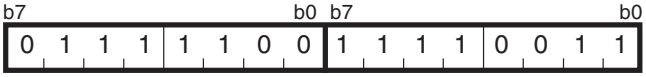
*1 记号“---”为不能选择。

【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

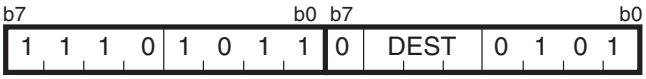


【字节数/周期数】

字节数/周期数	2/3

FCLR

(1) FCLR　　dest



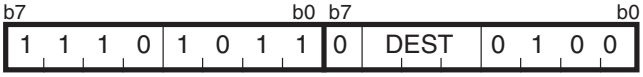
dest	DEST
C	0 0 0
D	0 0 1
Z	0 1 0
S	0 1 1
B	1 0 0
O	1 0 1
I	1 1 0
U	1 1 1

【字节数/周期数】

字节数/周期数	2/2

FSET

(1) FSET　　dest



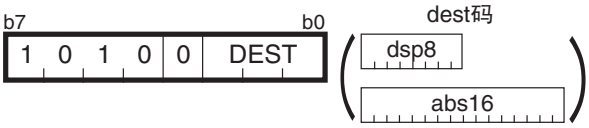
dest	DEST
C	0 0 0
D	0 0 1
Z	0 1 0
S	0 1 1
B	1 0 0
O	1 0 1
I	1 1 0
U	1 1 1

【字节数/周期数】

字节数/周期数	2/2

INC

(1) INC.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

INC

(2) INC.W　　dest

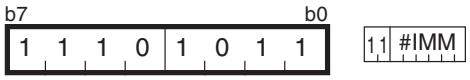
dest	DEST
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/1

INT

(1) INT　　#IMM

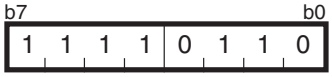


【字节数/周期数】

字节数/周期数	2/19

INTO

(1) INTO



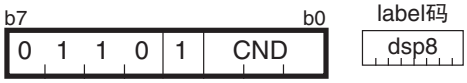
【字节数/周期数】

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。

J*Cnd*

(1) J*Cnd*　　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

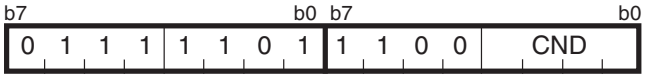
【字节数/周期数】

字节数/周期数	2/2

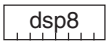
*2 在转移到label时，表中的周期数增加2个周期。

JCnd

(2) J*Cnd*　　label



label码



dsp8＝label表示的地址－(指令的起始地址＋2)

Cnd	CND	*Cnd*	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

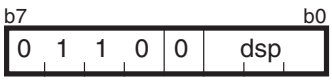
【字节数/周期数】

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



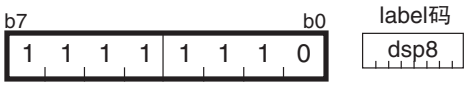
dsp＝label表示的地址－(指令的起始地址＋2)

【字节数/周期数】

字节数/周期数	1/5

JMP

(2) JMP.B　　label



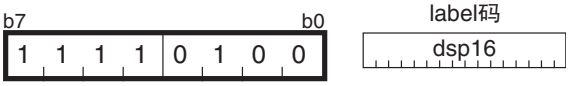
dsp8＝label表示的地址－(指令的起始地址＋1)

【字节数/周期数】

字节数/周期数	2/4

JMP

(3) JMP.W　　label



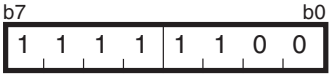
dsp16＝label表示的地址－(指令的起始地址＋1)

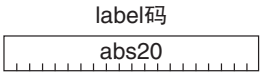
【字节数/周期数】

字节数/周期数	3/4

JMP

(4) JMP.A　　label



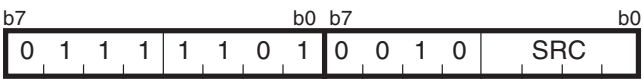


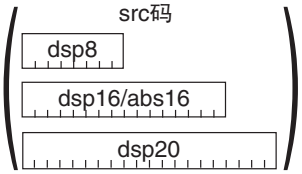
【字节数/周期数】

字节数/周期数	4/4

JMPI

(1) JMPI.W　　src





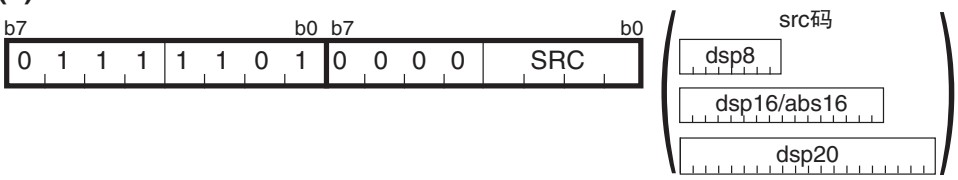
src		SRC	src		SRC
Rn	R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	A1	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1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



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

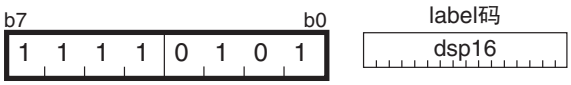
*1 记号“---”为不能选择。

【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

JSR

(1) JSR.W　　label



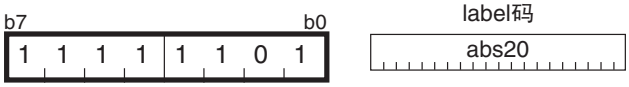
dsp16＝label表示的地址－(指令的起始地址＋1)

【字节数/周期数】

字节数/周期数	3/8

JSR

(2) JSR.A　　label

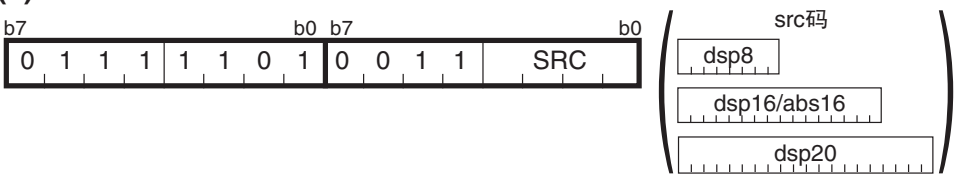


【字节数/周期数】

字节数/周期数	4/9

JSRI

(1) JSRI.W　　src



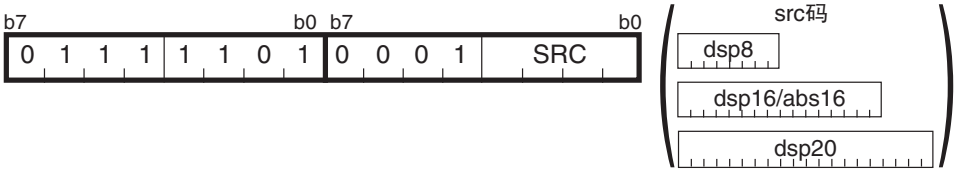
src		SRC	src		SRC
Rn	R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:20[An]	dsp:20[A0]	1 1 0 0
	A1	0 1 0 1		dsp:20[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1 1 0
	[A1]	0 1 1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

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
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

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

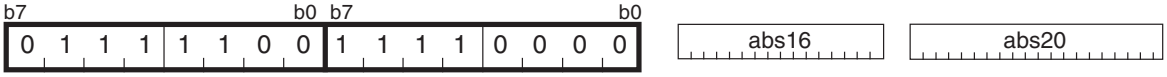
*1 记号“---”为不能选择。

【字节数/周期数】

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



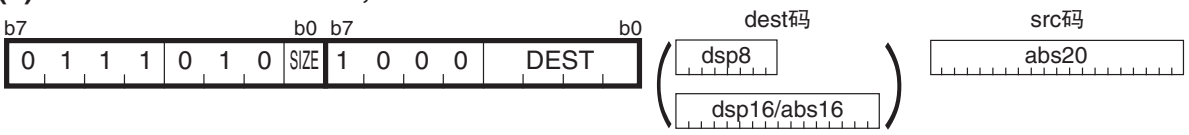
【字节数/周期数】

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

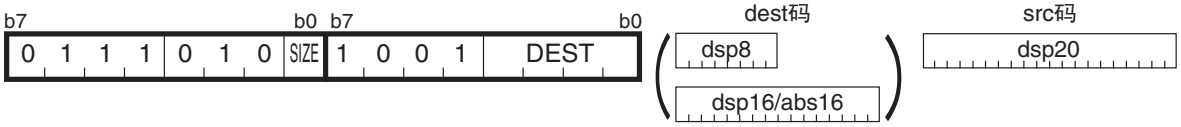
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

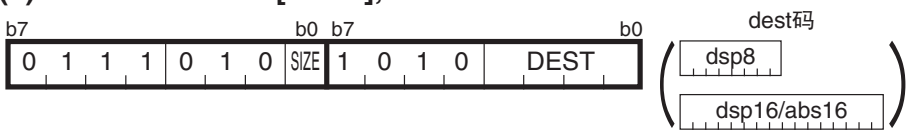
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

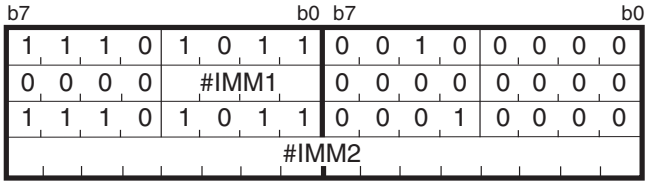
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	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	2/4	2/4	2/5	3/5	3/5	4/5	4/5	4/5

LDINTB

(1) LDINTB　　#IMM



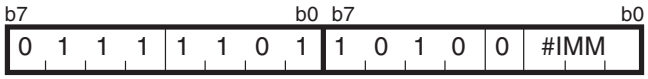
*1 #IMM1为#IMM的高4位
　#IMM2为#IMM的低16位。

【字节数/周期数】

字节数/周期数	8/4

LDIPL

(1) LDIPL　　#IMM

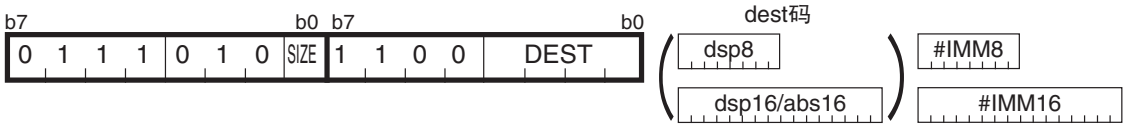


【字节数/周期数】

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

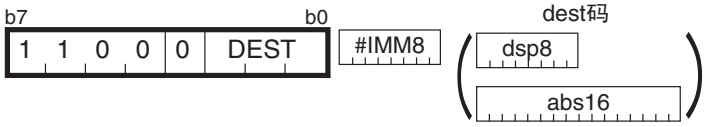
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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



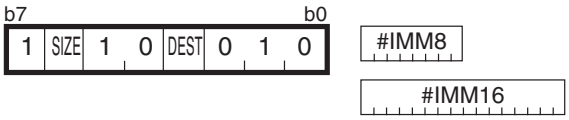
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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



.size	SIZE
.B	1
.W	0

dest	DEST
A0	0
A1	1

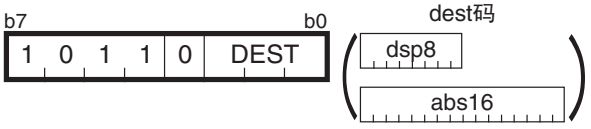
【字节数/周期数】

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



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	2/2	3/2

MOV

(6) MOV.size:G　　src, dest

.size	SIZE
.B	0
.W	1

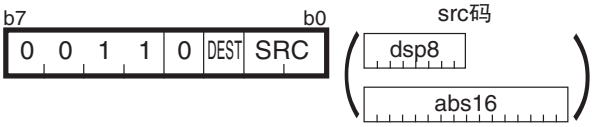
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

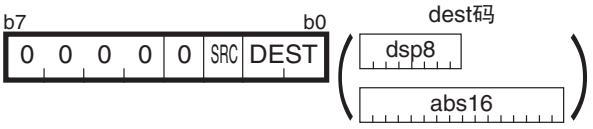
dest	DEST
A0	0
A1	1

【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

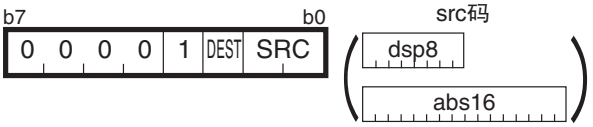
dest		DEST
dsp:8[SB/FB]	dsp:8[SB]	0 1
	dsp:8[FB]	1 0
abs16	abs16	1 1

【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

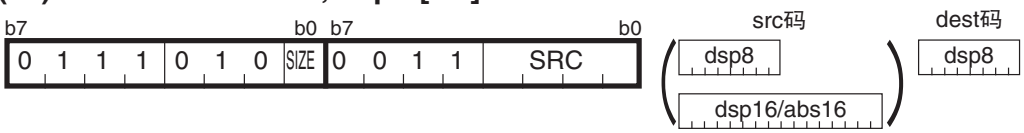
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

Dir	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

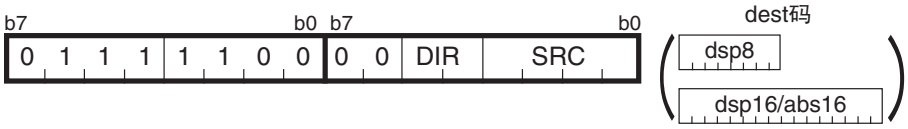
*1 记号“---”为不能选择。

【字节数/周期数】

dest	Rn	[An]	dsp:8[An]	dsp:8[SB/FB]	dsp:16[An]	dsp:16[SB]	abs16
MOVHH, MOVLL	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

MOV*Dir*

(2) MOV*Dir*　　src, R0L



Dir	DIR
LL	0 0
LH	1 0
HL	0 1
HH	1 1

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

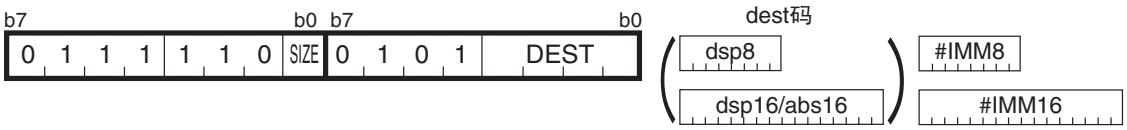
*1 记号“---”为不能选择。

【字节数/周期数】

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



.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。

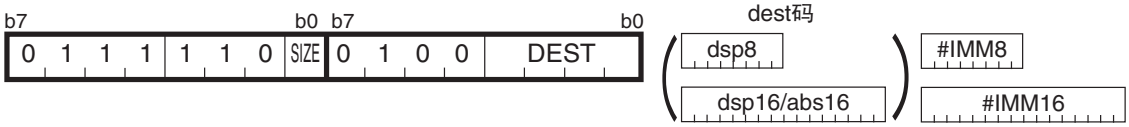
【字节数/周期数】

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。

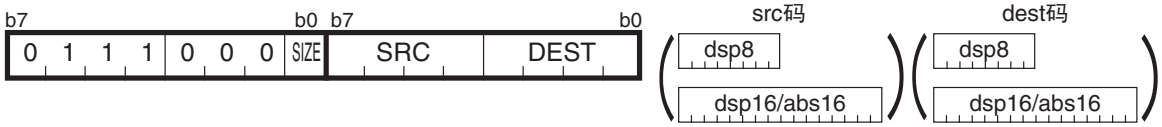
【字节数/周期数】

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。

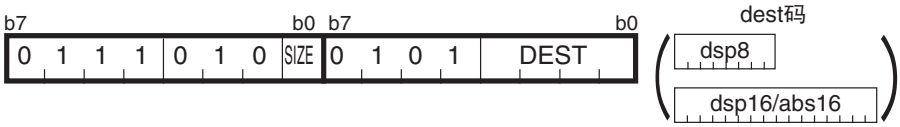
【字节数/周期数】

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
.B	0
.W	1

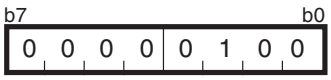
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

NOP

(1) NOP

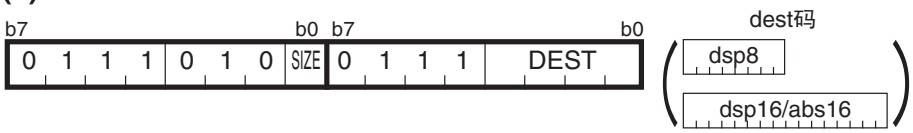


【字节数/周期数】

字节数/周期数	1/1

NOT

(1) NOT.size:G　　dest



.size	SIZE
.B	0
.W	1

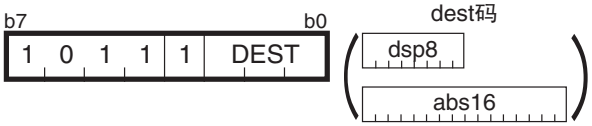
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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



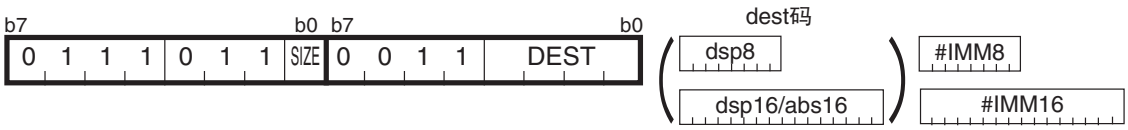
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

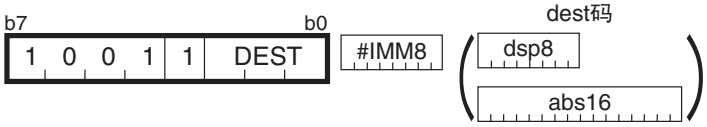
【字节数/周期数】

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



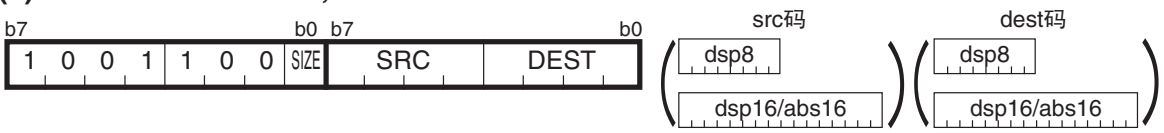
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

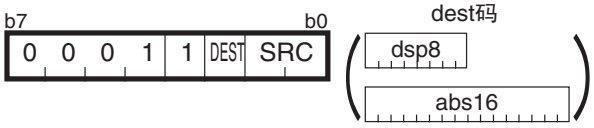
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

OR

(4) OR.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

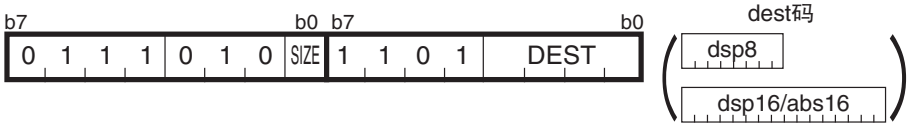
dest	DEST
R0L	0
R0H	1

【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

POP

(1) POP.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	2/3	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

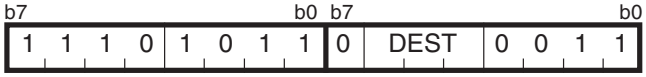
dest	DEST
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/3

POPC

(1) POPC　　dest



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

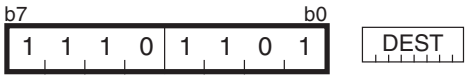
*1 记号“---”为不能选择。

【字节数/周期数】

字节数/周期数	2/3

POPM

(1) POPM　　dest



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”。

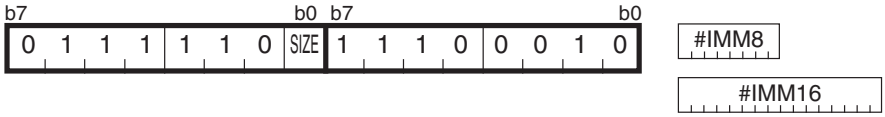
【字节数/周期数】

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

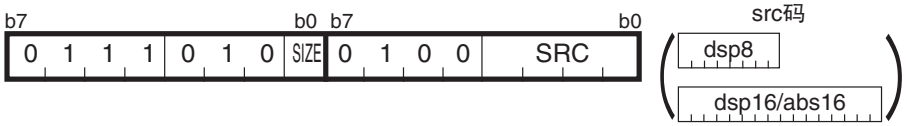
【字节数/周期数】

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	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	2/2	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

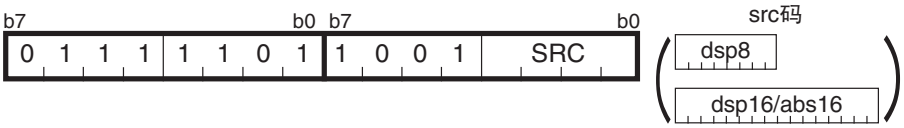
src	SRC
A0	0
A1	1

【字节数/周期数】

字节数/周期数	1/2

PUSHA

(1) PUSHA　　src



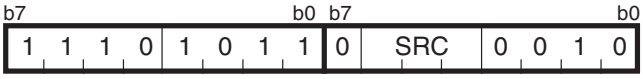
src		SRC
dsp:8[An]	dsp:8[A0]	1 0 0 0
	dsp:8[A1]	1 0 0 1
dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	dsp:8[FB]	1 0 1 1
dsp:16[An]	dsp:16[A0]	1 1 0 0
	dsp:16[A1]	1 1 0 1
dsp:16[SB]	dsp:16[SB]	1 1 1 0
abs16	abs16	1 1 1 1

【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

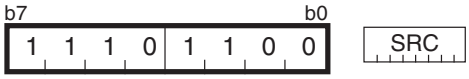
*1 记号“---”为不能选择。

【字节数/周期数】

字节数/周期数	2/2

PUSHM

(1) PUSHM　　src



src							
R0	R1	R2	R3	A0	A1	SB	FB
SRC*1							

*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”。

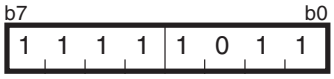
【字节数/周期数】

字节数/周期数	2/2×m

*2 m为保存的寄存器数。

REIT

(1) REIT

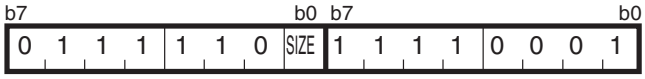


【字节数/周期数】

字节数/周期数	1/6

RMPA

(1) RMPA.size



.size	SIZE
.B	0
.W	1

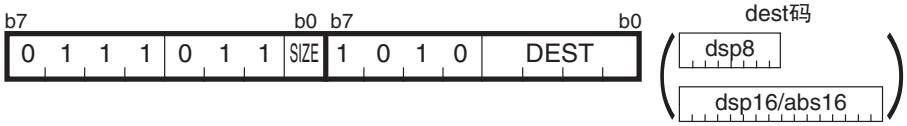
【字节数/周期数】

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

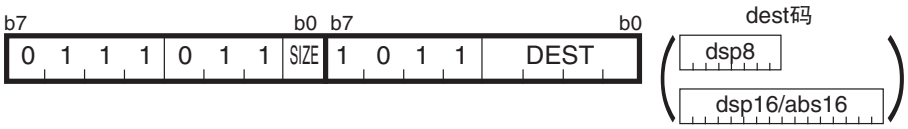
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

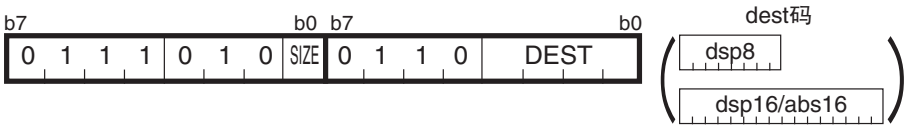
【字节数/周期数】

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。

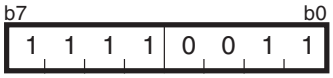
【字节数/周期数】

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

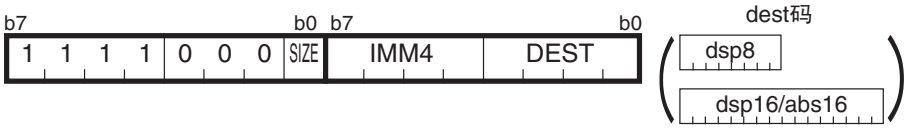
【字节数/周期数】

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	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

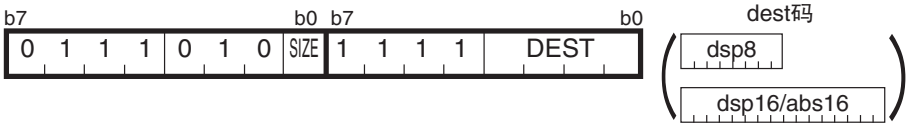
【字节数/周期数】

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。

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。

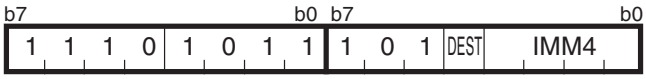
【字节数/周期数】

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

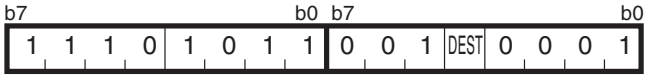
【字节数/周期数】

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

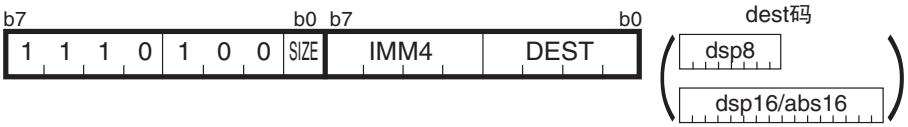
【字节数/周期数】

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

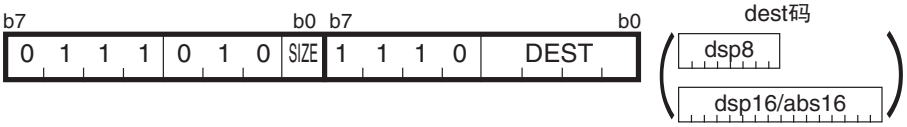
【字节数/周期数】

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
.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。

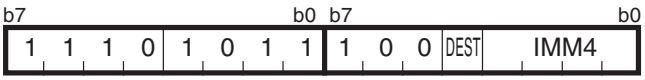
【字节数/周期数】

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。

SHL

(3) SHL.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

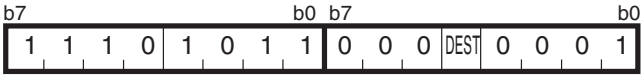
【字节数/周期数】

字节数/周期数	2/3+m

*2 m为移位的位数。

SHL

(4) SHL.L　　R1H, dest



dest	DEST
R2R0	0
R3R1	1

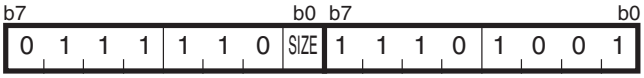
【字节数/周期数】

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

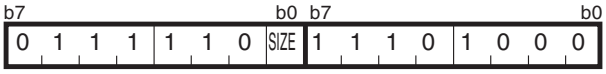
【字节数/周期数】

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

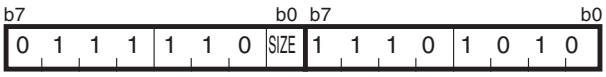
【字节数/周期数】

字节数/周期数	2/5+5×m

*1 m为传送次数。

SSTR

(1) SSTR.size



.size	SIZE
.B	0
.W	1

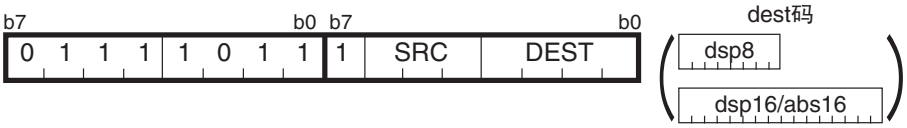
【字节数/周期数】

字节数/周期数	2/3+2×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

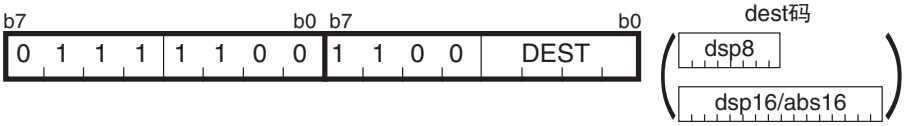
*1 记号“---”为不能选择。

【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

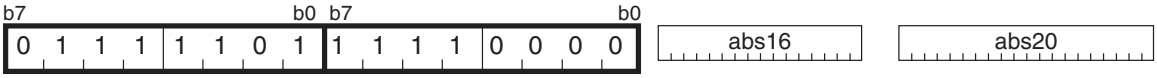
*1 记号“---”为不能选择。

【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



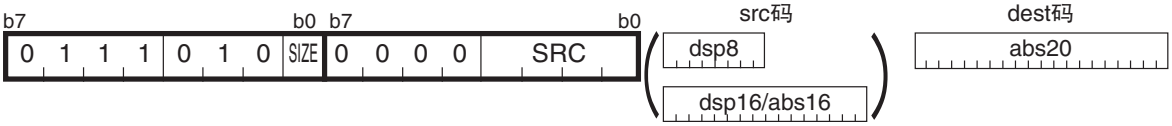
【字节数/周期数】

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

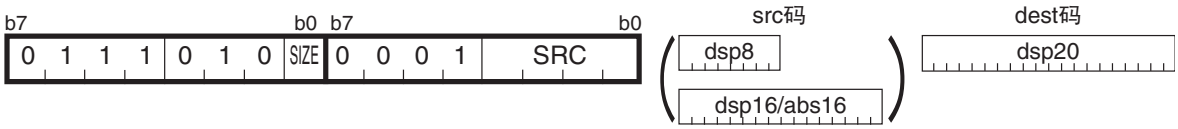
src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1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

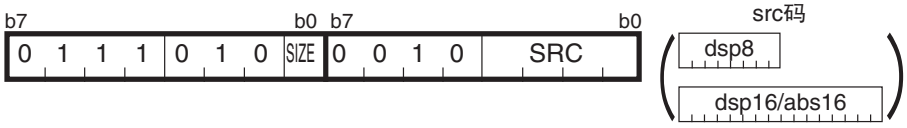
src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1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

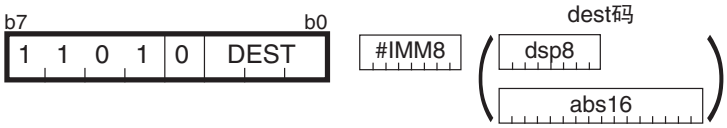
src		SRC	src		SRC
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]	1 1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



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

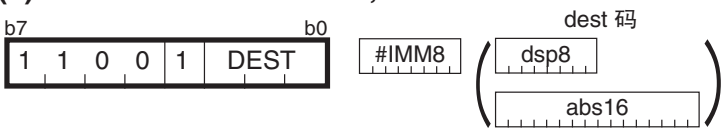
【字节数/周期数】

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



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。

STZX

(1) STZX　　#IMM81, #IMM82 , dest

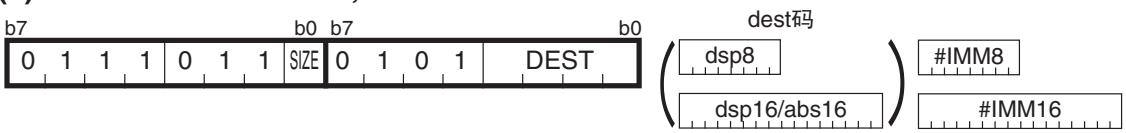
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【字节数/周期数】

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

SUB

(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

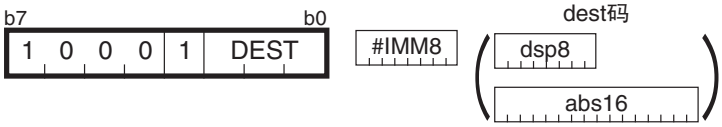
【字节数/周期数】

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



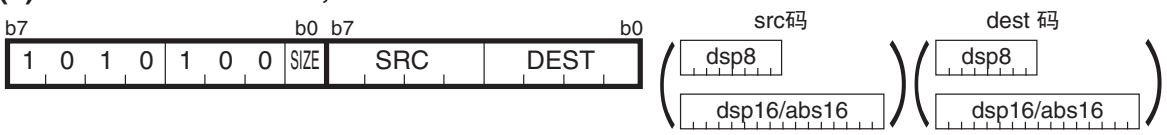
dest		DEST
Rn	R0H	0 1 1
	R0L	1 0 0
dsp:8[SB/FB]	dsp:8[SB]	1 0 1
	dsp:8[FB]	1 1 0
abs16	abs16	1 1 1

【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

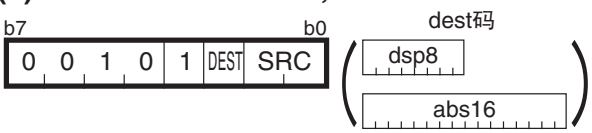
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

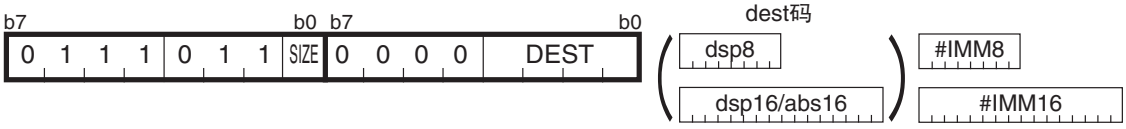
dest	DEST
R0L	0
R0H	1

【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

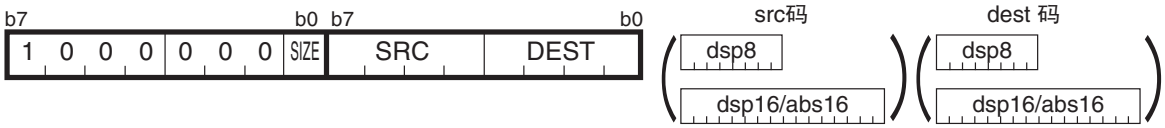
【字节数/周期数】

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

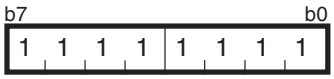
src/dest		SRC/DEST	src/dest		SRC/DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]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

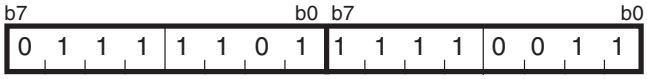


【字节数/周期数】

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

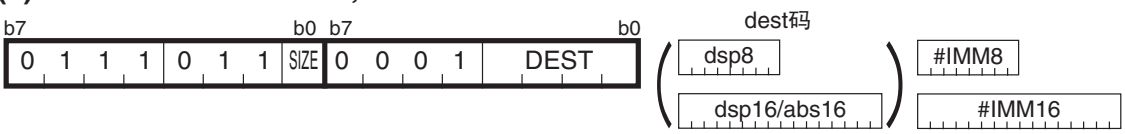
dest		DEST	dest		DEST
Rn	R0L/R0	0 0 0 0	dsp:8[An]	dsp:8[A0]	1 0 0 0
	R0H/R1	0 0 0 1		dsp:8[A1]	1 0 0 1
	R1L/R2	0 0 1 0	dsp:8[SB/FB]	dsp:8[SB]	1 0 1 0
	R1H/R3	0 0 1 1		dsp:8[FB]	1 0 1 1
An	A0	0 1 0 0	dsp:16[An]	dsp:16[A0]	1 1 0 0
	A1	0 1 0 1		dsp:16[A1]	1 1 0 1
[An]	[A0]	0 1 1 0	dsp:16[SB]	dsp:16[SB]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

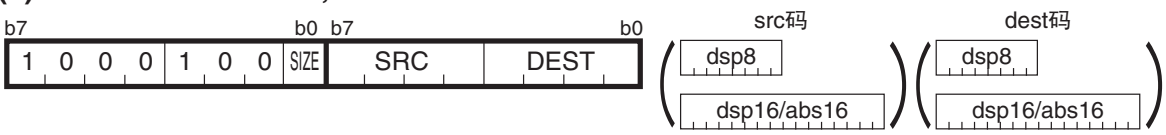
【字节数/周期数】

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。

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

如果接受中断请求，就转移到设定在中断向量表中的中断程序。在各中断向量表中必须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。

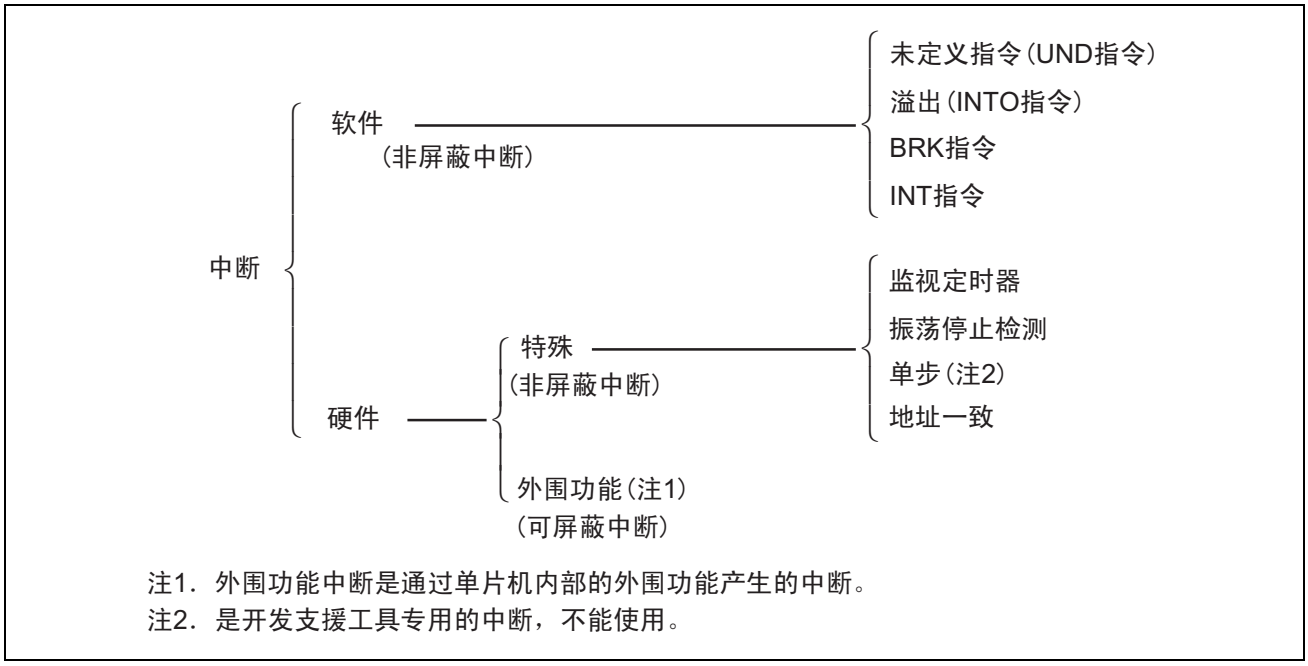


图5.1.1 中断分类

- 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

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)	备考
未定义指令	0FFDC16～0FFDF16	由UND指令中断
溢出	0FFE016～0FFE316	由INTO指令中断
BRK指令	0FFE416～0FFE716	在0FFE716地址的内容为FF16时，从可变向量表内的向量所指向的地址开始执行
地址一致	0FFE816～0FFEB16	有地址一致中断允许位
单步(注1)	0FFEC16～0FFEF16	禁止使用
监视定时器、振荡停止检测	0FFF016～0FFF316	
(保留)	0FFF416～0FFF716	
(保留)	0FFF816～0FFFB16	
复位	0FFFC16～0FFFF16	

注1. 是开发支援工具专用的中断，不能使用。

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。

●未定义指令中断

如果执行 UND 指令，就产生未定义指令中断。

●溢出中断

在 O 标志为“1”（运算结果溢出）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：

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

●BRK 中断

如果执行 BRK 指令，就产生 BRK 中断。

●INT 指令中断

如果执行INT指令，就产生INT指令中断。能用INT指令指定的软件中断号是0～63。由于软件中断号4～31分配给外围功能中断，因此能通过执行INT指令，执行和外围功能中断相同的中断程序。

对于软件中断号0～31，当执行指令时将U标志压栈，然后在将U标志置“0”（选择ISP）后，执行中断顺序。在从中断程序返回时，恢复被压栈的U标志。对于软件中断号32～63，当执行指令时U标志不变，使用当时选择的SP。

5.1.3　硬件中断

硬件中断有特殊中断和外围功能中断。

●特殊中断

特殊中断是非屏蔽中断。

(1) 监视定时器中断

它是由监视定时器产生的中断。必须在发生监视定时器中断后初始化监视定时器。监视定时器的详细内容请参照“硬件手册”。

(2) 振荡停止检测中断

它是由振荡停止检测功能产生的中断。振荡停止检测功能的详细内容请参照“硬件手册”。

(3) 单步中断

它是开发支援工具专用的中断，不能使用。

(4) 地址一致中断

在AIER寄存器的AIER0位和AIER1位中的任意一位为“1”（允许地址一致中断）时，在执行由对应的RMAD0～RMAD1寄存器指向的地址的指令前，产生地址一致中断。

●外围功能中断

外围功能中断是由单片机内部的外围功能产生的中断，是可屏蔽中断。

内部外围功能根据产品种类的不同而不同，各种的中断源也根据产品种类的不同而不同。有关外围功能的详细内容请参照“硬件手册”。

5.2　中断控制

说明如何允许或者禁止可屏蔽中断以及如何设定能接受的优先权。在此说明的内容不适用于非屏蔽中断。

通过FLG寄存器的I标志、IPL以及各中断控制寄存器的ILVL2～ILVL0位，允许或者禁止可屏蔽中断。另外，在各中断控制寄存器的IR位中表示有无中断请求。

中断控制寄存器的存储器分配和寄存器结构请参照“**硬件手册**”。

5.2.1　I 标志

I标志允许或者禁止可屏蔽中断。如果将I标志置“1”（允许），就允许可屏蔽中断；如果置“0”（禁止），就禁止所有可屏蔽中断。

在改变I标志时，改变的内容被反映到中断请求接受判断的时序如下：

- 在通过REIT指令改变时，从此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。

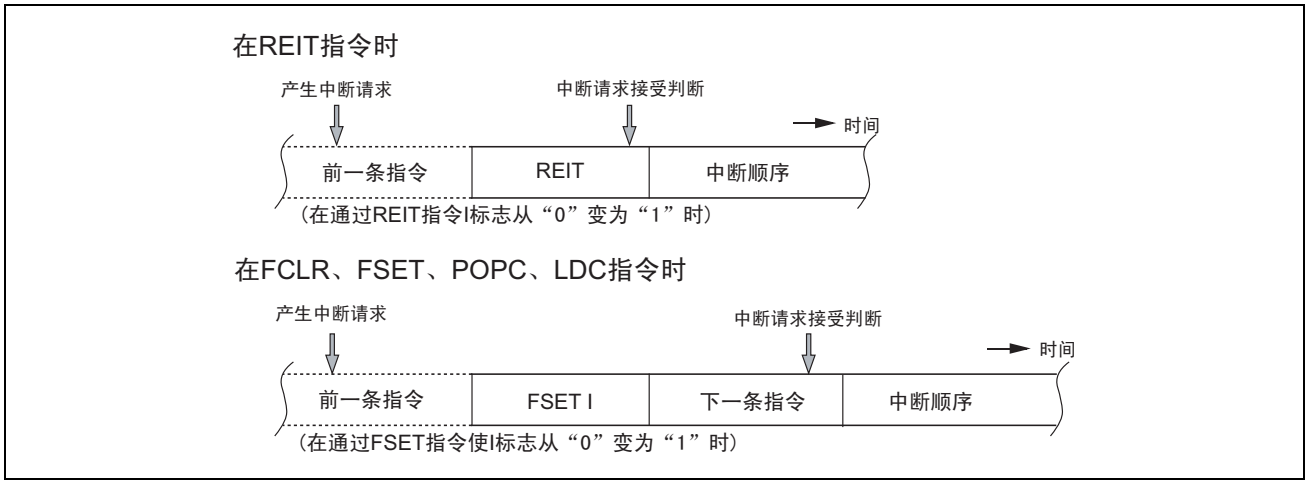


图5.2.1 在改变I标志时，对中断的反映时序

5.2.2　IR 位

如果产生中断请求，IR位就变为“1”（有中断请求）。在接受中断请求并转移到对应的中断向量后，IR位变为“0”（无中断请求）。

IR位能通过程序清“0”，不能写“1”。

5.2.3　ILVL2～ILVL0 位和 IPL

中断优先级能通过ILVL2～ILVL0位设定。

中断优先级的设定如表5.2.1所示，由IPL允许的中断优先级如表5.2.2所示。

接受中断请求的条件如下所示：

- I标志　　＝ 1
- IR位　　＝ 1
- 中断优先级　　＞ IPL

I标志、IR位、ILVL2～ILVL0位以及IPL各自独立互不影响。

表 10.3　中断优先级的设定

ILVL2～ILVL0	中断优先级	优先权
000_2	0级（中断禁止）	—
001_2	1级	低
010_2	2级	
011_2	3级	
100_2	4级	
101_2	5级	
110_2	6级	
111_2	7级	高

表 10.4　由 IPL 允许的中断优先级

IPL	允许的中断优先级
000_2	允许1级以上
001_2	允许2级以上
010_2	允许3级以上
011_2	允许4级以上
100_2	允许5级以上
101_2	允许6级以上
110_2	允许7级以上
111_2	禁止所有可屏蔽中断

在改变 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 IPL 时，在从 REIT 指令的最后时钟经过 2 个时钟后反映。
- 当通过 POPC、LDC、LDIPL 各指令改变 IPL 时，在从使用的指令的最后时钟经过 3 个时钟后反映。
- 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 3 个时钟后反映。

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，必须注意使用的指令。

・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，IR 位可能不变为“1”（有中断请求），中断被忽视。如果因此出现问题，必须使用以下指令改变寄存器：

对象指令…AND、OR、BCLR、BSET

・改变 IR 位

在将 IR 位置“0”（无中断请求）时，根据使用的指令，IR 位可能不变为“0”。必须用 MOV 指令将 IR 位置“0”。

(3) 在使用 I 标志禁止中断时，必须按照以下的程序例子设定 I 标志（程序例子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例子

```
INT_SWITCH1：
  FCLR    I              ；禁止中断
  AND.B   #00H，0056H    ；将 TXIC 寄存器置“0016”
  NOP                    ；
  NOP
  FSET    I              ；允许中断
```

例 2：通过虚读，使 FSET 指令等待的例子

```
INT_SWITCH2：
  FCLR    I              ；禁止中断
  AND.B   #00H，0056H    ；将 TXIC 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 标志的例子

```
INT_SWITCH3：
  PUSHC   FLG
  FCLR    I              ；禁止中断
  AND.B   #00H，0056H    ；将 TXIC 寄存器置“0016”
  POPC    FLG            ；允许中断
```

5.3　中断顺序

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顺序。但是，对于 SMOVB、SMOVF、SSTR 以及 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，转移到中断顺序。

中断顺序运行如下。中断顺序的执行时间如**图 5.3.1** 所示。

(1) 通过读 00000_{16} 地址，CPU 获得中断信息（中断号、中断请求级）。此后，该中断的 IR 位变为“0”（无中断请求）。
(2) 将中断顺序前的 FLG 寄存器保存到 CPU 内部的暂存器（注 1）。
(3) FLG 寄存器中的 I 标志、D 标志、U 标志变为：
I 标志为“0”（禁止中断）
D 标志为“0”（单步中断为中断禁止）
U 标志为“0”（指定 ISP）
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（注 1）压栈。
(5) 将 PC 压栈。
(6) 给 IPL 设定接受中断的中断优先级。
(7) 中断向量所设定的中断程序的起始地址存入 PC。

在中断顺序结束后，从中断程序的起始地址执行指令。

注 1．用户不能使用。

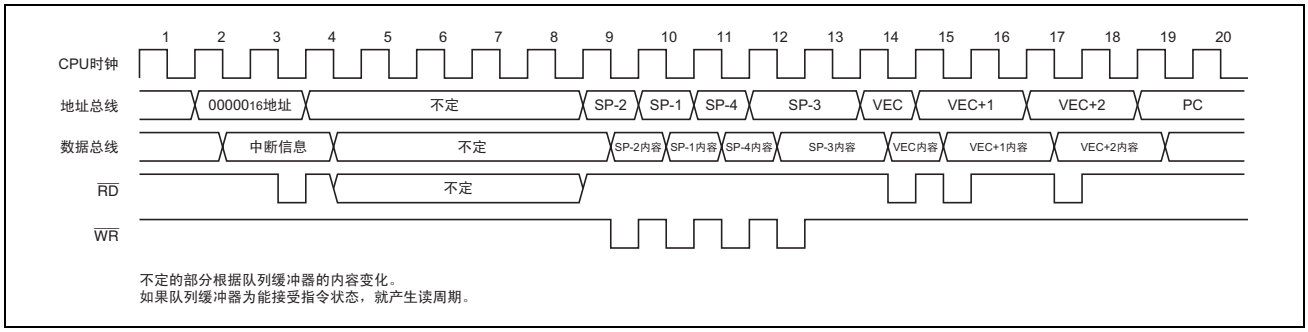
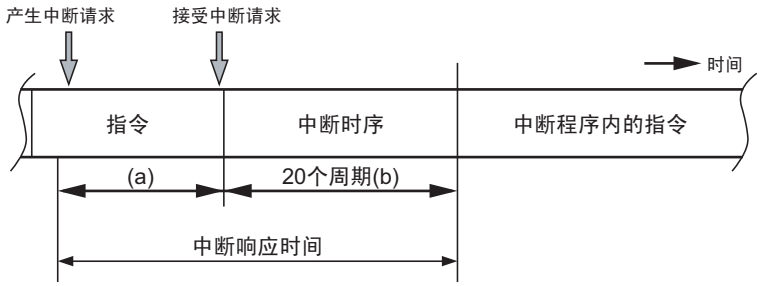


图5.3.1 中断顺序的执行时间

5.3.1　中断响应时间

中断响应时间如图 5.3.2 所示。中断响应时间是指从发生中断请求到执行中断程序内的最初指令为止的时间。该时间由从中断请求发生开始到当前正在执行的指令结束为止的时间（**图** 5.3.2 的(a)）和执行中断顺序的时间（20 个周期(b)）构成。



(a) 为从产生中断请求开始到此时正在执行的指令结束为止的时间，根据正在执行的指令不同而不同。
此时间最长的指令是DIVX指令，为30个周期(无等待、除数为寄存器时的周期数)。

(b) 地址一致中断，单步中断为21个周期。

图5.3.2 中断响应时间

5.3.2　接受中断请求时的 IPL 变化

如果接受可屏蔽中断的中断请求，就给 IPL 设定接受中断的中断优先级。

如果接受软件中断和特殊中断请求，就给 IPL 设定如**表** 5.3.1 所示的值。接受软件中断和特殊中断时的 IPL 值如**表** 5.3.1 所示。

表 5.3.1　接受软件中断和特殊中断时的 IPL 值

没有中断优先级的中断源	被设定的IPL值
监视定时器、振荡停止检测	7
软件、地址一致、单步	无变化

5.3.3　寄存器保存

在中断顺序，将 FLG 寄存器和 PC 压栈。

首先将 PC 的高 4 位、FLG 寄存器的高 4 位（IPL）和低 8 位压栈（全部为 16 位），然后将 PC 的低 16 位压栈。中断请求接受前后的堆栈状态如图 5.3.3 所示。

其它必要的寄存器必须通过程序在中断程序的最初保存。如果使用 PUSHM 指令，就能用 1 条指令保存除了 SP 以外的全部寄存器。

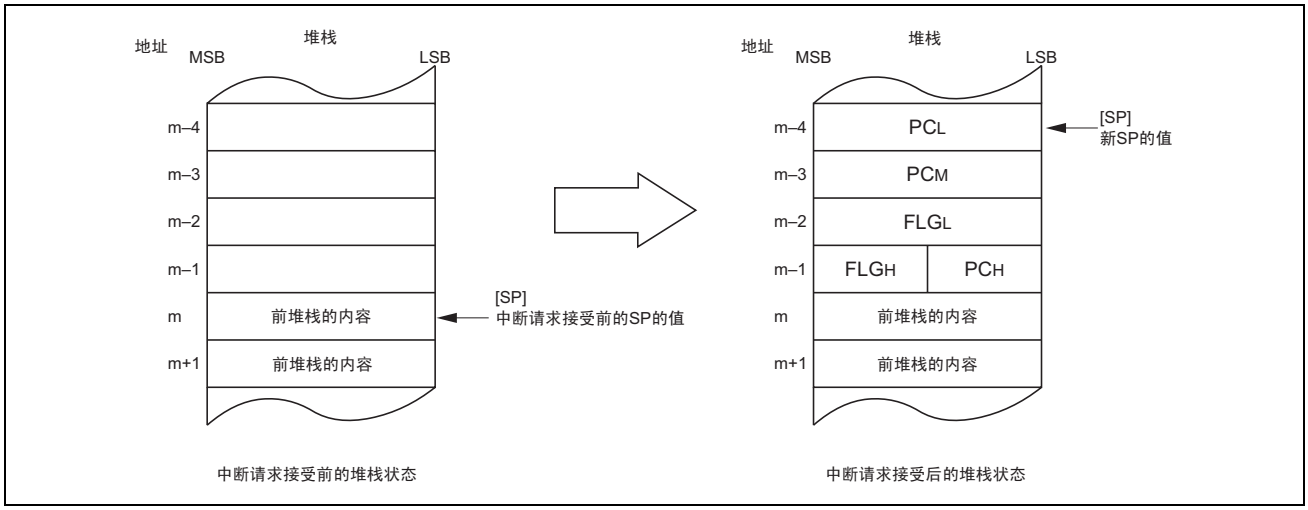


图5.3.3 中断请求接受前后的堆栈状态

在中断顺序进行的寄存器保存按 8 位分 4 次进行。寄存器保存运行如图 5.3.4 所示。

注 1.　在执行软件号 32～63 的 INT 指令时，为 U 标志指示的 SP；除此以外为 ISP。

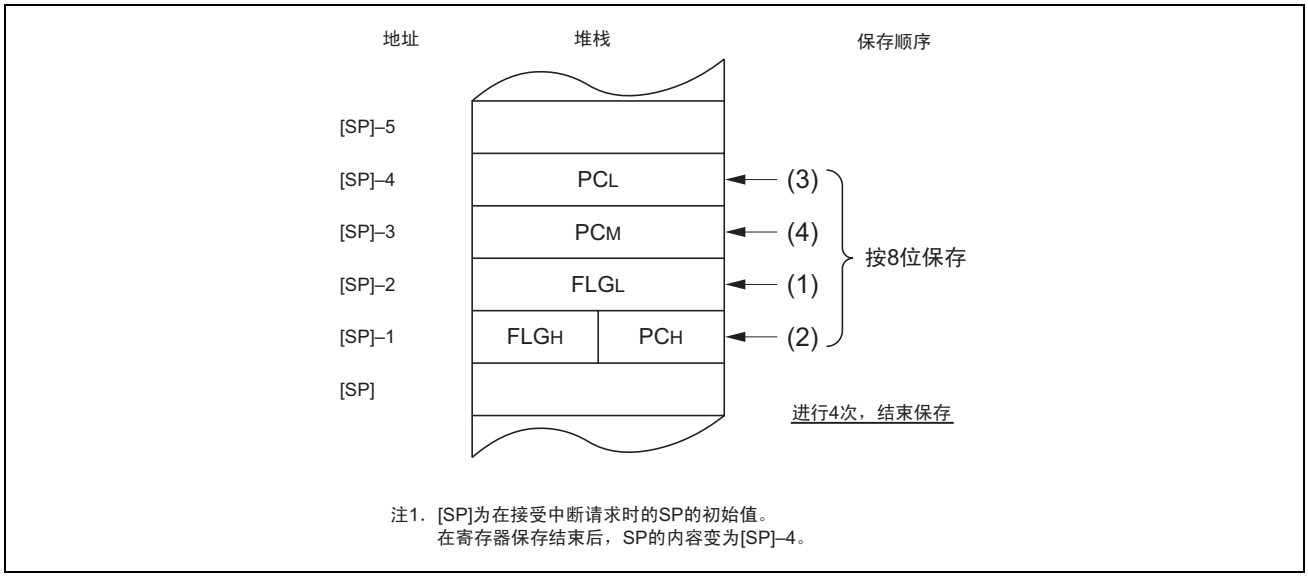


图5.3.4 寄存器保存的运行

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顺序前的 FLG 寄存器和 PC。然后，返回到在接受中断请求前执行的程序。

在中断程序内，通过程序保存的寄存器，必须在 REIT 指令执行前用 POPM 指令等恢复。

5.5　中断优先权

如果在 1 条指令执行中发生 2 个以上的中断请求，就接受优先权高的中断。

能通过 ILVL2～ILVL0 位任意选择可屏蔽中断（外围功能）的优先级。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权高的中断。

监视定时器中断等特殊中断的优先权由硬件设定。硬件中断的中断优先权如**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，就执行中断程序。

图5.5.1 硬件中断的中断优先权

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”
- 处理器中断优先级(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”，能接受高于处理器中断优先级(IPL)的优先权的中断请求。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　＞　被恢复的处理器中断优先级(IPL)

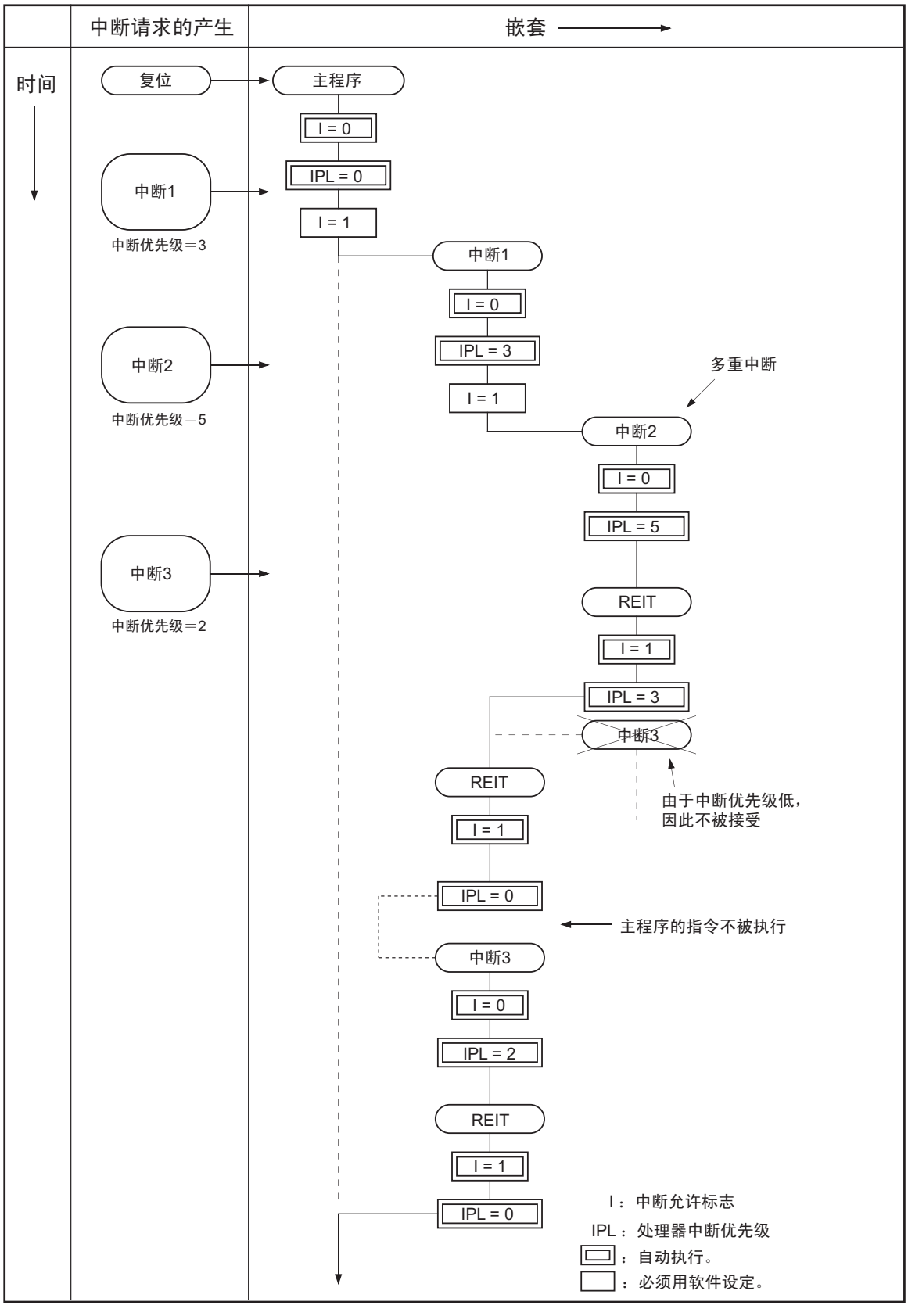

中断请求的产生
嵌套
时间
复位
主程序
I = 0
IPL = 0
I = 1
中断1
中断优先级＝3
中断1
I = 0
IPL = 3
I = 1
中断2
中断优先级＝5
多重中断
中断2
I = 0
IPL = 5
中断3
中断优先级＝2
REIT
I = 1
IPL = 3
中断3
由于中断优先级低，
因此不被接受
REIT
I = 1
IPL = 0
主程序的指令不被执行
中断3
I = 0
IPL = 2
REIT
I = 1
IPL = 0
I：中断允许标志
IPL：处理器中断优先级
：自动执行。
：必须用软件设定。

图5.6.1 多重中断

5.7　中断注意事项

5.7.1　读 0000016 地址

不能通过程序读 0000016 地址。在接受到可屏蔽中断的中断请求时，CPU 在中断序列中从 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 位变为“0”。

如果通过程序读 0000016 地址，在被允许的中断中，最高优先权的中断的 IR 位将变为“0”。因此，中断可能会被取消或者发生预料外的中断。

5.7.2　SP 的设定

必须在接受中断前给 SP 设定值。在复位后，SP 为“000016”。因此，如果在给 SP 设定值前接受中断，程序就会失控。

5.7.3　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，必须注意使用的指令。

・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，IR 位可能不变为“1”（有中断请求），中断被忽视。如果因此出现问题，必须使用以下指令改变寄存器：

对象指令…AND、OR、BCLR、BSET

・改变 IR 位

在将 IR 位置“0”（无中断请求）时，根据使用的指令，IR 位可能不变为“0”。必须用 MOV 指令将 IR 位置“0”。

(3) 在使用 I 标志禁止中断时，必须按照以下的程序例子设定 I 标志（中断控制寄存器的更改程序例子请参照(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例子

```
INT_SWITCH1：
  FCLR    I                ；禁止中断
  AND.B   #00H，0056H      ；将 TXIC 寄存器置“0016”
  NOP                      ；
  NOP
  FSET    I                ；允许中断
```

例 2：通过虚读，使 FSET 指令等待的例子

```
INT_SWITCH2：
  FCLR    I                ；禁止中断
  AND.B   #00H，0056H      ；将 TXIC 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 标志的例子

```
INT_SWITCH3：
  PUSHC   FLG
  FCLR    I                ；禁止中断
  AND.B   #00H，0056H      ；将 TXIC 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

R8C/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，对存储器以无软件等待读写 8 位数据时的周期数。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：

- 当从存在软件等待周期的区域读取指令码时

  读周期数将增加等待周期数的份。

■对存在软件等待周期的区域读写数据。

周期数将增加等待周期数的份。

■对 SFR 或者内部存储器读写 16 位数据。

对于 1 个数据的读写，将进行 2 次读写操作。因此，对于每个数据的读写，周期数将增加 1 个周期。

另外，在预取指令和数据存取同时发生时，数据存取被优先。

如果指令队列缓冲器内存在 3 字节以上的指令码，就认为指令队列缓冲器不为空状态，不预取指令。

开始读指令的情况（无软件等待）如图 6.1.1 所示。

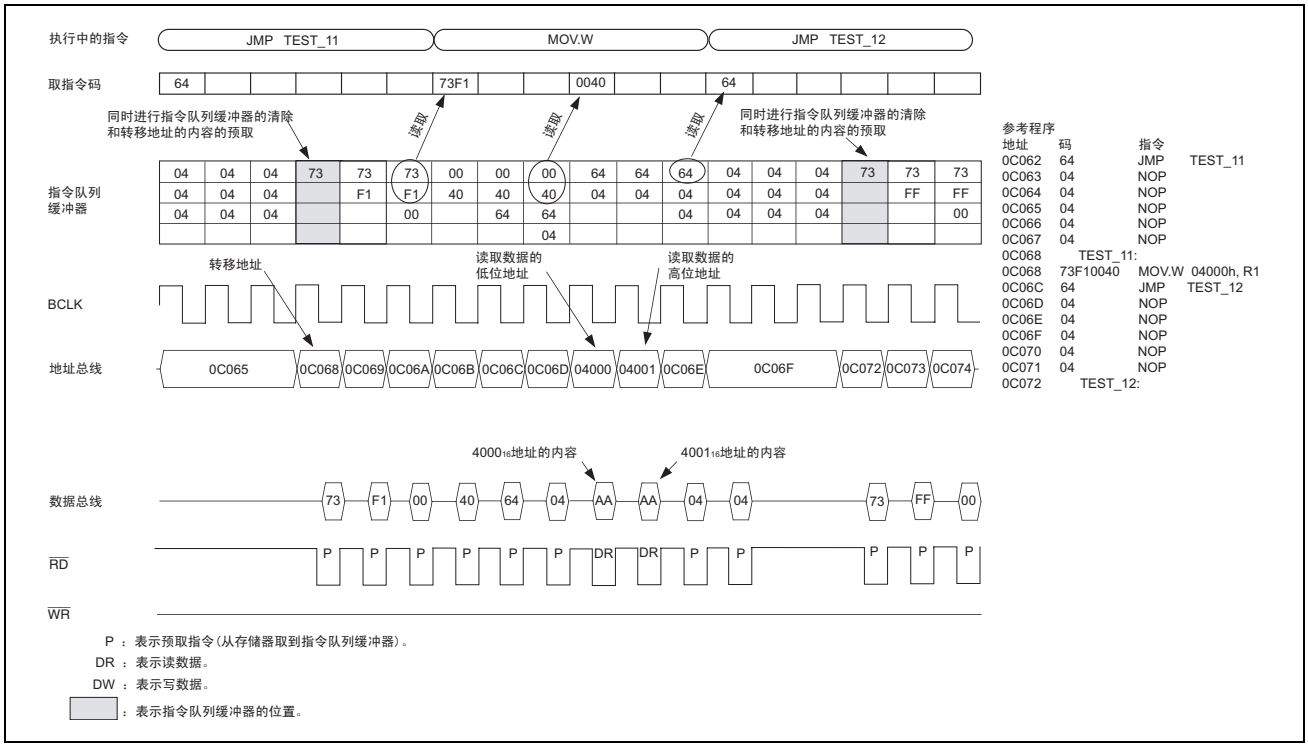


图6.1.1 开始读指令的情况（无软件等待）

Q&A

为了最大限度地正确使用 R8C/Tiny 系列，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字节表示操作码部，低位2字节为寻址方式部表现bit,base:16。低位2字节和bit,base:16的关系如下：

低位2字节 = base:16×8 + bit

例如，在BSET 2,0AH（将000A16地址的位2置1）的情况下，低位2字节为A×8 + 2=52H。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字节为8×8 + 18=52H，成为和BSET 2,AH相同的指令码。

base:16×8 +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。 表示数据的最低位的位。	MSB
MSB	Most Significant Bit的略称。 表示数据的最高位的位。	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R8C/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
PC_H	程序计数器的高位字节
PC_{ML}	程序计数器的中位字节和低位字节
FLG_H	标志寄存器的高 4 位
FLG_L	标志寄存器的低 4 位

索引

修订记录	R 8C/Tiny 系列 软件手册

Rev.	发行日	修订内容	
		页	修订处
1.00	2004.07.16	—	初版发行

瑞萨 16 位单片机软件手册
R8C/Tiny 系列

发行年月日　2004 年 7 月 16 日 Rev.1.00
发　　行　Sales Strategic Planning Div.
Renesas Technology Corp.
株式会社瑞萨科技
营业企画综合部
编　　辑　Technical Documentation & Information Department
Renesas Kodaira Semiconductor Co., Ltd.
株式会社瑞萨小平半导体技术文献部

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

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, United Kingdom
Tel: <44> (1628) 585 100, Fax: <44> (1628) 585 900

Renesas Technology Europe GmbH
Dornacher Str. 3, D-85622 Feldkirchen, Germany
Tel: <49> (89) 380 70 0, Fax: <49> (89) 929 30 11

Renesas Technology Hong Kong Ltd.
7/F., North Tower, World Finance Centre, Harbour City, Canton Road, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2375-6836

Renesas Technology Taiwan Co., Ltd.
FL 10, #99, Fu-Hsing N. Rd., Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology (Shanghai) Co., Ltd.
26/F., Ruijin Building, No.205 Maoming Road (S), Shanghai 200020, China
Tel: <86> (21) 6472-1001, Fax: <86> (21) 6415-2952

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

Colophon 1.0

R8C/Tiny 系列

瑞萨电子株式会社

RCJ09B0006-0100z