

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/30 (2006.01)



[12] 发明专利说明书

专利号 ZL 03138541.9

[45] 授权公告日 2006 年 3 月 22 日

[11] 授权公告号 CN 1246772C

[22] 申请日 2003.6.3 [21] 申请号 03138541.9

[30] 优先权

[32] 2002. 6. 3 [33] JP [31] 161381/2002

[71] 专利权人 松下电器产业株式会社

地址 日本大阪府

[72] 发明人 田中哲也 冈林羽月 瓶子岳人

小川一 古贺义宏 黑田学

铃木正人 清原督三 田中健

西田英志 宫阪修二

审查员 郑 红

[74] 专利代理机构 永新专利商标代理有限公司

代理人 黄剑峰

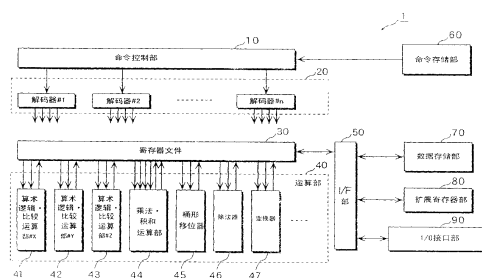
权利要求书 8 页 说明书 71 页 附图 500 页

[54] 发明名称

处理器

[57] 摘要

提供一种执行对作为 SIMD 运算对象的操作数位置的约束少、弹性高的 SIMD 运算的处理器。具备解码部(20)和运算部(40)等,若解码部(20)译码命令 [vxaddh Rc, Ra, Rb], 则算术逻辑、比较运算器(41)等(i)将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相加,将结果存储在寄存器 Rc 的上位 16 位中,同时,(ii)将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相加,将结果存储在寄存器 Rc 的下位 16 位中。



1、一种处理器，执行由1个指令来运算多个数据的SIMD型指令，其特征在于：具备

译码单元，译码指令；和

执行单元，根据译码单元的译码结果，执行指令，

上述执行单元在通过上述译码单元译码SIMD型指令，并且该SIMD型指令包含指定运算种类的指令代码、对由大于等于2的n个数据排列构成的第1数据群进行指定的第1操作数、和对由n个数据排列构成的第2数据群进行指定的第2操作数的情况下，对于由属于上述第1数据群的1个数据和属于上述第2数据群的1个数据而组成1组的n组，进行由上述指令代码指定的运算，

上述n组的至少1组由构成上述第1数据群的n个数据排列中的第i个数据和构成上述第2数据群的n个数据排列中的第j个数据构成，其中， $j \neq i$ ，

上述指令代码指定的运算种类中包含加法、减法、乘法、积和及积差。

2、根据权利要求1所述的处理器，其特征在于：

上述n为2，

上述第1数据群由第1个数据和第2个数据构成，

上述第2数据群由第1个数据和第2个数据构成，

上述执行单元对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组进行上述运算，并对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算。

3、根据权利要求2所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含对用于存放运算结果的第3数据进行指定的第3操

作数，

上述执行单元将对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组进行上述运算所得结果的低位部分、和对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算所得结果的低位部分，存放在上述第3数据中。

4、根据权利要求2所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含指定用于存放运算结果的第3数据的第3操作数，

上述执行单元将对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组进行上述运算所得结果的高位部分、和对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算所得结果的高位部分存放在上述第3数据中。

5、根据权利要求2所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含指定存放运算结果用第3数据的第3操作数，

上述执行单元将对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组进行上述运算所得的结果、和对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算所得的结果中的一个，存放在上述第3数据中。

6、根据权利要求1所述的处理器，其特征在于：

上述 n 大于 4，

上述 i 和上述 j 满足

$$j=n-i+1。$$

7、根据权利要求1所述的处理器，其特征在于：

上述 n 大于 4，

上述 i 和上述 j 满足

$$j=i-(-1)^{(i \bmod 2)}, \text{ ` 表示取幂, mod 表示取余数。}$$

8、根据权利要求1所述的处理器，其特征在于：

上述 n 大于 4，

上述 i 和上述 j 满足

$j = n - i + 1 + (-1)^{(i \bmod 2)}$ ， \wedge 表示取幂， \bmod 表示取余数。

9、根据权利要求1所述的处理器，其特征在于：

上述 n 为 4，

上述第 1 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成，

上述第 2 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成，

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 4 个数据组成的组、上述第 1 数据群的第 2 个数据和第 2 数据群的第 3 个数据组成的组、上述第 1 数据群的第 3 个数据和第 2 数据群的第 2 个数据组成的组、及上述第 1 数据群的第 4 个数据和第 2 数据群的第 1 个数据组成的组进行上述运算。

10、根据权利要求1所述的处理器，其特征在于：

上述 n 为 4，

上述第 1 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成，

上述第 2 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成，

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 2 个数据组成的组、上述第 1 数据群的第 2 个数据和第 2 数据群的第 1 个数据组成的组、上述第 1 数据群的第 3 个数据和第 2 数据群的第 4 个数据组成的组、及上述第 1 数据群的第 4 个数据和第 2 数据群的第 3 个数据组成的组进行上述运算。

11、根据权利要求1所述的处理器，其特征在于：

上述 n 为 4,

上述第 1 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成,

上述第 2 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成,

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 3 个数据组成的组、上述第 1 数据群的第 2 个数据和第 2 数据群的第 4 个数据组成的组、上述第 1 数据群的第 3 个数据和第 2 数据群的第 1 个数据组成的组、及上述第 1 数据群的第 4 个数据和第 2 数据群的第 2 个数据组成的组进行上述运算。

12、根据权利要求 9-11 之一所述的处理器, 其特征在于:

上述指令代码指定的运算种类是乘法、积和或积差,

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数,

上述执行单元将对上述 4 个组进行上述运算所得的各结果的低位部分存放在上述第 3 数据中。

13、根据权利要求 9-11 之一所述的处理器, 其特征在于:

上述指令代码指定的运算种类是乘法、积和或积差,

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数,

上述执行单元将对上述 4 个组进行上述运算所得的各结果的高位部分存放在上述第 3 数据中。

14、根据权利要求 9-11 之一所述的处理器, 其特征在于:

上述指令代码指定的运算种类是乘法、积和或积差,

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数,

上述执行单元将对上述 4 个组进行上述运算所得 4 个结果中的两个存放在上述第 3 数据中。

15、根据权利要求 1 所述的处理器, 其特征在于:

上述执行单元对上述 $i=1, 2, \dots, n$ 、上述 j =一定值的情况下的

上述第 1 数据群的第 i 个数据和第 2 数据群的第 j 个数据组成一组从而构成的 n 组分别进行上述运算。

16、根据权利要求 1 所述的处理器，其特征在于：

上述 n 为 2，

上述第 1 数据群由第 1 个数据和第 2 个数据构成，

上述第 2 数据群由第 1 个数据和第 2 个数据构成，

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 1 个数据组成的组进行上述运算，并对上述第 1 数据群的第 2 个数据和第 2 数据群的第 1 个数据组成的组进行上述运算。

17、根据权利要求 1 所述的处理器，其特征在于：

上述 n 为 2，

上述第 1 数据群由第 1 个数据和第 2 个数据构成，

上述第 2 数据群由第 1 个数据和第 2 个数据构成，

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 2 个数据组成的组进行上述运算，并对上述第 1 数据群的第 2 个数据和第 2 数据群的第 2 个数据组成的组进行上述运算。

18、根据权利要求 1 所述的处理器，其特征在于：

上述 n 为 2，

上述第 1 数据群由第 1 个数据和第 2 个数据构成，

上述第 2 数据群由第 1 个数据和第 2 个数据构成，

上述指令包括第 1 指令和第 2 指令，

上述执行单元在由上述译码单元译码第 1 指令的情况下，对上述第 1 数据群的第 1 个数据和第 2 数据群的第 1 个数据组成的组进行上述运算，并对上述第 1 数据群的第 2 个数据和第 2 数据群的第 1 个数据组成的组进行上述运算，

在由上述译码单元译码第 2 指令的情况下，对上述第 1 数据群的第 1 个数据和第 2 数据群的第 2 个数据组成的组进行上述运算，并对

上述第1数据群的第2个数据和第2数据群的第2个数据组成的组进行上述运算。

19、根据权利要求16-18之一所述的处理器，其特征在于：
上述指令代码指定的运算种类是乘法、积和或积差，
上述指令包含指定存放运算结果用第3数据的第3操作数，
上述执行单元将对上述两个组进行上述运算所得的各结果的低位部分存放在上述第3数据中。

20、根据权利要求16-18之一所述的处理器，其特征在于：
上述指令代码指定的运算种类是乘法、积和或积差，
上述指令包含指定存放运算结果用第3数据的第3操作数，
上述执行单元将对上述两个组进行上述运算所得的各结果的高位部分存放在上述第3数据中。

21、根据权利要求16-18之一所述的处理器，其特征在于：
上述指令代码指定的运算种类是乘法、积和或积差，
上述指令包含指定存放运算结果用第3数据的第3操作数，
上述执行单元将对上述两个组进行上述运算所得的两个结果之一存放在上述第3数据中。

22、根据权利要求1所述的处理器，其特征在于：
上述n为4，
上述第1数据群由第1个、第2个、第3个、第4个数据排列构成，
上述第2数据群由第1个、第2个、第3个、第4个数据排列构成，

上述执行单元对上述第1数据群的第1个数据和第2数据群的第1个数据组成的组、上述第1数据群的第2个数据和第2数据群的第1个数据组成的组、上述第1数据群的第3个数据和第2数据群的第1个数据组成的组、及上述第1数据群的第4个数据和第2数据群的

第1个数据组成的组进行上述运算。

23、根据权利要求1所述的处理器，其特征在于：

上述n为4，

上述第1数据群由第1个、第2个、第3个、第4个数据排列构成，

上述第2数据群由第1个、第2个、第3个、第4个数据排列构成，

上述执行单元对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组、上述第1数据群的第2个数据和第2数据群的第2个数据组成的组、上述第1数据群的第3个数据和第2数据群的第2个数据组成的组、及上述第1数据群的第4个数据和第2数据群的第2个数据组成的组进行上述运算。

24、根据权利要求1所述的处理器，其特征在于：

上述n为4，

上述第1数据群由第1个、第2个、第3个、第4个数据排列构成，

上述第2数据群由第1个、第2个、第3个、第4个数据排列构成，

上述执行单元对上述第1数据群的第1个数据和第2数据群的第3个数据组成的组、上述第1数据群的第2个数据和第2数据群的第3个数据组成的组、上述第1数据群的第3个数据和第2数据群的第3个数据组成的组、及上述第1数据群的第4个数据和第2数据群的第3个数据组成的组进行上述运算。

25、根据权利要求1所述的处理器，其特征在于：

上述n为4，

上述第1数据群由第1个、第2个、第3个、第4个数据排列构成，

上述第 2 数据群由第 1 个、第 2 个、第 3 个、第 4 个数据排列构成，

上述执行单元对上述第 1 数据群的第 1 个数据和第 2 数据群的第 4 个数据组成的组、上述第 1 数据群的第 2 个数据和第 2 数据群的第 4 个数据组成的组、上述第 1 数据群的第 3 个数据和第 2 数据群的第 4 个数据组成的组、及上述第 1 数据群的第 4 个数据和第 2 数据群的第 4 个数据组成的组进行上述运算。

26、根据权利要求 22-25 之一所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数，

上述执行单元将对上述 4 个组进行上述运算所得的各结果的低位部分存放在上述第 3 数据中。

27、根据权利要求 22-25 之一所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数，

上述执行单元将对上述 4 个组进行上述运算所得的各结果的高位部分存放在上述第 3 数据中。

28、根据权利要求 22-25 之一所述的处理器，其特征在于：

上述指令代码指定的运算种类是乘法、积和或积差，

上述指令包含指定存放运算结果用第 3 数据的第 3 操作数，

上述执行单元将对上述 4 个组进行上述运算所得 4 个结果中的两个存放在上述第 3 数据中。

处理器

技术领域

本发明涉及 DSP 或 CPU 等处理器，尤其是涉及执行 SIMD 型命令的处理器。

背景技术

以前，作为支持 SIMD (Single Instruction Multiple Data: 单命令多数据流) 型命令的处理器，有美国英特尔公司的 Pentium(R) / Pentium (R) III/ Pentium (R) 4 的 MMX/SSE/SSE2 等。

例如，若是 MMX，则以 64 比特长的 MMX 寄存器中存储的最大 8 个整数为对象，可由 1 个命令来执行相同的操作。

但是，在上述现有处理器中，存在对作为 SIMD 运算对象的操作数 (operand) 位置的约束多的问题。

例如，在第 1 寄存器的上位数位和下位数位中分别存储数值 A 及 B、在第 2 寄存器的上位数位和下位数位中分别存储数值 C 及 D 的状态下，在由现有处理器执行将第 1 寄存器和第 2 寄存器设为操作数的 SIMD 型加法命令的情况下，得到的加法值为 A+C 及 B+D。即，得到存储在两寄存器的上位数位中的数据彼此的加法值和存储在两寄存器的下位数位中的数据彼此的加法值，由数据在寄存器中的存储位置来唯一确定运算对象。

因此，例如在对上述第 1 寄存器及第 2 寄存器求出 A+D、及 B+C 的加法值的情况下，需要在调换存储在一方寄存器中的上位数位的数据与下位数位的数据的存储位置后，执行 SIMD 型的加法命令，或不使用 SIMD 型加法命令，而执行两次通常的 SISD (Single Instruction Single Data: 单命令单数据流) 型加法命令。

但是，随着近年来的通信数字化，在需要傅立叶变换或滤波处理等数字信号处理的图像处理或声音处理领域中，必需对多个数据实施相同的运算处理，但此时，例如必需对位于距数据排列中的中心位置左右对称的数据实施相同运算等处理的情况多。在这种情况下，需要反序排列作为运算对象的两种操作数，例如，将存储在两个寄存器中一方高位数位中的数据与存储在另一方低位数位中的数据作为对象，进行运算。

但是，在现有处理器的 SIMD 运算中，如上所述，因为必需按相同顺序排列作为运算对象的操作数，所以必需操作数的重新排列等，存在数字信号处理需要很多时间的问题。

发明内容

因此，鉴于这种状况，本发明的目的在于提供一种处理器，执行对作为 SIMD 运算对象的操作数位置的约束少、适应性强的 SIMD 运算。具体而言，提供一种例如适于高速执行数字信号处理的多媒体用途的处理器。

为了实现上述目的，本发明的处理器执行由 1 个指令来运算多个数据的 SIMD 型指令，其特征在于：具备译码单元，译码指令；和执行单元，根据译码单元的译码结果，执行指令，上述执行单元在上述译码单元译码 SIMD 型指令的情况下，其中，SIMD 型指令包含指定运算种类的指令代码、指定由 n (≥ 2) 个数据排列构成的第 1 数据群的第 1 操作数、和指定由 n 个数据排列构成的第 2 数据群的第 2 操作数，对以属于上述第 1 数据群的 1 个数据和属于上述第 2 数据群的 1 个数据为 1 组的 n 组进行由上述指令代码指定的运算，上述 n 组的至少 1 组由构成上述第 1 数据群的 n 个数据排列中的第 i 个数据和构成上述第 2 数据群的 n 个数据排列中的第 j ($\neq i$) 个数据构成。例如，上述 n 为 2，上述第 1 数据群由第 1 个数据和第 2 个数据构成，上述第 2 数据群由第 1 个数据和第 2 个数据构成，上述执行单元在对上述第 1 数据群的第 1 个数据和第 2 数据群的第 2 个数据组成

的组进行上述运算的同时,对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算。即,运算处于交叉位置关系的各个数据。

这里,在运算种类是乘法、积和或积差的情况下,也可仅输出所得各运算结果的下位部分,或仅输出上位部分,或仅输出部分运算结果。

另外,例如,也可是上述 n 为2,上述第1数据群由第1个数据和第2个数据构成,上述第2数据群由第1个数据和第2个数据构成,上述执行单元在对上述第1数据群的第1个数据和第2数据群的第1个数据组成的组进行上述运算的同时,对上述第1数据群的第2个数据和第2数据群的第1个数据组成的组进行上述运算,上述执行单元又在对上述第1数据群的第1个数据和第2数据群的第2个数据组成的组进行上述运算的同时,对上述第1数据群的第2个数据和第2数据群的第2个数据组成的组进行上述运算。即,也可执行将一方的数据排列中的对象固定为特定数据的SIMD型命令。

另外,也可执行移位执行SIMD型运算所得结果后输出、或累计 n 排列(矢量)数据后算出标量值、或符号扩展 n 个数据每个的SIMD型命令。

另外,本发明不仅可实现为执行这种特征命令的处理器,也可实现为对多个数据等的运算处理方法,或可实现为包含特征命令的程序。另外,不用说,这种程序可经CD-ROM等记录媒体或因特网等传输媒体进行流通。

附图说明

图1是本发明处理器的示意框图。

图2是该处理器的算术逻辑、比较运算器的示意图。

图3是表示该处理器的桶形(barrel)シタ构成的框图。

图4是表示该处理器的变换器构成的框图。

- 图 5 是表示该处理器的除法器构成的框图。
- 图 6 是表示该处理器的乘法、积和运算器构成的框图。
- 图 7 是表示该处理器的命令控制部构成的框图。
- 图 8 是表示该处理器的通用寄存器 (R0-R31) 构造的图。
- 图 9 是表示该处理器的连接寄存器 (LR) 构造的图。
- 图 10 是表示该处理器的分支寄存器 (TAR) 构造的图。
- 图 11 是表示该处理器的程序状态寄存器 (PSR) 构造的图。
- 图 12 是表示该处理器的条件标志寄存器 (CFR) 构造的图。
- 图 13 是表示该处理器的累加器 (M0、M1) 构造的图。
- 图 14 是表示该处理器的程序计数器 (PC) 构造的图。
- 图 15 是表示该处理器的 PC 保存用寄存器 (IPC) 构造的图。
- 图 16 是表示该处理器的 PSR 保存用寄存器 (IPSR) 构造的图。
- 图 17 是表示该处理器的管道 (pipe-line) 动作的时间图。
- 图 18 是表示该处理器执行命令时各管道动作的时间图。
- 图 19 是表示该处理器的排列动作的图。
- 图 20 是表示该处理器执行的命令格式的图。
- 图 21 是说明属于分类 [ALUadd (加法) 类] 的命令的图。
- 图 22 是说明属于分类 [ALUsub (减法) 类] 的命令的图。
- 图 23 是说明属于分类 [ALUlogic (逻辑运算) 类] 的命令的图。
- 图 24 是说明属于分类 [CMP (比较运算) 类] 的命令的图。
- 图 25 是说明属于分类 [mul (乘法) 类] 的命令的图。
- 图 26 是说明属于分类 [mac (积和运算) 类] 的命令的图。
- 图 27 是说明属于分类 [msu (积差运算) 类] 的命令的图。
- 图 28 是说明属于分类 [MEMld (存储器读取) 类] 的命令的图。
- 图 29 是说明属于分类 [MEMstore (存储器写出) 类] 的命令的图。
- 图 30 是说明属于分类 [BRA (分支) 类] 的命令的图。
- 图 31 是说明属于分类 [BSasl (算术桶形移位 (barrel shift))

类]的命令的图。

图 32 是说明属于分类[BS1sr (逻辑桶形移位) 类]的命令的图。

图 33 是说明属于分类[CNVvaln (算术变换) 类]的命令的图。

图 34 是说明属于分类[CNV (一般变换) 类]的命令的图。

图 35 是说明属于分类[SATv1pk (饱和处理) 类]的命令的图。

图 36 是说明属于分类[ETC (其他) 类]的命令的图。

图 37 是说明命令[1d Rb, (Ra, D10)]的图。

图 38 是说明命令[1d Rb3, (Ra3, D5)]的图。

图 39 是说明命令[1d Rb, (GP, D13)]的图。

图 40 是说明命令[1d Rb2, (GP, D6)]的图。

图 41 是说明命令[1d Rb2, (GP)]的图。

图 42 是说明命令[1d Rb, (SP, D13)]的图。

图 43 是说明命令[1d Rb2, (SP, D6)]的图。

图 44 是说明命令[1d Rb2, (SP)]的图。

图 45 是说明命令[1d Rb, (Ra+) I10]的图。

图 46 是说明命令[1d Rb, (Ra+)]的图。

图 47 是说明命令[1d Rb2, (Ra2+)]的图。

图 48 是说明命令[1d Rb, (Ra)]的图。

图 49 是说明命令[1d Rb2, (Ra2)]的图。

图 50 是说明命令[1dh Rb, (Ra, D9)]的图。

图 51 是说明命令[1dh Rb3, (Ra3, D4)]的图。

图 52 是说明命令[1dh Rb, (GP, D12)]的图。

图 53 是说明命令[1dh Rb2, (GP, D5)]的图。

图 54 是说明命令[1dh Rb2, (GP)]的图。

图 55 是说明命令[1dh Rb, (SP, D12)]的图。

图 56 是说明命令[1dh Rb2, (SP, D5)]的图。

图 57 是说明命令[1dh Rb2, (SP)]的图。

- 图 58 是说明命令[ldh Rb, (Ra+) I9]的图。
- 图 59 是说明命令[ldh Rb, (Ra+)]的图。
- 图 60 是说明命令[ldh Rb2, (Ra2+)]的图。
- 图 61 是说明命令[ldh Rb, (Ra)]的图。
- 图 62 是说明命令[ldh Rb2, (Ra2)]的图。
- 图 63 是说明命令[ldhu Rb, (Ra, D9)]的图。
- 图 64 是说明命令[ldhu Rb, (GP, D12)]的图。
- 图 65 是说明命令[ldhu Rb, (SP, D12)]的图。
- 图 66 是说明命令[ldhu Rb, (Ra+) I9]的图。
- 图 67 是说明命令[ldhu Rb, (Ra+)]的图。
- 图 68 是说明命令[ldhu Rb, (Ra)]的图。
- 图 69 是说明命令[ldb Rb, (Ra, D8)]的图。
- 图 70 是说明命令[ldb Rb, (GP, D11)]的图。
- 图 71 是说明命令[ldb Rb, (SP, D11)]的图。
- 图 72 是说明命令[ldb Rb, (Ra+) I8]的图。
- 图 73 是说明命令[ldb Rb, (Ra+)]的图。
- 图 74 是说明命令[ldb Rb, (Ra)]的图。
- 图 75 是说明命令[ldbu Rb, (Ra, D8)]的图。
- 图 76 是说明命令[ldbu Rb, (GP, D11)]的图。
- 图 77 是说明命令[ldbu Rb, (SP, D11)]的图。
- 图 78 是说明命令[ldbu Rb, (Ra+) I8]的图。
- 图 79 是说明命令[ldbu Rb, (Ra+)]的图。
- 图 80 是说明命令[ldbu Rb, (Ra)]的图。
- 图 81 是说明命令[ldp Rb:Rb+1, (Ra, D11)]的图。
- 图 82 是说明命令[ldp Rb:Rb+1, (GP, D14)]的图。
- 图 83 是说明命令[ldp Rb:Rb+1, (SP, D14)]的图。
- 图 84 是说明命令[ldp Rb:Rb+1, (SP, D7)]的图。

- 图 85 是说明命令[1dp Rb:Rb+1, (SP)]的图。
- 图 86 是说明命令[1dp Rb:Rb+1, (Ra+) I11]的图。
- 图 87 是说明命令[1dp Rb2:Rb2+1, (Ra2+)]的图。
- 图 88 是说明命令[1dp Rb:Rb+1, (Ra+)]的图。
- 图 89 是说明命令[1dp Rb:Rb+1, (Ra)]的图。
- 图 90 是说明命令[1dp LR:SVR, (Ra, D14)]的图。
- 图 91 是说明命令[1dp LR:SVR, (Ra)]的图。
- 图 92 是说明命令[1dp LR:SVR, (GP, D14)]的图。
- 图 93 是说明命令[1dp LR:SVR, (SP, D14)]的图。
- 图 94 是说明命令[1dp LR:SVR, (SP, D7)]的图。
- 图 95 是说明命令[1dp LR:SVR, (SP)]的图。
- 图 96 是说明命令[1dp TAR:UDR, (Ra, D11)]的图。
- 图 97 是说明命令[1dp TAR:UDR, (GP, D14)]的图。
- 图 98 是说明命令[1dp TAR:UDR, (SP, D14)]的图。
- 图 99 是说明命令[1dhp Rb:Rb+1, (Ra, D10)]的图。
- 图 100 是说明命令[1dhp Rb:Rb+1, (Ra+) I10]的图。
- 图 101 是说明命令[1dhp Rb2:Rb2+1, (Ra2+)]的图。
- 图 102 是说明命令[1dhp Rb:Rb+1, (Ra+)]的图。
- 图 103 是说明命令[1dhp Rb:Rb+1, (Ra)]的图。
- 图 104 是说明命令[1dbp Rb:Rb+1, (Ra, D9)]的图。
- 图 105 是说明命令[1dbp Rb:Rb+1, (Ra+) I9]的图。
- 图 106 是说明命令[1dbp Rb:Rb+1, (Ra+)]的图。
- 图 107 是说明命令[1dbp Rb:Rb+1, (Ra)]的图。
- 图 108 是说明命令[1dbh Rb, (Ra+) I7]的图。
- 图 109 是说明命令[1dbh Rb, (Ra+)]的图。
- 图 110 是说明命令[1dbh Rb, (Ra)]的图。
- 图 111 是说明命令[1dbuh Rb, (Ra+) I7]的图。

图 112 是说明命令[l**dbuh** Rb, (Ra+)]的图。

图 113 是说明命令[l**dbuh** Rb, (Ra)]的图。

图 114 是说明命令[l**dbhp** Rb:Rb+1, (Ra+)I7]的图。

图 115 是说明命令[l**dbhp** Rb:Rb+1, (Ra+)]的图。

图 116 是说明命令[l**dbhp** Rb:Rb+1, (Ra)]的图。

图 117 是说明命令[l**dbuhp** Rb:Rb+1, (Ra+)I7]的图。

图 118 是说明命令[l**dbuhp** Rb:Rb+1, (Ra+)]的图。

图 119 是说明命令[l**dbuhp** Rb:Rb+1, (Ra)]的图。

图 120 是说明命令[**st** (Ra, D10), Rb] 的图。

图 121 是说明命令[**st** (Ra3, D5), Rb3] 的图。

图 122 是说明命令[**st** (GP, D13), Rb] 的图。

图 123 是说明命令[**st** (GP, D6), Rb2] 的图。

图 124 是说明命令[**st** (GP), Rb2] 的图。

图 125 是说明命令[**st** (SP, D13), Rb] 的图。

图 126 是说明命令[**st** (SP, D6), Rb2] 的图。

图 127 是说明命令[**st** (SP), Rb2] 的图。

图 128 是说明命令[**st** (Ra+)I10, Rb] 的图。

图 129 是说明命令[**st** (Ra+), Rb] 的图。

图 130 是说明命令[**st** (Ra2+), Rb] 的图。

图 131 是说明命令[**st** (Ra), Rb] 的图。

图 132 是说明命令[**st** (Ra2), Rb2] 的图。

图 133 是说明命令[**sth** (Ra, D9), Rb] 的图。

图 134 是说明命令[**sth** (Ra3, D4), Rb3] 的图。

图 135 是说明命令[**sth** (GP, D12), Rb] 的图。

图 136 是说明命令[**sth** (GP, D5), Rb2] 的图。

图 137 是说明命令[**sth** (GP), Rb2] 的图。

图 138 是说明命令[**sth** (SP, D12), Rb] 的图。

- 图 139 是说明命令[sth (SP, D5), Rb2] 的图。
- 图 140 是说明命令[sth (SP), Rb2] 的图。
- 图 141 是说明命令[sth (Ra+) I9, Rb] 的图。
- 图 142 是说明命令[sth (Ra+), Rb] 的图。
- 图 143 是说明命令[sth (Ra2+), Rb2] 的图。
- 图 144 是说明命令[sth (Ra), Rb] 的图。
- 图 145 是说明命令[st (Ra2), Rb2] 的图。
- 图 146 是说明命令[stb (Ra, D8), Rb] 的图。
- 图 147 是说明命令[stb (GP, D11), Rb] 的图。
- 图 148 是说明命令[stb (SP, D11), Rb] 的图。
- 图 149 是说明命令[stb (Ra+) I8, Rb] 的图。
- 图 150 是说明命令[stb (Ra+), Rb] 的图。
- 图 151 是说明命令[stb (Ra), Rb] 的图。
- 图 152 是说明命令[stp (Ra, D11), Rb:Rb+1] 的图。
- 图 153 是说明命令[stp (GP, D14), Rb:Rb+1] 的图。
- 图 154 是说明命令[stp (SP, D14), Rb:Rb+1] 的图。
- 图 155 是说明命令[stp (SP, D7), Rb:Rb+1] 的图。
- 图 156 是说明命令[stp (SP), Rb:Rb+1] 的图。
- 图 157 是说明命令[stp (Ra+) I11, Rb:Rb+1] 的图。
- 图 158 是说明命令[stp (Ra+), Rb:Rb+1] 的图。
- 图 159 是说明命令[stp (Ra2+), Rb2:Rb2+1] 的图。
- 图 160 是说明命令[stp (Ra), Rb:Rb+1] 的图。
- 图 161 是说明命令[stp (Ra, D11), LR:SVR] 的图。
- 图 162 是说明命令[stp (Ra), LR:SVR] 的图。
- 图 163 是说明命令[stp (GP, D14), LR:SVR] 的图。
- 图 164 是说明命令[stp (SP, D14), LR:SVR] 的图。
- 图 165 是说明命令[stp (SP, D7), LR:SVR] 的图。

- 图 166 是说明命令[stp (SP), LR:SVR] 的图。
- 图 167 是说明命令[stp (Ra, D11), TAR:UDR] 的图。
- 图 168 是说明命令[stp (GP, D14), TAR:UDR] 的图。
- 图 169 是说明命令[stp (SP, D14), TAR:UDR] 的图。
- 图 170 是说明命令[sthp (Ra, D10), Rb:Rb+1] 的图。
- 图 171 是说明命令[sthp (Ra+) I10, Rb:Rb+1] 的图。
- 图 172 是说明命令[sthp (Ra+), Rb:Rb+1] 的图。
- 图 173 是说明命令[sthp (Ra2+) I10, Rb2:Rb2+1] 的图。
- 图 174 是说明命令[sthp (Ra), Rb:Rb+1] 的图。
- 图 175 是说明命令[stbp (Ra, D9), Rb:Rb+1] 的图。
- 图 176 是说明命令[stbp (Ra+) I9, Rb:Rb+1] 的图。
- 图 177 是说明命令[stbp (Ra+), Rb:Rb+1] 的图。
- 图 178 是说明命令[stbp (Ra), Rb:Rb+1] 的图。
- 图 179 是说明命令[stbp (Ra+) I7, Rb] 的图。
- 图 180 是说明命令[stbp (Ra+), Rb] 的图。
- 图 181 是说明命令[stbp (Ra), Rb] 的图。
- 图 182 是说明命令[stbhp (Ra+) I7, Rb:Rb+1] 的图。
- 图 183 是说明命令[stbhp (Ra+), Rb:Rb+1] 的图。
- 图 184 是说明命令[stbhp (Ra), Rb:Rb+1] 的图。
- 图 185 是说明命令[dpref (Ra, D8)] 的图。
- 图 186 是说明命令[ldestb Rb, (Ra)] 的图。
- 图 187 是说明命令[rd C0:C1, Rb, (D11)] 的图。
- 图 188 是说明命令[rd C0:C1, Rb, (Ra, D5)] 的图。
- 图 189 是说明命令[rd C0:C1, Rb, (Ra)] 的图。
- 图 190 是说明命令[rd C0:C1, Rb2, (Ra2)] 的图。
- 图 191 是说明命令[rd C2:C3, Rb, (Ra, D5)] 的图。
- 图 192 是说明命令[rd C2:C3, Rb, (Ra)] 的图。

图 193 是说明命令[rde C0:C1, Rb, (Ra, D5)] 的图。

图 194 是说明命令[rde C0:C1, Rb, (Ra)] 的图。

图 195 是说明命令[rde C2:C3, Rb, (Ra, D5)] 的图。

图 196 是说明命令[rde C2:C3, Rb, (Ra)] 的图。

图 197 是说明命令[wt C0:C1, (D11), Rb] 的图。

图 198 是说明命令[wt C0:C1, (Ra, D5), Rb] 的图。

图 199 是说明命令[wt C0:C1, (Ra), Rb] 的图。

图 200 是说明命令[wt C0:C1, (Ra2), Rb2] 的图。

图 201 是说明命令[wt C2:C3, (Ra, D5), Rb] 的图。

图 202 是说明命令[wt C2:C3, (Ra), Rb] 的图。

图 203 是说明命令[wte C0:C1, (Ra, D5), Rb] 的图。

图 204 是说明命令[wte C0:C1, (Ra), Rb] 的图。

图 205 是说明命令[wte C2:C3, (Ra, D5), Rb] 的图。

图 206 是说明命令[wte C2:C3, (Ra), Rb] 的图。

图 207 是说明命令[br D20] 的图。

图 208 是说明命令[br D9] 的图。

图 209 是说明命令[br1 D20] 的图。

图 210 是说明命令[call D20] 的图。

图 211 是说明命令[br1 D9] 的图。

图 212 是说明命令[call D9] 的图。

图 213 是说明命令[jmp LR] 的图。

图 214 是说明命令[jmp TAR] 的图。

图 215 是说明命令[jmpl LR] 的图。

图 216 是说明命令[call LR] 的图。

图 217 是说明命令[jmpl TAR] 的图。

图 218 是说明命令[call TAR] 的图。

图 219 是说明命令[jmpr LR] 的图。

- 图 220 是说明命令[ret] 的图。
- 图 221 是说明命令[jmpf LR] 的图。
- 图 222 是说明命令[jmpf C6, C2:C4, TAR] 的图。
- 图 223 是说明命令[jmpf Cm, TAR] 的图。
- 图 224 是说明命令[jmpf TAR] 的图。
- 图 225 是说明命令[jloop C6, TAR, Ra, I8] 的图。
- 图 226 是说明命令[jloop C6, TAR, Ra] 的图。
- 图 227 是说明命令[jloop C6, TAR, Ra2] 的图。
- 图 228 是说明命令[jloop C6, TAR, Ra2, 1] 的图。
- 图 229 是说明命令[jloop C6, Cm, TAR, Ra, I8] 的图。
- 图 230 是说明命令[jloop C6, Cm, TAR, Ra] 的图。
- 图 231 是说明命令[jloop C6, Cm, TAR, Ra2] 的图。
- 图 232 是说明命令[jloop C6, Cm, TAR, Ra2, 1] 的图。
- 图 233 是说明命令[jloop C6, C2:C4, TAR, Ra, I8] 的图。
- 图 234 是说明命令[jloop C6, C2:C4, TAR, Ra] 的图。
- 图 235 是说明命令[jloop C6, C2:C4, TAR, Ra2] 的图。
- 图 236 是说明命令[jloop C6, C2:C4, TAR, Ra2, 1] 的图。
- 图 237 是说明命令[jloop C5, LR, Ra, I8] 的图。
- 图 238 是说明命令[jloop C5, LR, Ra] 的图。
- 图 239 是说明命令[settar D9] 的图。
- 图 240 是说明命令[settar C6, Cm, D9] 的图。
- 图 241 是说明命令[settar C6, D9] 的图。
- 图 242 是说明命令[settar C6, C2:C4, D9] 的图。
- 图 243 是说明命令[settar C6, C4, D9] 的图。
- 图 244 是说明命令[setlr D9] 的图。
- 图 245 是说明命令[setlr C5, D9] 的图。
- 图 246 是说明命令[setbb TAR] 的图。

图 247 是说明命令[setbb LR] 的图。

图 248 是说明命令[intd] 的图。

图 249 是说明命令[inte] 的图。

图 250 是说明命令[vmpswd] 的图。

图 251 是说明命令[vmpswe] 的图。

图 252 是说明命令[vmpsleee] 的图。

图 253 是说明命令[vmpwait] 的图。

图 254 是说明命令[vmpsus] 的图。

图 255 是说明命令[rti] 的图。

图 256 是说明命令[piNl (pi0l, pi1l, pi2l, pi3l, pi4l, pi5l, pi6l, pi7l)] 的图。

图 257 是说明命令[piN(pi0, pi1, pi2, pi3, pi4, pi5, pi6, pi7)] 的图。

图 258 是说明命令[scN(sc0, scl, sc2, sc3, sc4, sc5, sc6, sc7)] 的图。

图 259 是说明命令[add Rc, Ra, Rb] 的图。

图 260 是说明命令[add Rc3, Ra3, Rb3] 的图。

图 261 是说明命令[add Ra2, Rb2] 的图。

图 262 是说明命令[add Rb, Ra2, I12] 的图。

图 263 是说明命令[add Ra2, I5] 的图。

图 264 是说明命令[add SP, I19] 的图。

图 265 是说明命令[add SP, I11] 的图。

图 266 是说明命令[addu Rb, GP, I13] 的图。

图 267 是说明命令[addu Rb, SP, I13] 的图。

图 268 是说明命令[addu Ra3, SP, I6] 的图。

图 269 是说明命令[addvw Rc, Ra, Rb] 的图。

图 270 是说明命令[addvh Rc, Ra, Rb] 的图。

- 图 271 是说明命令[addc Rc, Ra, Rb] 的图。
- 图 272 是说明命令[adds Rc, Ra, Rb] 的图。
- 图 273 是说明命令[addsr Rc, Ra, Rb] 的图。
- 图 274 是说明命令[sladd Rc, Ra, Rb] 的图。
- 图 275 是说明命令[sladd Rc3, Ra3, Rb3] 的图。
- 图 276 是说明命令[s2add Rc, Ra, Rb] 的图。
- 图 277 是说明命令[s2add Rc3, Ra3, Rb3] 的图。
- 图 278 是说明命令[addmsk Rc, Ra, Rb] 的图。
- 图 279 是说明命令[addarvw Rc, Ra, Rb] 的图。
- 图 280 是说明命令[sub Rc, Rb, Ra] 的图。
- 图 281 是说明命令[sub Rc3, Rb3, Ra3] 的图。
- 图 282 是说明命令[sub Rb2, Ra2] 的图。
- 图 283 是说明命令[sub Rb2, Ra2, I12] 的图。
- 图 284 是说明命令[sub Ra2, I5] 的图。
- 图 285 是说明命令[sub SP, I19] 的图。
- 图 286 是说明命令[sub SP, I11] 的图。
- 图 287 是说明命令[subc Rc, Rb, Ra] 的图。
- 图 288 是说明命令[subvw Rc, Rb, Ra] 的图。
- 图 289 是说明命令[subvh Rc, Rb, Ra] 的图。
- 图 290 是说明命令[subs Rc, Rb, Ra] 的图。
- 图 291 是说明命令[submsk Rc, Rb, Ra] 的图。
- 图 292 是说明命令[rsub Rb, Ra, I8] 的图。
- 图 293 是说明命令[rsub Ra2, I4] 的图。
- 图 294 是说明命令[rsub Ra2, Rb2] 的图。
- 图 295 是说明命令[neg Rb, Ra] 的图。
- 图 296 是说明命令[neg Ra2] 的图。
- 图 297 是说明命令[negvh Rb, Ra] 的图。

- 图 298 是说明命令[negvw Rb, Ra] 的图。
- 图 299 是说明命令[abs Rb, Ra] 的图。
- 图 300 是说明命令[absvw Rb, Ra] 的图。
- 图 301 是说明命令[absvh Rb, Ra] 的图。
- 图 302 是说明命令[max Rc, Ra, Rb] 的图。
- 图 303 是说明命令[min Rc, Ra, Rb] 的图。
- 图 304 是说明命令[and Rc, Ra, Rb] 的图。
- 图 305 是说明命令[and Ra2, Rb2] 的图。
- 图 306 是说明命令[and Rb, Ra, I8] 的图。
- 图 307 是说明命令[andn Rc, Ra, Rb] 的图。
- 图 308 是说明命令[andn Ra2, Rb2] 的图。
- 图 309 是说明命令[andn Rb, Ra, I8] 的图。
- 图 310 是说明命令[or Rc, Ra, Rb] 的图。
- 图 311 是说明命令[or Ra2, Rb2] 的图。
- 图 312 是说明命令[or Rb, Ra, I8] 的图。
- 图 313 是说明命令[xor Rc, Ra, Rb] 的图。
- 图 314 是说明命令[xor Ra2, Rb2] 的图。
- 图 315 是说明命令[xor Rb, Ra, I8] 的图。
- 图 316 是说明命令[not Rb, Ra] 的图。
- 图 317 是说明命令[not Ra2] 的图。
- 图 318 是说明命令[cmpCC Rm, Ra, Rb] 的图。
- 图 319 是说明命令[cmpCC C6, Ra2, Rb2] 的图。
- 图 320 是说明命令[cmpCC] 的图。
- 图 321 是说明命令[cmpCC R6, Ra2, I4] 的图。
- 图 322 是说明命令[cmpCC] 的图。
- 图 323 是说明命令[cmpCC] 的图。
- 图 324 是说明命令[cmpCCn Cm, Ra, Rb, Cn] 的图。

- 图 325 是说明命令 [cmpCCn Cm, Ra, I5, Cn] 的图。
- 图 326 是说明命令 [cmpCCn Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 327 是说明命令 [cmpCCn Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 328 是说明命令 [cmpCCa Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 329 是说明命令 [cmpCCa Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 330 是说明命令 [cmpCCo Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 331 是说明命令 [cmpCCo Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 332 是说明命令 [tstz Cm, Ra, Rb] 的图。
- 图 333 是说明命令 [tstz C6, Ra2, Rb2] 的图。
- 图 334 是说明命令 [tstz Cm, Ra, I5] 的图。
- 图 335 是说明命令 [tstz C6, Ra2, I4] 的图。
- 图 336 是说明命令 [tstz Cm:Cm+1, Ra, Rb] 的图。
- 图 337 是说明命令 [tstz Cm:Cm+1, Ra, I5] 的图。
- 图 338 是说明命令 [tstzn Cm, Ra, Rb, Cn] 的图。
- 图 339 是说明命令 [tstzn Cm, Ra, I5, Cn] 的图。
- 图 340 是说明命令 [tstzn Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 341 是说明命令 [tstzn Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 342 是说明命令 [tstza Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 343 是说明命令 [tstza Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 344 是说明命令 [tstzo Cm:Cm+1, Ra, Rb, Cn] 的图。
- 图 345 是说明命令 [tstzo Cm:Cm+1, Ra, I5, Cn] 的图。
- 图 346 是说明命令 [tstn Cm, Ra, Rb] 的图。
- 图 347 是说明命令 [tstn C6, Ra2, Rb2] 的图。
- 图 348 是说明命令 [tstn Cm, Ra, I5] 的图。
- 图 349 是说明命令 [tstn C6, Ra2, I4] 的图。
- 图 350 是说明命令 [tstn Cm:Cm+1, Ra, Rb] 的图。
- 图 351 是说明命令 [tstn Cm:Cm+1, Ra, I5] 的图。

图 352 是说明命令 [tstnn Cm, Ra, Rb, Cn] 的图。

图 353 是说明命令 [tstnn Cm, Ra, I5, Cn] 的图。

图 354 是说明命令 [tstnn Cm:Cm+1, Ra, Rb, Cn] 的图。

图 355 是说明命令 [tstnn Cm:Cm+1, Ra, I5, Cn] 的图。

图 356 是说明命令 [tstna Cm:Cm+1, Ra, Rb, Cn] 的图。

图 357 是说明命令 [tstna Cm:Cm+1, Ra, I5, Cn] 的图。

图 358 是说明命令 [tstno Cm:Cm+1, Ra, Rb, Cn] 的图。

图 359 是说明命令 [tstno Cm:Cm+1, Ra, I5, Cn] 的图。

图 360 是说明命令 [mov Rb, Ra] 的图。

图 361 是说明命令 [mov Ra2, Rb] 的图。

图 362 是说明命令 [mov Ra, I16] 的图。

图 363 是说明命令 [mov Ra2, I8] 的图。

图 364 是说明命令 [mov Rb, TAR] 的图。

图 365 是说明命令 [mov Rb2, TAR] 的图。

图 366 是说明命令 [mov Rb, LR] 的图。

图 367 是说明命令 [mov Rb2, LR] 的图。

图 368 是说明命令 [mov Rb, SVR] 的图。

图 369 是说明命令 [mov Rb, PSR] 的图。

图 370 是说明命令 [mov Rb, CFR] 的图。

图 371 是说明命令 [mov Rb, MH0] 的图。

图 372 是说明命令 [mov Rb2, MH0] 的图。

图 373 是说明命令 [mov Rb, MH1] 的图。

图 374 是说明命令 [mov Rb2, MH1] 的图。

图 375 是说明命令 [mov Rb, ML0] 的图。

图 376 是说明命令 [mov Rb, ML1] 的图。

图 377 是说明命令 [mov Rb, IPC] 的图。

图 378 是说明命令 [mov Rb, IPSR] 的图。

图 379 是说明命令 [mov Rb, PC] 的图。
图 380 是说明命令 [mov Rb, EPC] 的图。
图 381 是说明命令 [mov Rb, EPSR] 的图。
图 382 是说明命令 [mov Rb, PSR0] 的图。
图 383 是说明命令 [mov Rb, PSR1] 的图。
图 384 是说明命令 [mov Rb, PSR2] 的图。
图 385 是说明命令 [mov Rb, PSR3] 的图。
图 386 是说明命令 [mov Rb, CFR0] 的图。
图 387 是说明命令 [mov Rb, CFR1] 的图。
图 388 是说明命令 [mov Rb, CFR2] 的图。
图 389 是说明命令 [mov Rb, CFR3] 的图。
图 390 是说明命令 [mov LR, Rb] 的图。
图 391 是说明命令 [mov LR, Rb2] 的图。
图 392 是说明命令 [mov TAR, Rb] 的图。
图 393 是说明命令 [mov TAR, Rb2] 的图。
图 394 是说明命令 [mov SVR, Rb] 的图。
图 395 是说明命令 [mov PSR, Rb] 的图。
图 396 是说明命令 [mov CFR, Rb] 的图。
图 397 是说明命令 [mov MH0, Rb] 的图。
图 398 是说明命令 [mov MH0, Rb2] 的图。
图 399 是说明命令 [mov MH1, Rb] 的图。
图 400 是说明命令 [mov MH1, Rb2] 的图。
图 401 是说明命令 [mov ML0, Rb] 的图。
图 402 是说明命令 [mov ML1, Rb2] 的图。
图 403 是说明命令 [mov IPC, Rb] 的图。
图 404 是说明命令 [mov IPSR, Rb] 的图。
图 405 是说明命令 [mov EPC, Rb] 的图。

- 图 406 是说明命令[`mov EPSR, Rb`] 的图。
- 图 407 是说明命令[`mov PSR0, Rb`] 的图。
- 图 408 是说明命令[`mov PSR1, Rb`] 的图。
- 图 409 是说明命令[`mov PSR2, Rb`] 的图。
- 图 410 是说明命令[`mov PSR3, Rb`] 的图。
- 图 411 是说明命令[`mov CFR0, Rb`] 的图。
- 图 412 是说明命令[`mov CFR1, Rb`] 的图。
- 图 413 是说明命令[`mov CFR2, Rb`] 的图。
- 图 414 是说明命令[`mov CFR3, Rb`] 的图。
- 图 415 是说明命令[`mvclovs Cm:Cm+1`] 的图。
- 图 416 是说明命令[`movcf Ci, Cj, Cm, Cn`] 的图。
- 图 417 是说明命令[`mvclcas Cm:Cm+1`] 的图。
- 图 418 是说明命令[`sethi Ra, I16`] 的图。
- 图 419 是说明命令[`setlo Ra, I16`] 的图。
- 图 420 是说明命令[`vcchk`] 的图。
- 图 421 是说明命令[`nop`] 的图。
- 图 422 是说明命令[`asl Rc, Ra, Rb`] 的图。
- 图 423 是说明命令[`asl Rc, Ra, I5`] 的图。
- 图 424 是说明命令[`asl Ra2, I4`] 的图。
- 图 425 是说明命令[`aslvw Rc, Ra, Rb`] 的图。
- 图 426 是说明命令[`aslvw Rb, Ra, I5`] 的图。
- 图 427 是说明命令[`aslvh Rc, Ra, Rb`] 的图。
- 图 428 是说明命令[`aslvh Rb, Ra, I5`] 的图。
- 图 429 是说明命令[`asr Rc, Ra, Rb`] 的图。
- 图 430 是说明命令[`asr Rb, Ra, I5`] 的图。
- 图 431 是说明命令[`aslvh Ra2, I4`] 的图。
- 图 432 是说明命令[`asrvw Rc, Ra, Rb`] 的图。

- 图 433 是说明命令 [asrvh Rc, Ra, Rb] 的图。
- 图 434 是说明命令 [lsl Rc, Ra, Rb] 的图。
- 图 435 是说明命令 [lsl Rc, Ra, I5] 的图。
- 图 436 是说明命令 [lsl Ra2, I4] 的图。
- 图 437 是说明命令 [lsr Rc, Ra, Rb] 的图。
- 图 438 是说明命令 [lsr Rb, Ra, I5] 的图。
- 图 439 是说明命令 [rol Rc, Ra, Rb] 的图。
- 图 440 是说明命令 [rol Rb, Ra, I5] 的图。
- 图 441 是说明命令 [ror Rb, Ra, I5] 的图。
- 图 442 是说明命令 [aslp Mm, Ra, Mn, Rb] 的图。
- 图 443 是说明命令 [aslp Mm, Rb, Mn, I6] 的图。
- 图 444 是说明命令 [aslp Mm, Rc, MHn, Ra, Rb] 的图。
- 图 445 是说明命令 [aslp Mm, Rb, MHn, Ra, I6] 的图。
- 图 446 是说明命令 [aslpvw Mm, Ra, Mn, Rb] 的图。
- 图 447 是说明命令 [aslpvw Mm, Rb, Mn, I6] 的图。
- 图 448 是说明命令 [asrp Mm, Ra, Mn, Rb] 的图。
- 图 449 是说明命令 [asrp Mm, Rb, Mn, I6] 的图。
- 图 450 是说明命令 [asrp Mm, Rc, MHn, Ra, Rb] 的图。
- 图 451 是说明命令 [asrp Mm, Rb, MHn, Ra, I6] 的图。
- 图 452 是说明命令 [asrpvw Mm, Ra, Mn, Rb] 的图。
- 图 453 是说明命令 [lslp Mm, Ra, Mn, Rb] 的图。
- 图 454 是说明命令 [lslp Mm, Rb, Mn, I6] 的图。
- 图 455 是说明命令 [lslp Mm, Rc, MHn, Ra, Rb] 的图。
- 图 456 是说明命令 [lslp Mm, Rc, MHn, Ra, I6] 的图。
- 图 457 是说明命令 [lsrp Mm, Ra, Mn, Rb] 的图。
- 图 458 是说明命令 [lsrp Mm, Ra, Mn, I6] 的图。
- 图 459 是说明命令 [lsrp Mm, Rc, MHn, Ra, Rb] 的图。

图 460 是说明命令[lslp Mm, Rc, MHn, Ra, I6] 的图。

图 461 是说明命令[extr Rc, Ra, Rb] 的图。

图 462 是说明命令[extr Rb, Ra, Ib5, Ia5] 的图。

图 463 是说明命令[ext Rb, Ra, I5] 的图。

图 464 是说明命令[exth Ra2] 的图。

图 465 是说明命令[extb Ra2] 的图。

图 466 是说明命令[extru Rc, Ra, Rb]的图。

图 467 是说明命令[extru Rb, Ra, Ib5, Ia5] 的图。

图 468 是说明命令[extu Rb, Ra, I5] 的图。

图 469 是说明命令[exthu Ra2] 的图。

图 470 是说明命令[extbu Ra2] 的图。

图 471 是说明命令[mskgen Rc, Rb]的图。

图 472 是说明命令[mskgen Rb, Ib5, Ia5] 的图。

图 473 是说明命令[msk Rc, Ra, Rb]的图。

图 474 是说明命令[msk Rb, Ra, Ib5, Ia5] 的图。

图 475 是说明命令[satw Mm, Rb, Mn] 的图。

图 476 是说明命令[sath Rb, Ra] 的图。

图 477 是说明命令[sat12 Rb, Ra] 的图。

图 478 是说明命令[sat9 Rb, Ra] 的图。

图 479 是说明命令[satb Rb, Ra] 的图。

图 480 是说明命令[satbu Rb, Ra] 的图。

图 481 是说明命令[extw Mm, Rb, Ra] 的图。

图 482 是说明命令[vintl1h Rc, Ra, Rb]的图。

图 483 是说明命令[vintl1hh Rc, Ra, Rb]的图。

图 484 是说明命令[vintl1b Rc, Ra, Rb]的图。

图 485 是说明命令[vintl1hb Rc, Ra, Rb]的图。

图 486 是说明命令[valn Rc, Ra, Rb]的图。

- 图 487 是说明命令[valn1 Rc, Ra, Rb]的图。
- 图 488 是说明命令[valn2 Rc, Ra, Rb]的图。
- 图 489 是说明命令[valn3 Rc, Ra, Rb]的图。
- 图 490 是说明命令[valnvc1 Rc, Ra, Rb]的图。
- 图 491 是说明命令[valnvc2 Rc, Ra, Rb]的图。
- 图 492 是说明命令[valnvc3 Rc, Ra, Rb]的图。
- 图 493 是说明命令[valnvc4 Rc, Ra, Rb]的图。
- 图 494 是说明命令[vxchngh Rb, Ra]的图。
- 图 495 是说明命令[bytereve Rb, Ra]的图。
- 图 496 是说明命令[vstovb Rb, Ra]的图。
- 图 497 是说明命令[vstovh Rb, Ra]的图。
- 图 498 是说明命令[vlunpkh Rb:Rb+1, Ra]的图。
- 图 499 是说明命令[vlunpkhu Rb:Rb+1, Ra]的图。
- 图 500 是说明命令[vlunpkb Rb:Rb+1, Ra]的图。
- 图 501 是说明命令[vlunpkbu Rb:Rb+1, Ra]的图。
- 图 502 是说明命令[vhunpkh Rb:Rb+1, Ra]的图。
- 图 503 是说明命令[vhunpkb Rb:Rb+1, Ra]的图。
- 图 504 是说明命令[vunpk1 Rb, Mn]的图。
- 图 505 是说明命令[vunpk2 Rb, Mn]的图。
- 图 506 是说明命令[vlpkh Rc, Ra, Rb]的图。
- 图 507 是说明命令[vlpkhu Rc, Ra, Rb]的图。
- 图 508 是说明命令[vlpkb Rc, Ra, Rb]的图。
- 图 509 是说明命令[vlpkbu Rc, Ra, Rb]的图。
- 图 510 是说明命令[vhpkh Rc, Ra, Rb]的图。
- 图 511 是说明命令[vhpkb Rc, Ra, Rb]的图。
- 图 512 是说明命令[vexth Mm, Rb, Ra]的图。
- 图 513 是说明命令[bseq0 Rb, Ra]的图。

- 图 514 是说明命令[bseq1 Rb, Ra]的图。
- 图 515 是说明命令[bseq Rb, Ra]的图。
- 图 516 是说明命令[bcnt1 Rb, Ra]的图。
- 图 517 是说明命令[rndvh Rb, Ra]的图。
- 图 517 是说明命令[rndvh Rc, Ra, Rb]的图。
- 图 518 是说明命令[mskbrvb Rc, Ra, Rb]的图。
- 图 519 是说明命令[mskbrvh Rc, Ra, Rb]的图。
- 图 520 是说明命令[movp Rc:Rc+1, Ra, Rb]的图。
- 图 521 是说明命令[hmul Mm, Rc, Ra, Rb]的图。
- 图 522 是说明命令[lmul Mm, Rc, Ra, Rb]的图。
- 图 523 是说明命令[fmulhh Mm, Rc, Ra, Rb]的图。
- 图 524 是说明命令[fmulhhr Mm, Rc, Ra, Rb]的图。
- 图 525 是说明命令[fmulhw Mm, Rc, Ra, Rb]的图。
- 图 526 是说明命令[fmulhww Mm, Rc, Ra, Rb]的图。
- 图 527 是说明命令[mul Mm, Rc, Ra, Rb]的图。
- 图 528 是说明命令[mul Mm, Rc, Ra, I8]的图。
- 图 529 是说明命令[mulu Mm, Rc, Ra, Rb]的图。
- 图 530 是说明命令[mulu Mm, Rc, Ra, I8]的图。
- 图 531 是说明命令[fmulww Mm, Rc, Ra, Rb]的图。
- 图 532 是说明命令[hmac Mm, Rc, Ra, Rb, Mn]的图。
- 图 533 是说明命令[hmac M0, Rc, Ra, Rb, Rx]的图。
- 图 534 是说明命令[lmac Mm, Rc, Ra, Rb, Mn]的图。
- 图 535 是说明命令[lmac M0, Rc, Ra, Rb, Rx]的图。
- 图 536 是说明命令[fmachh Mm, Rc, Ra, Rb, Mn]的图。
- 图 537 是说明命令[fmachh M0, Rc, Ra, Rb, Rx]的图。
- 图 538 是说明命令[fmachhr Mm, Rc, Ra, Rb, Mn]的图。
- 图 539 是说明命令[fmachhr M0, Rc, Ra, Rb, Rx]的图。

- 图 540 是说明命令 [fmachw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 541 是说明命令 [fmachw M0, Rc, Ra, Rb, Rx] 的图。
- 图 542 是说明命令 [fmachww Mm, Rc, Ra, Rb, Mn] 的图。
- 图 543 是说明命令 [fmachww M0, Rc, Ra, Rb, Rx] 的图。
- 图 544 是说明命令 [mac Mm, Rc, Ra, Rb, Mn] 的图。
- 图 545 是说明命令 [mac M0, Rc, Ra, Rb, Rx] 的图。
- 图 546 是说明命令 [fmacww Mm, Rc, Ra, Rb, Mn] 的图。
- 图 547 是说明命令 [fmacww M0, Rc, Ra, Rb, Rx] 的图。
- 图 548 是说明命令 [hmsu Mm, Rc, Ra, Rb, Mn] 的图。
- 图 549 是说明命令 [hmsu M0, Rc, Ra, Rb, Rx] 的图。
- 图 550 是说明命令 [lmsu Mm, Rc, Ra, Rb, Mn] 的图。
- 图 551 是说明命令 [lmsu M0, Rc, Ra, Rb, Rx] 的图。
- 图 552 是说明命令 [fmsuhh Mm, Rc, Ra, Rb, Mn] 的图。
- 图 553 是说明命令 [fmsuhh M0, Rc, Ra, Rb, Rx] 的图。
- 图 554 是说明命令 [fmsuhhr Mm, Rc, Ra, Rb, Mn] 的图。
- 图 555 是说明命令 [fmsuhhr M0, Rc, Ra, Rb, Rx] 的图。
- 图 556 是说明命令 [fmsuhw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 557 是说明命令 [fmsuhw M0, Rc, Ra, Rb, Rx] 的图。
- 图 558 是说明命令 [fmsuhww Mm, Rc, Ra, Rb, Mn] 的图。
- 图 559 是说明命令 [fmsuhww M0, Rc, Ra, Rb, Rx] 的图。
- 图 560 是说明命令 [msu Mm, Rc, Ra, Rb, Mn] 的图。
- 图 561 是说明命令 [msu M0, Rc, Ra, Rb, Rx] 的图。
- 图 562 是说明命令 [fmsuww Mm, Rc, Ra, Rb, Mn] 的图。
- 图 563 是说明命令 [fmsuww M0, Rc, Ra, Rb, Rx] 的图。
- 图 564 是说明命令 [div MHm, Rc, MHn, Ra, Rb] 的图。
- 图 565 是说明命令 [divu MHm, Rc, MHn, Ra, Rb] 的图。
- 图 566 是说明命令 [dbgm0] 的图。

- 图 567 是说明命令[dbgm1]的图。
- 图 568 是说明命令[dbgm2 I15]的图。
- 图 569 是说明命令[dbgm3 I15]的图。
- 图 570 是说明命令[vaddh Rc, Ra, Rb]的图。
- 图 571 是说明命令[vxaddh Rc, Ra, Rb]的图。
- 图 572 是说明命令[vhaddh Rc, Ra, Rb]的图。
- 图 573 是说明命令[vladdh Rc, Ra, Rb]的图。
- 图 574 是说明命令[vaddhvh Rc, Ra, Rb]的图。
- 图 575 是说明命令[vxaddhvh Rc, Ra, Rb]的图。
- 图 576 是说明命令[vhaddhvh Rc, Ra, Rb]的图。
- 图 577 是说明命令[vladdhvh Rc, Ra, Rb]的图。
- 图 578 是说明命令[vsaddh Rb, Ra, I8]的图。
- 图 579 是说明命令[vaddsh Rc, Ra, Rb]的图。
- 图 580 是说明命令[vaddsrh Rc, Ra, Rb]的图。
- 图 581 是说明命令[vaddhvc Rc, Ra, Rb]的图。
- 图 582 是说明命令[vaddrhvc Rc, Ra, Rb]的图。
- 图 583 是说明命令[vaddb Rc, Ra, Rb]的图。
- 图 584 是说明命令[vsaddb Rb, Ra, I8]的图。
- 图 585 是说明命令[vaddsb Rc, Ra, Rb]的图。
- 图 586 是说明命令[vaddsrb Rc, Ra, Rb]的图。
- 图 587 是说明命令[vsubh Rc, Rb, Ra]的图。
- 图 588 是说明命令[vxsubh Rc, Rb, Ra]的图。
- 图 589 是说明命令[vhsubh Rc, Rb, Ra]的图。
- 图 590 是说明命令[vlsubh Rc, Rb, Ra]的图。
- 图 591 是说明命令[vsubhvh Rc, Rb, Ra]的图。
- 图 592 是说明命令[vxsubhvh Rc, Rb, Ra]的图。
- 图 593 是说明命令[vhsubhvh Rc, Rb, Ra]的图。

图 594 是说明命令 [vlsubhvh Rc, Rb, Ra] 的图。
图 595 是说明命令 [vssubh Rb, Ra, I8] 的图。
图 596 是说明命令 [vsubb Rc, Rb, Ra] 的图。
图 597 是说明命令 [vssubb Rb, Ra, I8] 的图。
图 598 是说明命令 [vsubsh Rc, Rb, Ra] 的图。
图 599 是说明命令 [vsrsubh Rb, Ra, I8] 的图。
图 600 是说明命令 [vsrsubb Rb, Ra, I8] 的图。
图 601 是说明命令 [vsumh Rb, Ra] 的图。
图 602 是说明命令 [vsumh2 Rb, Ra] 的图。
图 603 是说明命令 [vsumrh2 Rb, Ra] 的图。
图 604 是说明命令 [vnegh Rb, Ra] 的图。
图 605 是说明命令 [vneghvh Rb, Ra] 的图。
图 606 是说明命令 [vnegb Rb, Ra] 的图。
图 607 是说明命令 [vabshvh Rb, Ra] 的图。
图 608 是说明命令 [vasubb Rc, Rb, Ra] 的图。
图 609 是说明命令 [vsgh Rb, Ra] 的图。
图 610 是说明命令 [vmaxh Rc, Ra, Rb] 的图。
图 611 是说明命令 [vmaxb Rc, Ra, Rb] 的图。
图 612 是说明命令 [vminh Rc, Ra, Rb] 的图。
图 613 是说明命令 [vminb Rc, Ra, Rb] 的图。
图 614 是说明命令 [vsel Rc, Ra, Rb] 的图。
图 615 是说明命令 [vmovt Rb, Ra] 的图。
图 616 是说明命令 [vscmpeq b Ra, I5] 的图。
图 617 是说明命令 [vscmpne b Ra, I5] 的图。
图 618 是说明命令 [vscmpgt b Ra, I5] 的图。
图 619 是说明命令 [vscmple b Ra, I5] 的图。
图 620 是说明命令 [vscmpge b Ra, I5] 的图。

图 621 是说明命令[vscmpltb Ra, I5]的图。
图 622 是说明命令[vscmpeqh Ra, I5]的图。
图 623 是说明命令[vscmpneh Ra, I5]的图。
图 624 是说明命令[vscmpgth Ra, I5]的图。
图 625 是说明命令[vscmpleh Ra, I5]的图。
图 626 是说明命令[vscmpgeh Ra, I5]的图。
图 627 是说明命令[vscmplth Ra, I5]的图。
图 628 是说明命令[vcmpeqh Ra, Rb]的图。
图 629 是说明命令[vcmpneh Ra, Rb]的图。
图 630 是说明命令[vcmpgth Ra, Rb]的图。
图 631 是说明命令[vcmpleh Ra, Rb]的图。
图 632 是说明命令[vcmpgeh Ra, Rb]的图。
图 633 是说明命令[vcmplth Ra, Rb]的图。
图 634 是说明命令[vcmpeqb Ra, Rb]的图。
图 635 是说明命令[vcmpneb Ra, Rb]的图。
图 636 是说明命令[vcmpgtb Ra, Rb]的图。
图 637 是说明命令[vcmpleb Ra, Rb]的图。
图 638 是说明命令[vcmpgeb Ra, Rb]的图。
图 639 是说明命令[vcmpltb Ra, Rb]的图。
图 640 是说明命令[vaslh Rc, Ra, Rb]的图。
图 641 是说明命令[vaslh Rb, Ra, I4]的图。
图 642 是说明命令[vaslvh Rc, Ra, Rb]的图。
图 643 是说明命令[vaslvh Rb, Ra, I4]的图。
图 644 是说明命令[vasrh Rb, Ra, I4]的图。
图 645 是说明命令[vasrvh Rc, Ra, Rb]的图。
图 646 是说明命令[vlslh Rc, Ra, Rb]的图。
图 647 是说明命令[vlslh Rb, Ra, I4]的图。

图 648 是说明命令 [vlsrh Rc, Ra, Rb] 的图。
图 649 是说明命令 [vlsrh Rb, Ra, I4] 的图。
图 650 是说明命令 [vrolh Rc, Ra, Rb] 的图。
图 651 是说明命令 [vrolh Rb, Ra, I4] 的图。
图 652 是说明命令 [vrroh Rb, Ra, I4] 的图。
图 653 是说明命令 [vasrh Rc, Ra, Rb] 的图。
图 654 是说明命令 [vaslb Rc, Ra, Rb] 的图。
图 655 是说明命令 [vaslb Rb, Ra, I3] 的图。
图 656 是说明命令 [vasrb Rc, Ra, Rb] 的图。
图 657 是说明命令 [vasrb Rb, Ra, I3] 的图。
图 658 是说明命令 [vlslb Rc, Ra, Rb] 的图。
图 659 是说明命令 [vlslb Rb, Ra, I3] 的图。
图 660 是说明命令 [vlsrb Rc, Ra, Rb] 的图。
图 661 是说明命令 [vlsrb Rb, Ra, I3] 的图。
图 662 是说明命令 [vrolb Rc, Ra, Rb] 的图。
图 663 是说明命令 [vrolb Rb, Ra, I3] 的图。
图 664 是说明命令 [vrorb Rb, Ra, I3] 的图。
图 665 是说明命令 [vasl Mm, Ra, Mn, Rb] 的图。
图 666 是说明命令 [vasl Mm, Rb, Mn, I5] 的图。
图 667 是说明命令 [vaslvw Mm, Ra, Mn, Rb] 的图。
图 668 是说明命令 [vaslvw Mm, Rb, Mn, I5] 的图。
图 669 是说明命令 [vasr Mm, Ra, Mn, Rb] 的图。
图 670 是说明命令 [vasr Mm, Rb, Mn, I5] 的图。
图 671 是说明命令 [vasrvw Mm, Ra, Mn, Rb] 的图。
图 672 是说明命令 [vls1 Mm, Ra, Mn, Rb] 的图。
图 673 是说明命令 [vls1 Mm, Rb, Mn, I5] 的图。
图 674 是说明命令 [vlsr Mm, Ra, Mn, Rb] 的图。

- 图 675 是说明命令 [vlsr Mm, Rb, Mn, I5] 的图。
- 图 676 是说明命令 [vsath Mm, Rb, Mn] 的图。
- 图 677 是说明命令 [vsath12 Rb, Ra] 的图。
- 图 678 是说明命令 [vsath9 Rb, Ra] 的图。
- 图 679 是说明命令 [vsath8 Rb, Ra] 的图。
- 图 680 是说明命令 [vsath8u Rb, Ra] 的图。
- 图 681 是说明命令 [vrndvh Rb, Mn] 的图。
- 图 682 是说明命令 [vabssumb Rc, Ra, Rb] 的图。
- 图 683 是说明命令 [vmul Mm, Rc, Ra, Rb] 的图。
- 图 684 是说明命令 [vxmul Mm, Rc, Ra, Rb] 的图。
- 图 685 是说明命令 [vhmul Mm, Rc, Ra, Rb] 的图。
- 图 686 是说明命令 [vlmul Mm, Rc, Ra, Rb] 的图。
- 图 687 是说明命令 [vfmulh Mm, Rc, Ra, Rb] 的图。
- 图 688 是说明命令 [vxfmulh Mm, Rc, Ra, Rb] 的图。
- 图 689 是说明命令 [vhfmulh Mm, Rc, Ra, Rb] 的图。
- 图 690 是说明命令 [vlfmulh Mm, Rc, Ra, Rb] 的图。
- 图 691 是说明命令 [vfmulhr Mm, Rc, Ra, Rb] 的图。
- 图 692 是说明命令 [vxfmulhr Mm, Rc, Ra, Rb] 的图。
- 图 693 是说明命令 [vhfmulhr Mm, Rc, Ra, Rb] 的图。
- 图 694 是说明命令 [vlfmulhr Mm, Rc, Ra, Rb] 的图。
- 图 695 是说明命令 [vfmulw Mm, Rc, Ra, Rb] 的图。
- 图 696 是说明命令 [vxfmulw Mm, Rc, Ra, Rb] 的图。
- 图 697 是说明命令 [vhfmulw Mm, Rc, Ra, Rb] 的图。
- 图 698 是说明命令 [vlfmulw Mm, Rc, Ra, Rb] 的图。
- 图 699 是说明命令 [vpfmulhww Mm, Rc:Rc+1, Ra, Rb] 的图。
- 图 700 是说明命令 [vmac Mm, Rc, Ra, Rb, Mn] 的图。
- 图 701 是说明命令 [vmac M0, Rc, Ra, Rb, Rx] 的图。

- 图 702 是说明命令 [vxmac Mm, Rc, Ra, Rb, Mn] 的图。
- 图 703 是说明命令 [vxmac M0, Rc, Ra, Rb, Rx] 的图。
- 图 704 是说明命令 [vhmac Mm, Rc, Ra, Rb, Mn] 的图。
- 图 705 是说明命令 [vhmac M0, Rc, Ra, Rb, Rx] 的图。
- 图 706 是说明命令 [vlmac Mm, Rc, Ra, Rb, Mn] 的图。
- 图 707 是说明命令 [vlmac M0, Rc, Ra, Rb, Rx] 的图。
- 图 708 是说明命令 [vfmach Mm, Rc, Ra, Rb, Mn] 的图。
- 图 709 是说明命令 [vfmach M0, Rc, Ra, Rb, Rx] 的图。
- 图 710 是说明命令 [vxfmach Mm, Rc, Ra, Rb, Mn] 的图。
- 图 711 是说明命令 [vxfmach M0, Rc, Ra, Rb, Rx] 的图。
- 图 712 是说明命令 [vhfmach Mm, Rc, Ra, Rb, Mn] 的图。
- 图 713 是说明命令 [vhfmach M0, Rc, Ra, Rb, Rx] 的图。
- 图 714 是说明命令 [vlfmach Mm, Rc, Ra, Rb, Mn] 的图。
- 图 715 是说明命令 [vlfmach M0, Rc, Ra, Rb, Rx] 的图。
- 图 716 是说明命令 [vfmachr Mm, Rc, Ra, Rb, Mn] 的图。
- 图 717 是说明命令 [vfmachr M0, Rc, Ra, Rb, Rx] 的图。
- 图 718 是说明命令 [vxfmachr Mm, Rc, Ra, Rb, Mn] 的图。
- 图 719 是说明命令 [vxfmachr M0, Rc, Ra, Rb, Rx] 的图。
- 图 720 是说明命令 [vhfmachr Mm, Rc, Ra, Rb, Mn] 的图。
- 图 721 是说明命令 [vhfmachr M0, Rc, Ra, Rb, Rx] 的图。
- 图 722 是说明命令 [vlfmachr Mm, Rc, Ra, Rb, Mn] 的图。
- 图 723 是说明命令 [vlfmachr M0, Rc, Ra, Rb, Rx] 的图。
- 图 724 是说明命令 [vfmacw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 725 是说明命令 [vxfmacw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 726 是说明命令 [vhfmacw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 727 是说明命令 [vlfmacw Mm, Rc, Ra, Rb, Mn] 的图。
- 图 728 是说明命令 [vpfmachww Mm, Rc:Rc+1, Ra, Rb, Mn] 的图。

图 729 是说明指令 [vmsu Mm, Rc, Ra, Rb, Mn] 的图。
图 730 是说明指令 [vmsu MO, Rc, Ra, Rb, Rx] 的图。
图 731 是说明指令 [vxmsu Mm, Rc, Ra, Rb, Mn] 的图。
图 732 是说明指令 [vxmsu MO, Rc, Ra, Rb, Rx] 的图。
图 733 是说明指令 [vhmsu Mm, Rc, Ra, Rb, Mn] 的图。
图 734 是说明指令 [vhmsu MO, Rc, Ra, Rb, Rx] 的图。
图 735 是说明指令 [vlmsu Mm, Rc, Ra, Rb, Mn] 的图。
图 736 是说明指令 [vlmsu MO, Rc, Ra, Rb, Rx] 的图。
图 737 是说明指令 [vfmsuh Mm, Rc, Ra, Rb, Mn] 的图。
图 738 是说明指令 [vfmsuh MO, Rc, Ra, Rb, Rx] 的图。
图 739 是说明指令 [vxfmsuh Mm, Rc, Ra, Rb, Mn] 的图。
图 740 是说明指令 [vxfmsuh MO, Rc, Ra, Rb, Rx] 的图。
图 741 是说明指令 [vhfmsuh Mm, Rc, Ra, Rb, Mn] 的图。
图 742 是说明指令 [vhfmsuh MO, Rc, Ra, Rb, Rx] 的图。
图 743 是说明指令 [vlfmsuh Mm, Rc, Ra, Rb, Mn] 的图。
图 744 是说明指令 [vlfmsuh MO, Rc, Ra, Rb, Rx] 的图。
图 745 是说明指令 [vfmsuw Mm, Rc, Ra, Rb, Mn] 的图。
图 746 是说明指令 [vxfmsuw Mm, Rc, Ra, Rb, Mn] 的图。
图 747 是说明指令 [vhfmsuw Mm, Rc, Ra, Rb, Mn] 的图。
图 748 是说明指令 [vlfmsuw Mm, Rc, Ra, Rb, Mn] 的图。

具体实施方式

说明本发明处理器的结构。本处理器的指令与通常的微型计算机相比，排列性高，是将 AV 媒体类信号处理技术领域作为目标来开发的通用处理器。通过在便携电话、移动 AV 设备、数字 TV、DVD 等中使用共同芯片，可提高软件再利用性。另外，本处理器可以高性能、高性价比实现许多媒体处理，并且，提供将提高开发效率为目的的高级语言开发环境。

图 1 是本处理器的示意框图。本处理器 1 包括命令控制部 10、解码部 20、寄存器文件 30、运算部 40、I/F 部 50、命令存储部 60、数据存储部 70、扩展寄存器部 80 和 I/O 接口部 90。运算部 40 由执行 SIMD 型命令运算的算术逻辑、比较运算器 41-43、乘法、积和运算器 44、桶形移位器 45、除法器 46 及变换器 47 构成。乘法、积和运算器 44 以最长 65 比特累计，以便不使比特精度下降。另外，乘法、积和运算器 44 与算术逻辑、比较运算器 41-43 一样，可执行 SIMD 型命令。另外，处理器 1 最大可并行执行 3 个算术逻辑、比较运算命令。

图 2 是算术逻辑、比较运算器 41-43 的示意图。算术逻辑、比较运算器 41-43 分别由 ALU 部 41a、饱和处理部 41b、及标志部 41c 构成。ALU 部 41a 由算术运算器、逻辑运算器、比较器、TST 器构成。对应的运算数据的位宽度为 8 位（并排使用 4 个运算器）、16 比特（并排使用 2 个运算器）、32 比特（所有运算器进行 32 比特数据处理）。由标志部 41c 等对算术运算结果进行溢出检测和条件标志生成。各运算器、比较器、TST 器的结果，进行算术右移位、饱和处理部 41b 的饱和、最大、最小值检测、绝对值生成处理。

图 3 是表示桶形移位器(バレルシフト)45 构成的框图。桶形移位器 45 由选择器 45a、45b、上位桶形移位器 45c、下位桶形移位器 45d 及饱和处理部 45e 构成，执行数据的算术移位(shift)（2 的补数体系的移位）或逻辑移位（无符号移位）。通常，以 32 比特或 64 比特的数据作为输入输出。由其它寄存器或当前值对存储在寄存器 32a、32b 中的被移位数据指定移位量。对数据进行左 63 比特-右 63 比特的算术或逻辑移位，以输入比特长度进行输出。

另外，桶形移位器 45 可对 SIMD 型命令移位 8、16、32、64 比特的数据。例如，可并排处理 4 个 8 位数据的移位。

算术移位是 2 的补码体系的移位，用于加法或减法时的小数点位置的一致或 2 的指数乘法（2、2 的 2 次方、2 的 (-1) 次方、2 的 (-2)

次方等)。

图 4 是表示变换器 47 构成的框图。变换器 47 由饱和功能块(SAT) 47a、BSEQ 功能块 47b、MSKGEN 功能块 47c、VSUMB 功能块 47d、BCNT 功能块 47e、IL 功能块 47f 构成。

饱和功能块 (SAT) 47a 对输入数据进行饱和处理。通过具有两个饱和处理 32 比特数据的功能块，支持两排的 SIMD 型命令。

BSEQ 功能块 47b 从 MSB 开始计数连续的 0 或 1。

MSKGEN 功能块 47c 将指定比特区间输出为 1，将此外的区间输出为 0。

VSUMB 功能块 47d 将输入数据区分成指定的位宽度，输出其总和。

BCNT 功能块 47e 计数输入数据中变为 1 的比特数。

IL 功能块 47f 将输入数据区分成指定的位宽度，输出替换各数据块的值。

图 5 是表示除法器 46 构成的框图。除法器 46 将被除数作为 64 比特，将除数作为 32 比特，以每 32 比特输出商和余数。在求出商和余数之前，必需循环 34 次。可处理有符号、无符号两种数据。有无符号的设定对被除数和除数一样。另外，具有输出溢出标志、0 除法标志的功能。

图 6 是表示乘法、积和运算器 44 构成的框图。乘法、积和运算器 44 由两个 32 位乘法器 (MUL) 44a、44b、3 个 64 位加法器 (Adder) 44c-44e、选择器 44f 及饱和处理部 (Saturation) 44g 构成，进行如下的乘法、积和运算。

- 32×32 比特的 signed 的乘法、积和、积差运算
- 32×32 比特的 unsigned 的乘法
- 16×16 比特的 2 排 signed 的乘法、积和、积差运算
- 32×16 比特的 2 排 signed 的乘法、积和、积差运算

对整数、固定小数点格式 (h1、h2、w1、w2) 的数据进行这些运

算。另外，对这些运算进行舍入、饱和。

图 7 是表示命令控制部 10 构成的框图。命令控制部 10 由命令高速缓冲存储器 10a、地址管理部 10b、命令缓冲器 10c-10e、转移缓冲器 10f 及旋转部 (rotation) 10g 构成，在平常和分支时提供命令。通过具有 3 个 128 位的命令缓冲器 (命令缓冲器 10c-10e)，对应于最大排列执行数。就分支处理而言，在分支执行前，经转移缓冲器 10f 等，事先将分支目的地地址存储在后述的 TAR 寄存器中 (settar 命令)。使用存储在 TAR 寄存器中的分支目的地地址进行分支。

本处理器 1 是具有 VLIW 结构的处理器。这里，所谓 VLIW 结构是在 1 个指令字中存储多个命令 (加载、存储、运算、分支等)，并同时执行所有这些命令的结构。编程器通过将可并排执行的命令记述为 1 个发行组，可并排处理该发行组。在本说明书中，用 “; ;” 表示发行组的区分。下面示出表述例。

(例 1)

```
mov r1, 0x223; ;mov
```

该命令记述表示仅执行命令 mov。

(例 2)

```
mov r1, 0x38
```

```
add r0, r1, r2
```

```
sub r3, r1, r2; ;
```

这些命令记述表示并列执行 3 个命令 mov、add、sub。

命令控制部 10 识别发行组，发送给解码部 20。解码部 20 分析发行组的命令，控制必要的资源。

下面，说明本处理器 1 具备的寄存器。

本处理器 1 的寄存器组如下表 1 所示。

表 1

另外，本处理器 1 的标志组 (由后述条件标志寄存器等管理的标

志)如下表 2 所示。

表 2

图 8 是表示通用寄存器 (R0-R31) 30a 构造的图。通用寄存器 (R0-R31) 30a 构成作为执行对象的任务的上下文 (context) 的一部分, 是存储数据或地址的 32 比特的寄存器群。通常寄存器 R30 和 R31 分别作为全程 (global) 指示器、堆栈指示器, 硬件使用。

图 9 是表示连接寄存器 (LR) 30c 构造的图。与该连接寄存器 (LR) 30c 关联, 本处理器 1 还具备未图示的保存寄存器 (SVR)。连接寄存器 (LR) 30c 是存储函数调用时的返回地址的 32 比特的寄存器。保存寄存器 (SVR) 是保存函数调用时的条件标志寄存器的条件标志 (CFR.CF) 的 16 比特的寄存器。连接寄存器 (LR) 30c 与后述的分支寄存器 (TAR) 一样, 也用于循环高速化中。下位 1 位总是读取 0, 但在写入时必需写入 0。

例如, 在执行 `call(brl, jmp1)` 命令的情况下, 本处理器 1 返回连接寄存器 (LR) 30c, 保存地址, 将条件标志 (CFR.CF) 保存在保存寄存器 (SVR) 中。另外, 在执行 `jmp` 命令的情况下, 从连接寄存器 (LR) 30c 中取出返回地址 (分支目的地地址), 使程序计数器 (PC) 返回。另外, 在执行 `ret(jmpr)` 命令的情况下, 从连接寄存器 (LR) 30c 中取出分支目的地地址 (返回地址), 存储 (返回) 到程序计数器 (PC) 中。另外, 从保存寄存器 (SVR) 中取出条件标志, 存储 (返回) 到条件标志寄存器 (CFR) 32 的条件标志区域 CFR.CF 中。

图 10 是表示分支寄存器 (TAR) 30d 构造的图。分支寄存器 (TAR) 30d 是存储分支目标地址的 32 位的寄存器。主要用于回路的高速化。下位 1 位总是读取 0, 但在写入时必需写入 0。

例如, 在执行 `jmp, jloop` 命令的情况下, 本处理器 1 从分支寄存器 (TAR) 30d 中取出分支目的地地址, 存储在程序计数器 (PC) 中。在存储在分支寄存器 (TAR) 30d 中的地址命令被存储在分支用命令

缓冲器中的情况下，分支损失 (penalty) 变为 0。通过将回路的开头地址存储在分支寄存器 (TAR) 30d 中，可高速化回路。

图 11 是表示程序状态寄存器 (PSR) 31 构造的图。程序状态寄存器 (PSR) 31 构成作为执行对象的任务上下文的一部分，是存储如下示出的处理器状态信息的 32 比特的寄存器。

比特 SWE: 表示 VMP (Virtual Multi-Processor) 的 LP (Logical Processor) 切换激活。[0]表示不允许 LP 切换，[1]表示允许 LP 切换。

比特 FXP: 表示固定小数点模式。[0]表示模式 0，[1]表示模式 1。

比特 IH: 是中断处理标志，表示在可屏蔽 (maskable) 的中断处理中。[1]表示中断处理中，[0]表示不在中断处理中。若发生中断，则自动设置。为了知道通过 rti 命令从中断后到返回，是到其它中断处理中还是到程序处理中而使用该标志。

比特 EH: 是表示错误或 NMI 在处理中的标志。[0]表示不在错误/NMI 中断处理中，[1]表示错误/NMI 中断处理中。EH=1 时，在发生非同步错误或 NMI 的情况下，进行屏蔽。另外，在激活 VMP 时，屏蔽 VMP 的板极切换。

比特 PL[1: 0]: 表示特权电平。[00]表示特权电平 0，即，表示处理器抽象电平，[01]表示特权电平 1 (不能设定)，[10]表示特权电平 2，即，系统程序电平，[11]表示特权电平 3，即用户程序电平。

比特 LPIE3: 表示 LP 固有中断 3 激活。[1]表示允许中断，[0]表示不允许中断。

比特 LPIE2: 表示 LP 固有中断 2 激活。[1]表示允许中断，[0]表示不允许中断。

比特 LPIE1: 表示 LP 固有中断 1 激活。[1]表示允许中断，[0]表示不允许中断。

比特 LPIE0: 表示 LP 固有中断 0 激活。[1]表示允许中断, [0]表示不允许中断。

比特 AEE: 表示未对齐 (misalignment) 例外激活。[1]表示允许未对齐例外, [0]表示不允许未对齐例外。

比特 IE: 表示电平中断激活。[1]表示允许电平中断, [0]表示不允许电平中断。

比特 IM[7: 0]: 表示中断屏蔽。可定义到电平 0-7, 由各电平进行屏蔽。电平 0 是最高的电平。处理器 1 仅受理未由 IM 屏蔽的中断请求中具有最高电平的中断请求。一旦受理中断请求, 则硬件自动屏蔽小于受理电平的电平。IM[0]是电平 0 的屏蔽, IM[1]是电平 1 的屏蔽, IM[2]是电平 2 的屏蔽, IM[3]是电平 3 的屏蔽, IM[4]是电平 4 的屏蔽, IM[5]是电平 5 的屏蔽, IM[6]是电平 6 的屏蔽, IM[7]是电平 7 的屏蔽。

reserved: 表示预约比特。总是读取 0。在写入时必需写入 0。

图 12 是表示条件标志寄存器 (CFR) 32 构造的图。条件标志寄存器 (CFR) 32 是构成作为执行对象的任务上下文的一部分的 32 比特的寄存器, 由条件标志 (condition 标志)、运算标志 (operation 标志)、矢量条件标志 (矢量 condition 标志)、运算命令用位位置指定字段、SIMD 数据调整 (data alignment) 信息字段构成。

比特 ALN[1: 0]: 表示调整 (align) 模式。设定 valnvc 命令的调整模式。

比特 BP0[4: 0]: 表示位位置。由必需指定位位置的命令使用。

比特 VC0-VC3: 是矢量条件标志。从 LSB 侧的字节或半字开始依次对应于 VC0, MSB 侧对应于 VC3。

比特 OVS: 是溢出标志 (汇总)。在发生饱和或检测溢出中进行设置。在未检测的情况下, 保持命令执行前的值。必需由软件进行清除。

比特 CAS：进位标志（汇总）。由 addc 命令发生进位、或由 subc 命令发生借位的情况下，进行设置。在 addc 命令未发生进位、或 subc 命令未发生借位的情况下，保持命令执行前的值。必需由软件进行清除。

比特 C0-C7：是条件标志。标志 C7 的值总为 1。忽视 FALSE 条件对标志 C7 的反映（写入 0）。

reserved：表示预约比特。总是读取 0。在写入时必需写入 0。

图 13 是表示累加器（M0、M1）30b 构造的图。累加器（M0、M1）30b 构成作为执行对象任务的上下文一部分，并由在图 13（a）中示出的 32 比特寄存器 MH0-MH1（乘法、积和用寄存器（上位 32 比特））、和图 13（b）中示出的 32 比特寄存器 ML0-ML1 乘法、积和用寄存器（下位 32 比特）构成。

寄存器 MH0-MH1 用于在乘法命令中存储结果的上位 32 比特。在积和命令中用作累加器的上位 32 比特。另外，在处理比特流的情况下，可与通用寄存器组合来使用。寄存器 ML0-ML1 用于在乘法命令中存储结果的下位 32 比特。在积和命令中用作累加器的下位 32 比特。

图 14 是表示程序计数器（PC）33 构造的图。程序计数器（PC）33 构成作为执行对象的任务上下文的一部分，是保持执行中命令地址的 32 比特的计数器。

图 15 是表示 PC 保存用寄存器（IPC）34 构造的图。PC 保存用寄存器（IPC）34 是构成作为执行对象的任务上下文一部分的 32 比特的寄存器。

图 16 是表示 PSR 保存用寄存器（IPSR）35 构造的图。PSR 保存用寄存器（IPSR）35 构成作为执行对象的任务上下文一部分，是保存程序状态寄存器（PSR）31 用的 32 比特的寄存器，对应于程序状态寄存器（PSR）31 的预约比特的部分总是读取 0，但在写入时必需写入 0。

下面，说明本处理器 1 的存储器空间。在本处理器 1 中，将 4GB 的线性存储器空间分割 32 份，对 128MB 单位的空间分配命令 SRAM (Static RAM) 和数据 SRAM。将该 128MB 的空间作为 1 个块，设定想访问 SAR (SRAM Area Register) 的块。在访问的地址是由 SAR 设定的空间的情况下，直接对命令 SRAM/数据 SRAM 进行访问，但在不是由 SAR 设定的空间的情况下，对总线控制器 (BCU) 发出访问请求。在 BCU 上连接单片存储器 (OCM)、外部存储器、外部设备、I/O 端子等，可对这些设备进行读写。

图 17 是表示本处理器 1 的管道动作的时间图。如图 17 所示，本处理器 1 基本上由命令取出、命令分配 (分派)、解码、执行、写入 5 段管道构成。

图 18 是表示本处理器 1 执行命令时各管道动作的时间图。在命令取出阶段，访问由程序计数器 (PC) 33 指定地址的命令存储器，将命令传送给命令缓冲器 10c-10e 等。在命令分配阶段，对分支类命令输出分支目的地地址信息，输出输入寄存器控制信号，分配可变长命令，将命令传送到指令寄存器 (IR)。在解码阶段，将 IR 输入解码部 20，输出运算器控制信号、存储器访问信号。在执行阶段，执行运算，将运算结果输出到数据存储器或通用寄存器 (R0-R31) 30a。在写入阶段，将数据传送、运算结果存储在通用寄存器中。

处理器 1 通过 VLIW 结构，最高可并排进行 3 个上述处理。因此，在图 18 所示动作中，处理器 1 在图 19 的定时下并排执行。

下面，说明如上构成的处理器 1 的命令组。

下表 3-表 5 是将处理器 1 执行的命令分类成不同类别的表。

表 3

表 4

表 5

表中的[运算器]表示该命令使用的运算器。运算器的略号含义如下。即，[A]表示 ALU 命令，[B]表示分支命令，[C]表示变换命令，[DIV]表示除法命令，[DBGM]表示调试命令，[M]表示存储器访问命令，[S1]、[S2]表示移位命令，[X1]、[X2]表示乘法命令。

图 20 是表示本处理器 1 执行的命令格式的图。

图中略号的含义如下。即，[P]表示判定 (predicate) (执行条件：指定 8 个条件标志 C0-C7 之一)，[OP]表示操作码字段，[R]表示寄存器字段，[I]表示当前值字段，[D]表示置换字段 (ディスプースメント)。

图 21-图 36 是说明处理器 1 执行的命令的示意功能图。即，图 21 是说明属于分类[ALUadd (加法)类]的命令的图，图 22 是说明属于分类[ALUsub (减法)类]的命令的图，图 23 是说明属于分类[ALUlogic (逻辑运算)类]的命令的图，图 24 是说明属于分类[CM (比较运算)类]的命令的图，图 25 是说明属于分类[mul (乘法)类]的命令的图，图 26 是说明属于分类[mac (积和运算)类]的命令的图，图 27 是说明属于分类[msu (积差运算)类]的命令的图，图 28 是说明属于分类[MEMld (存储器读取)类]的命令的图，图 29 是说明属于分类[MEMstore (存储器写出)类]的命令的图，图 30 是说明属于分类[BRA (分支)类]的命令的图，图 31 是说明属于分类[BSasl (算术桶形移位 (barrel shift))类]的命令的图，图 32 是说明属于分类[BSlsr (逻辑桶形移位)类]的命令的图，图 33 是说明属于分类[CNVvaln (算术变换)类]的命令的图，图 34 是说明属于分类[CNV (一般变换)类]的命令的图，图 35 是说明属于分类[SATv1pk (饱和处理)类]的命令的图，图 36 是说明属于分类[ETC (其他)类]的命令的图。

在这些图中，项目[SIMD]表示该命令类型 (SISD (SINGLE) 或 SIMD 的区别)，项目[尺寸]表示作为运算对象的操作数的尺寸

(size), 项目[命令]表示该命令的操作码, 项目[操作数]表示该命令的操作数, 项目[CFR]表示条件标志寄存器的变化, 项目[PSR]表示处理器状态寄存器的变化, 项目[代表动作]表示动作示意, 项目[运算器]表示使用的运算器, 项目[3116]表示命令尺寸。

图 37-图 748 是说明本处理器 1 执行的命令的详细功能的图。命令说明中使用的各种记号的含义如下表 6-表 10。

表 6

表 7

表 8

表 9

表 10

图 37-图 119 是说明[load]关联命令的图。

图 120-图 184 是说明[store]关联命令的图。

图 185-图 186 是说明[memory(etc)]关联命令的图。

图 187-图 206 是说明[external register]关联命令的图。

图 207-图 247 是说明[branch]关联命令的图。

图 248-图 264 是说明[VMP/interrupt]关联命令的图。

图 265-图 258 是说明[program interrupt]关联命令的图。

图 259-图 303 是说明[arithmetic]关联命令的图。

图 304-图 317 是说明[logic]关联命令的图。

图 318-图 359 是说明[compare]关联命令的图。

图 360-图 420 是说明[move]关联命令的图。

图 421 是说明[nop]关联命令的图。

图 422-图 441 是说明[shift(S1)]关联命令的图。

图 442-图 460 是说明[shift(S2)]关联命令的图。

图 461-图 470 是说明[extract]关联命令的图。

图 471-图 474 是说明[mask]关联命令的图。

图 475-图 480 是说明[shaturation]关联命令的图。
图 481-图 512 是说明[conversion]关联命令的图。
图 513-图 516 是说明[bit count]关联命令的图。
图 517-图 520 是说明[etc]关联命令的图。
图 521-图 526 是说明[mul (X1)]关联命令的图。
图 527-图 531 是说明[mul (X2)]关联命令的图。
图 532-图 543 是说明[mac (X1)]关联命令的图。
图 544-图 547 是说明[mac (X2)]关联命令的图。
图 548-图 559 是说明[msu (X1)]关联命令的图。
图 560-图 563 是说明[msu (X2)]关联命令的图。
图 564-图 565 是说明[divide]关联命令的图。
图 566-图 569 是说明[debug]关联命令的图。
图 570-图 615 是说明[SIMD arithmetic]关联命令的图。
图 616-图 639 是说明[SIMD compare]关联命令的图。
图 640-图 664 是说明[SIMD shift (S1)]关联命令的图。
图 665-图 675 是说明[SIMD shift (S2)]关联命令的图。
图 676-图 680 是说明[SIMD saturation]关联命令的图。
图 681-图 682 是说明[SIMD etc]关联命令的图。
图 683-图 699 是说明[SIMD mul (X2)]关联命令的图。
图 700-图 728 是说明[SIMD mac (X2)]关联命令的图。
图 729-图 748 是说明[SIMD msu (X2)]关联命令的图。

下面，说明本处理器 1 在几个特征命令中的动作。

(1) 交叉操作数来进行 SIMD2 项运算的命令

首先，说明在两排 SIMD 运算中，将位于交叉位置关系上的操作数作为对象进行运算的命令。

[命令 vxaddh]

命令 vxaddh 是以半字（16 比特）单位将位于交叉位置关系上的

两组操作数相加的 SIMD 型命令。例如，若为

`vxaddh Rc, Ra, Rb`，则处理器 1 通过算术逻辑、比较运算器 41，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相加，将结果存储在寄存器 Rc 的上位 16 位中，与之并行，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相加，将结果存储在寄存器 Rc 的下位 16 位中。

这种命令在为了由对称型滤波器（距中心左右对称的系数）减去乘法，而将事先乘以相同系数的值彼此相加（或相减）的情况下有效。

另外，本处理器 1 即使对减法命令（`vxsubh` 等）也执行与该加法命令一样的处理。

[命令 `vxmul`]

命令 `vxmul` 是以半字（16 比特）单位将位于交叉位置关系上的两组操作数相乘、并分别保留（SIMD 存储）这些结果的下位半字的 SIMD 型命令。例如，若为

`vxmul Rc, Ra, Rb`，则处理器 1 通过算术逻辑、比较运算器 41，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的上位 16 位和运算寄存器 MLm 的上位 16 位中，同时，将该乘法结果的下位 16 位存储在寄存器 Rc 的上位 16 位中，与之并行，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的下位 16 位和运算寄存器 MLm 的下位 16 位中，同时，将该乘法结果的下位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令对复数的内积有效。取出结果的下位在处理整数数据的情况下有效（主要是图像）。

[命令 `vxfmulh`]

命令 `vxfmulh` 是以半字（16 比特）单位将位于交叉位置关系上

的两组操作数相乘、并分别保留（SIMD 存储）这些结果的上位半字的 SIMD 型命令。例如，若为

`vxfmulh Rc, Ra, Rb`，则处理器 1 通过乘法、积和运算器 44，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的上位 16 位和运算寄存器 MLm 的上位 16 位中，同时，将该乘法结果的下位 16 位存储在寄存器 Rc 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的下位 16 位和运算寄存器 MLm 的下位 16 位中，同时，将该乘法结果的上位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令对复数的内积有效。取出结果的上位在处理固定小数点数据的情况下有效。可适用于称为 Q31/Q15 的标准格式（MSB 装入）中。

[命令 `vxfmulw`]

命令 `vxfmulw` 是以半字（16 比特）单位将位于交叉位置关系上的两组操作数相乘、并仅保留（非 SIMD 存储）一方乘法结果的 SIMD 型命令。例如，若为

`vxfmulw Rc, Ra, Rb`，则处理器 1 通过乘法、积和运算器 44，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的上位 16 位和运算寄存器 MLm 的上位 16 位中，同时，将该乘法结果（字）存储在寄存器 Rc 中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的下位 16 位和运算寄存器 MLm 的下位 16 位中（不存储在寄存器 Rc 中）。

这种命令在精度不足 16 比特、不能维持 SIMD 的情况下有效（音

频等)。

[命令 vxmac]

命令 vxmac 是以半字 (16 比特) 单位进行位于交叉位置关系上的两组操作数的积和运算、并分别保留 (SIMD 存储) 这些结果的下位半字的 SIMD 型命令。例如, 若为

vxmac Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘, 将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相加, 将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中, 同时, 将该加法结果的下位 16 位存储在寄存器 Rc 的上位 16 位中, 与之并行,

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加, 将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中, 同时, 将该加法结果的下位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令对复数的内积有效。取出结果的下位在处理整数数据的情况下有效 (主要是图像)。

[命令 vxfmach]

命令 vxfmach 是以半字 (16 比特) 单位进行位于交叉位置关系上的两组操作数的积和运算、并分别保留 (SIMD 存储) 这些结果的上位半字的 SIMD 型命令。例如, 若为

vxfmach Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘, 将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相

加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中，同时，将该加法结果的上位 16 位存储在寄存器 Rc 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的下位 16 位构成的 32 位区域中，同时，将该加法结果的上位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令对复数的内积有效。取出结果的上位在处理固定小数点数据的情况下有效。可适用于称为 Q31/Q15 的标准格式（MSB 装入）中。

[命令 vxfmaw]

命令 vxfmaw 是以半字（16 比特）单位，将位于交叉位置关系上的两组操作数相乘、并仅保留（非 SIMD 存储）一方乘法结果的 SIMD 型命令。例如，若为

vxfmaw Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的下位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中，同时，将该加法结果的 32 比特存储在寄存器 Rc 中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的下位 16 位构成的 32 位区域中（不存储在寄存器 Rc 中）。

这种命令在精度不足 16 比特、不能维持 SIMD 的情况下有效（音

频等)。

另外,本处理器 1 即使对积差命令 (vxmsu、vxmsuh、vxmsuw 等) 也执行与这些积和命令一样的处理。

另外,本处理器 1 也可具备不仅执行对以上处于交叉位置关系的两组操作数的运算 (两排 SIMD 的加法、减法、乘法、积和、积差)、而且还执行对扩展其的 n 组操作数的运算 (4 排、8 排 SIMD 运算等) 的功能。

例如,在设寄存器 Ra 中存储的 4 个字节数据从上位开始为 Ra1、Ra2、Ra3、Ra4, 设寄存器 Rb 中存储的 4 个字节数据从上位开始为 Rb1、Rb2、Rb3、Rb4 的情况下, 作为将寄存器 Ra 与寄存器 Rb 作为操作数的 SIMD 运算命令, 也可具备对处于以下交叉位置关系上的字节数据并行执行运算的命令。即

(i) 1 对称交叉命令

也可具备分别将 Ra1 与 Rb4、Ra2 与 Rb3、Ra3 与 Rb2、Ra4 与 Rb1 作为运算对象的 4 排 SIMD 型运算命令。

(ii) 2 对称交叉命令

也可具备分别将 Ra1 与 Rb2、Ra2 与 Rb1、Ra3 与 Rb4、Ra4 与 Rb3 作为运算对象的 4 排 SIMD 型运算命令。

(iii) 双重交叉命令

也可具备分别将 Ra1 与 Rb3、Ra2 与 Rb4、Ra3 与 Rb1、Ra4 与 Rb2 作为运算对象的 4 排 SIMD 型运算命令。

另外,这些 4 排 SIMD 型运算的种类与上述 2 排 SIMD 型运算一样, 也可适用于加法、减法、乘法、积和、积差之一。并且,对乘法、积和、积差而言,与上述 2 排 SIMD 型运算命令 (vxmul、vxfmulh、vxfmulw 等) 一样, 也可具备仅将 4 个运算结果各自的下位字节 SIMD 存储在寄存器 Rc 等中、又仅将 4 个运算结果各自的上位字节 SIMD 存储在寄存器 Rc 等中、又仅将 4 个运算结果中的两个 SIMD 存储在寄存器 Rc

等中的命令。

对处于上述交叉位置关系的数据的 3 种运算一般化时，可表现如下。即，若设作为运算对象的数据组为构成第 1 数据群的 n 个数据排列中的第 i 个数据和构成第 2 数据群的 n 个数据排列中的第 j 个数据时，则

在上述 (i) 1 对称交叉命令中， $j=n-i+1$ 成立，

在上述 (ii) 2 对称交叉命令中， $j=i-(-1)^{(i \bmod 2)}$ 成立，

在上述 (iii) 双重交叉命令中， $j=n-i+1+(-1)^{(i \bmod 2)}$ 。

这里， $[\]$ 表示指数， $[\text{mod}]$ 表示余数。

以上命令在复数的内积等同时运算两个复数的情况下有效。

(2) 固定一方操作数来进行 SIMD2 项运算的命令

接着，说明在两排 SIMD 运算中，固定一方操作数来进行（设一方的操作数为共同的）运算命令。

[命令 vhaddh]

命令 vhaddh 是以半字（16 比特）单位将设一方操作数（寄存器中的上位 16 位）为共同的两组操作数相加的 SIMD 型命令。例如，若为

vhaddh Rc, Ra, Rb, 则处理器 1 通过算术逻辑、比较运算器 41,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相加，将结果存储在寄存器 Rc 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相加，将结果存储在寄存器 Rc 的下位 16 位中。

这种命令在由于两个排列的要素间的加减法、使两个排列的对齐错位、难以适用 SIMD 的情况下有效。

另外，本处理器 1 即使对减法命令（vhsubh 等）也执行与该加法命令一样的处理。

[命令 vhmul]

命令 `vhmul` 是以半字（16 比特）单位将设一方操作数（寄存器中的上位 16 位）为共同的两组操作数相乘、并保留（SIMD 存储）这些结果的各下位半字的 SIMD 型命令。例如，若为

`vhmul Rc, Ra, Rb`, 则处理器 1 通过乘法、积和运算器 44 等,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果存储在运算寄存器 M_{Hm} 的上位 16 位和运算寄存器 M_{Lm} 的上位 16 位中, 同时, 将该乘法结果的下位 16 位存储在寄存器 Rc 的上位 16 位中, 同时,

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果存储在运算寄存器 M_{Hm} 的下位 16 位和运算寄存器 M_{Lm} 的下位 16 位中, 同时, 将该乘法结果的下位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令在增益控制等向所有要素乘以系数的情况下, 通过迭代展开循环, 并由 SIMD 进行并列处理时, 要素错位而难以适用 SIMD 的情况下有效。基本上, 与下述下位固定命令成对使用。取出结果的下位在处理整数数据的情况下有效（主要是图像）。

[命令 `vhfmulh`]

命令 `vhfmulh` 是以半字（16 比特）单位将设一方操作数（寄存器中的上位 16 位）为共同的两组操作数相乘、并保留（SIMD 存储）这些结果的各 H 位半字的 SIMD 型命令。例如，若为

`vhfmulh Rc, Ra, Rb`, 则处理器 1 通过乘法、积和运算器 44 等,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果存储在运算寄存器 M_{Hm} 的上位 16 位和运算寄存器 M_{Lm} 的上位 16 位中, 同时, 将该乘法结果的上位 16 位存储在寄存器 Rc 的上位 16 位中, 与之并行,

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果存储在运算寄存器 M_{Hm} 的下位 16 位和运算寄存器 M_{Lm}

的下位 16 位中，同时，将该乘法结果的上位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令与上述一样有效。取出结果的上位在处理固定小数点数据的情况下有效。可适用于称为 Q31/Q15 的标准格式（MSB 装入）中。

[命令 vhfmulw]

命令 vhfmulw 是以半字（16 比特）单位将设一方操作数（寄存器中的上位 16 位）为共同的两组操作数相乘、并仅保留（非 SIMD 存储）一方乘法结果的 SIMD 型命令。例如，若为

vhfmulw Rc, Ra, Rb, 则处理器 1 通过乘法、积和运算器 44 等，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的上位 16 位和运算寄存器 MLm 的上位 16 位中，同时，将该乘法结果（字）存储在寄存器 Rc 中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果存储在运算寄存器 MHm 的下位 16 位和运算寄存器 MLm 的下位 16 位中（不存储在寄存器 Rc 中）。

这种命令对确保精度有效。

[命令 vhmacc]

命令 vhmacc 是以半字（16 比特）单位进行设一方操作数（寄存器中的上位 16 位）为共同的两组操作数的积和运算、并分别保留（SIMD 存储）这些结果的下位半字的 SIMD 型命令。例如，若为

vhmacc Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44，

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中，同时，将该加法结果的下位 16 位存储在寄存

器 Rc 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的下位 16 位构成的 32 位区域中，同时，将该加法结果的下位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令在由 FIR（滤波器）等通过迭代展开循环，并由 SIMD 进行并列处理时，要素错位而难以适用 SIMD 的情况下有效。基本上，与下述下位固定命令成对（交互）使用。取出结果的下位在处理整数数据的情况下有效（主要是图像）。

[命令 vhfmach]

命令 vhfmach 是以半字（16 比特）单位进行设一方操作数（寄存器中的上位 16 位）为共同的两组操作数的积和运算、并分别保留（SIMD 存储）这些结果的上位半字的 SIMD 型命令。例如，若为

vhfmach Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中，同时，将该加法结果的上位 16 位存储在寄存器 Rc 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘，将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加，将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的下位 16 位构成的 32 位区域中，同时，将该加法结果的上位 16 位存储在寄存器 Rc 的下位 16 位中。

这种命令与上述一样有效。取出结果的上位在处理固定小数点数

据的情况下有效。可适用于称为 Q31/Q15 的标准格式 (MSB 装入) 中。

[命令 vhfmacw]

命令 vhfmacw 是以半字 (16 比特) 单位将设一方操作数 (寄存器中的上位 16 位) 为共同的两组操作数相乘、并仅保留 (非 SIMD 存储) 一方乘法结果的 SIMD 型命令。例如, 若为

vhfmacw Mm, Rc, Ra, Rb, Mn, 则处理器 1 通过乘法、积和运算器 44,

(i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果与由运算寄存器 MHn 及 MLn 的上位 16 位构成的 32 位相加, 将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的上位 16 位构成的 32 位区域中, 同时, 将该加法结果的 32 比特存储在寄存器 Rc 中, 同时,

(ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相乘, 将该乘法结果与由运算寄存器 MHn 及 MLn 的下位 16 位构成的 32 位相加, 将加法结果的 32 比特存储在由运算寄存器 MHm 及 MLm 的下位 16 位构成的 32 位区域中 (不存储在寄存器 Rc 中)。

这种命令对确保精度有效。

另外, 本处理器 1 即使对积差命令 (vhmsu、vhmsuh、vhmsuw 等) 也执行与这些积和命令一样的处理。

另外, 在这些命令中, 固定 (共同化) 寄存器中的上位 16 位, 但即使对固定 (共同化) 寄存器中下位 16 位的命令 (vladdh, vlsubh, vlmul, vlfmulh, vlfmulw, vlmac, vlmsu, vlfmach, vlmsuh, vlfmacw, vlmsuw 等), 本处理器 1 也可执行与这些运算命令一样的处理。这种命令与上述上位固定命令成对使用时有效。

另外, 本处理器 1 也可具备不仅执行对以上设一方操作数 (寄存器中的上位 16 位) 共同的两组操作数的运算 (两排 SIMD 的加法、减法、乘法、积和、积差)、而且还执行对扩展其的 n 组操作数的运算

(4排、8排 SIMD 运算等)的功能。

例如,在设寄存器 Ra 中存储的 4 个字节数据从上位开始为 Ra1、Ra2、Ra3、Ra4,设寄存器 Rb 中存储的 4 个字节数据从上位开始为 Rb1、Rb2、Rb3、Rb4 的情况下,作为将寄存器 Ra 与寄存器 Rb 作为操作数的 SIMD 运算命令,也可具备对以下设一方操作数(寄存器中的 1 字节)共同的字节数据并行执行运算的命令。即

(i) 最上位固定命令

也可具备分别将 Ra1 与 Rb1、Ra2 与 Rb1、Ra3 与 Rb1、Ra4 与 Rb1 作为运算对象的 4 排 SIMD 型运算命令。

(ii) 中上位固定命令

也可具备分别将 Ra1 与 Rb2、Ra2 与 Rb2、Ra3 与 Rb2、Ra4 与 Rb2 作为运算对象的 4 排 SIMD 型运算命令。

(iii) 中下位固定命令

也可具备分别将 Ra1 与 Rb3、Ra2 与 Rb3、Ra3 与 Rb3、Ra4 与 Rb3 作为运算对象的 4 排 SIMD 型运算命令。

(iv) 最下位固定命令

也可具备分别将 Ra1 与 Rb4、Ra2 与 Rb4、Ra3 与 Rb4、Ra4 与 Rb4 作为运算对象的 4 排 SIMD 型运算命令。

另外,这些 4 排 SIMD 型运算的种类与上述 2 排 SIMD 型运算一样,也可适用于加法、减法、乘法、积和、积差之一。并且,对乘法、积和、积差而言,与上述 2 排 SIMD 型运算命令(vhmul、vhfmulh、vhfmulw 等)一样,也可具备仅将 4 个运算结果各自的下位字节 SIMD 存储在寄存器 Rc 等中、又仅将 4 个运算结果各自的上位字节 SIMD 存储在寄存器 Rc 等中、又仅将 4 个运算结果中的两个 SIMD 存储在寄存器 Rc 等中的命令。这种命令对以下情况有效。即,在将两个要素每错 1 要素并对每个要素进行运算的情况下有效。这是因为必需错位 1 个、错位 2 个、错位 3 个的运算。

若一般化表现以上固定操作数一方的运算时，则如下所述。即，本处理器 1 也可对于包含指定由 n (≥ 2) 个数据排列构成的第 1 数据群的第 1 操作数、和指定由 n 个数据排列构成的第 2 数据群的第 2 操作数的 SIMD 型命令，分别对 $i=1, 2, \dots, n$ 、上述 j =恒定值的情况下的第 1 数据群的第 i 个数据和第 2 数据组的第 j 个数据组成组所构成的 n 组进行运算。

(3) 执行 SIMD2 项运算、并位移其结果的命令

下面，说明在两排 SIMD 运算中，将位于交叉位置关系上的操作数作为对象进行运算的命令。

[命令 vaddsh]

命令 vaddsh 是以半字（16 比特）单位将两组操作数相加、并将结果仅算术右移位 1 位的 SIMD 型命令。例如，若为

vaddsh Rc, Ra, Rb, 则处理器 1 通过算术逻辑、比较运算器 41,

- (i) 将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相加，将结果算术右移位 1 位后的值存储在寄存器 Rc 的上位 16 位中，同时，
- (ii) 将寄存器 Ra 的下位 16 位与寄存器 Rb 的上位 16 位相加，将结果算术右移位 1 位后的值存储在寄存器 Rc 的下位 16 位中。

这种命令在通过加法未达到 16 比特精度之前下移以确保精度时有效。根据情况需要舍入。在 FFT（蝶形）中，由于反复加减法各复数，而被较多使用。

另外，本处理器 1 即使对减法（vsubsh 等）也执行与该加法命令一样的处理。

另外，本处理器 1 也可具备不仅执行对以上两组操作数的运算（两排 SIMD 的加法、减法）、而且还执行对扩展其的 n 组操作数的运算（4 排、8 排 SIMD 运算等）的功能。

例如，在设寄存器 Ra 中存储的 4 个字节数据从上位开始为 Ra1、Ra2、Ra3、Ra4，设寄存器 Rb 中存储的 4 个字节数据从上位开始为

Rb1、Rb2、Rb3、Rb4 的情况下，作为将寄存器 Ra 与寄存器 Rb 作为操作数的 SIMD 运算命令，也可具备进行以下运算和位移位的命令。即，也可具备分别将 Ra1 与 Rb1、Ra2 与 Rb2、Ra3 与 Rb3、Ra4 与 Rb4 作为运算对象的 4 排 SIMD 型运算命令。例如，是以字节单位将 4 组操作数相加，将结果仅算术右移 1 位的命令 `vaddsb` 等。

这种命令同样对确保精度有效。主要用于取平均的情况（纵向平均）。

另外，进行这种 SIMD 运算和移位的特征命令不限于上述向右仅移位 1 位的命令。即，移位量也可是固定及可变之一，也可是向右及向左之一。另外，也可在右移位时舍入溢出的比特（例如命令 `vaddsrh` 或命令 `vaddsrb` 等）。

（4）累加 SIMD（矢量）数据后进行标量化或矢量低维化的命令
下面，说明对矢量数据进行标量化或矢量低维化的 SIMD 型命令。

[命令 `vsumh`]

命令 `vsumh` 是以半字（16 比特）单位将两个 SIMD 数据（矢量数据）相加、并标量化的 SIMD 型命令。例如，若为

`vsumh Rb, Ra`，则处理器 1 通过算术逻辑、比较运算器 41，将寄存器 Ra 的上位 16 位与寄存器 Rb 的上位 16 位相加，将结果存储在寄存器 Rb 中。

这种命令可用于各种用途。可使平均（横向平均）、单独运算（积和、加法）的结果一致等。

[命令 `vsumh2`]

命令 `vsumh2` 是以半字单位、对设两个 SIMD 数据（矢量数据）为 1 组的两组分别进行要素累加、并标量化的 SIMD 型命令。例如，若为

`vsumh2 Rb, Ra`，则处理器 1 通过算术逻辑、比较运算器 41，

（i）将寄存器 Ra 的最上位字节与中上位字节累加，将结果存储

在寄存器 Rb 的上位 16 位中，同时，

(ii) 将寄存器 Ra 的中下位字节与最下位字节累加，将结果存储在寄存器 Rb 的下位 16 位中。

这种命令作为图像处理、动作补偿 (MC)、半对 (ペル) 用命令是有效的。

另外，本处理器 1 也可具备不仅执行对以上两排 SIMD 数据标量化的运算、而且还执行标量化扩展其的 n (4、8 等) 个要素构成的 n 排 SIMD 数据运算的功能。

例如，也可具备在设寄存器 Ra 中存储的 4 个字节数据从上位开始为 Ra1、Ra2、Ra3、Ra4 的情况下，累加这些 Ra1、Ra2、Ra3、Ra4，将结果存储在寄存器 Rb 中的命令。

另外，本处理器 1 不仅将由多个要素数据构成的矢量变换为由 1 个要素数据构成的标量，而且也可进行使要素数据的个数减少的矢量低维化。

另外，运算的种类不仅限于加法，也可是算出平均值的运算。这种命令对平均、运算结果的一致等有效。

(5) 其它关于 SIMD 的命令

下面，说明不属于以上分类的其它 SIMD 型命令。

[命令 vexth]

命令 vexth 是以半字 (16 比特) 单位、分别符号扩展两个 SIMD 数据的 SIMD 型命令。例如，若为

vexth Mm, Rb, Ra, 则处理器 1 通过变换器 47 的饱和功能块 (SAT) 47a 等，

(i) 将寄存器 Ra 的上位 16 位符号扩展为 32 比特，将结果存储在运算寄存器 MHm 的上位 16 位和运算寄存器 MLm 的上位 16 位中，与之并行，

(iii) 将寄存器 Ra 的上位 16 位符号扩展为 32 比特，将结果存

储在运算寄存器 MHm 的下位 16 位和运算寄存器 MLm 的下位 16 位中，与之并行，

(iii) 将寄存器 Ra 的 32 比特存储在寄存器 Rb 中。

所谓符号扩展是维持原样延长符号信息的数据长度，例如，将由半字表现的带符号数值变换为由字表现的相同数值。具体而言，是埋尽由原始数据的符号位（最上位位）值扩展的上位位的处理。

这种命令对向累加器传送 SIMD 数据（必需精度的情况下）有效。

[命令 vasubb]

命令 vasubb 是以字节单位、分别减去 4 组 SIMD 数据、并将结果得到的 4 个符号存储在条件标志寄存器中的 SIMD 型命令。例如，若为

vasubb Rc, Rb, Ra, 则处理器 1 通过算术逻辑、比较运算器 41 等，

(i) 从寄存器 Rb 的最上位 8 位中减去寄存器 Ra 的最上位 8 位，将结果存储在寄存器 Rc 的最上位 8 位中，同时，将其符号存储在条件标志寄存器（CFR）32 的 VC3 中，与之并行，

(ii) 从寄存器 Rb 的中上位 8 位中减去寄存器 Ra 的中上位 8 位，将结果存储在寄存器 Rc 的中上位 8 位中，同时，将其符号存储在条件标志寄存器（CFR）32 的 VC2 中，与之并行，

(iii) 从寄存器 Rb 的中下位 8 位中减去寄存器 Ra 的中下位 8 位，将结果存储在寄存器 Rc 的中下位 8 位中，同时，将其符号存储在条件标志寄存器（CFR）32 的 VC1 中，与之并行，

(iv) 从寄存器 Rb 的最下位 8 位中减去寄存器 Ra 的最下位 8 位，将结果存储在寄存器 Rc 的最下位 8 位中，同时，将其符号存储在条件标志寄存器（CFR）32 的 VC0 中。

这种命令在每次得到绝对值差的和时暂时需要 9 比特精度的情况下有效。

[命令 vabssumb]

命令 vabssumb 是以字节单位、分别相加 4 组 SIMD 数据的绝对值、并将其结果与其它 4 字节数据相加的 SIMD 型命令。例如，若为

vabssumb Rc, Rb, Ra, 则处理器 1 通过算术逻辑、比较运算器 41 等，将寄存器 Ra 的最上位 8 位的绝对值、中上位 8 位的绝对值、中下位 8 位的绝对值和最下位 8 位的绝对值相加，并将其结果与寄存器 Rb 的 32 比特相加，将所得结果存储在寄存器 Rc 中。另外，本处理器 1 通过条件标志寄存器 (CFR) 32 的标志 VC0-VC3，特定存储在寄存器 Ra 中的各字节的绝对值。

这种命令通过与上述命令 vasubb 组合使用，在对多组数据算出各组的差后，得到合计各差绝对值的值，算出图像处理动作预测的差分绝对值等情况下有效。

(6) 其它关于屏蔽运算等的命令

下面，说明不是 SIMD 型命令、但进行特征处理的命令。

[命令 addmsk]

命令 addmsk 是屏蔽两个操作字一方的部分比特 (上位) 后相加的命令。例如，若为

addmsk Rc, Rb, Ra, 则处理器 1 通过算术逻辑、比较运算器 41 或变换器 47 等，仅在由条件标志寄存器 (CFR) 32 的 BP0 指定的范围 (下位比特) 中存储在寄存器 Ra 和寄存器 Rb 中的数据相加后存储在寄存器 Rc 中，同时，对未指定的范围 (上位位)，将寄存器 Ra 的值原样存储在寄存器 Rc 中。

这种命令作为与模数寻址 (在 DSP 中一般) 的对应有效。由于蝶形运算的准备而事先将数据重新排列为特定图案中，必需该命令。

本处理器 1 即使对减法命令 (submsk) 也可执行与该加法命令一样的处理。

[命令 mskbrvh]

命令 `mskbrvh` 是颠倒（将比特的排列变为反序）两个操作字一方的部分比特（下位）后、与另一方操作数进行比特连接的命令。例如，若为

`mskbrvh Rc, Rb, Ra`，则处理器 1 通过变换器 47 等，在对寄存器 `Rb` 的下位 16 位进行反序后，在由条件标志寄存器（CFR）32 的 `BPO` 指定的位位置上，连接寄存器 `Ra` 的数据和寄存器 `Rb` 的数据，将结果存储在寄存器 `Rc` 中。此时，将寄存器 `Rb` 的上位 16 位中、比 `BPO` 指定的位置靠下位部分屏蔽为 0。

这种命令在对应于颠倒寻址，由于蝶形运算的准备而事先将数据重新排列为特定图案中，需要该命令。

另外，本处理器 1 不仅对 16 比特进行反序，即使对将 1 字节等区域反序的命令（`mskbrvb` 等），也可执行与该命令一样的处理。

[命令 `msk`]

命令 `msk` 是将构成操作数的比特中、由指定的两个位位置夹持的区域屏蔽（为 0）、或屏蔽该区域之外的命令。例如，若为

`msk Rc, Rb, Ra`，则处理器 1 通过变换器 47 等，

(i) 在 $Rb[12: 8] \cong Rb[4: 0]$ 的情况下，

将寄存器 `Ra` 中存储的 32 比特中，将从由寄存器 `Rb` 的第 0-4 的 5 比特 `Rb[4: 0]` 指定的位位置开始由寄存器 `Rb` 的第 8-12 的 5 比特 `Rb[12: 8]` 指定的位位置保持原样不变，将其它位屏蔽（为 0），存储在寄存器 `Rc` 中。

(i) 在 $Rb[12: 8] < Rb[4: 0]$ 的情况下，

将寄存器 `Ra` 中存储的 32 比特中，将从由寄存器 `Rb` 的第 8-12 的 5 比特 `Rb[12: 8]` 指定的位位置开始由寄存器 `Rb` 的第 0-4 的 5 比特 `Rb[4: 0]` 指定的位位置屏蔽（为 0），将其它比特保持原样不变，存储在寄存器 `Rc` 中。

这种命令可用于比特字段的抽取、插入（组装）中。可用于由软

件进行 VLD/VLC 的情况。

[命令 bseq]

命令 bseq 是从操作数的 MSB 的下一比特开始计数连接的符号位数的命令。例如，若为

bseq Rb, Ra, 则处理器 1 通过变换器 47 的 BSEQ 功能块 47b 等, 从寄存器 Ra 的下一比特中计数连续的符号位数, 并将结果存储在寄存器 Rb 中。另外, 在寄存器 Ra 的值为 0 的情况下, 将 0 存储在寄存器 Rb 中。

这种命令可用于检测有效进位。因为动态范围取得大, 所以有时进行部分浮动小数点运算。例如, 在每次与排列内最有效进位多的数据一致标准化所有数据并进行运算时, 可使用该命令。

[命令 ldbq]

命令 ldbq 是符号扩展 2 字节数据后从存储器下载到寄存器中的命令。例如, 若为

ldbq Rb: Rb+1, (Ra, D9), 则处理器 1 通过 I/F 部 50 等, 根据将位移值 (D9) 加到寄存器 Ra 的值上的地址, 符号扩展两个字节数据, 分别下载到寄存器 Ra 和寄存器 (Ra+1) 中。

这种命令高速提供数据。

另外, 本处理器 1 不仅对两个寄存器下载, 即使对下载到 1 个寄存器的上位半字和下位半字等 (1bdh 等), 也可执行与该下载命令一样的处理 (伴随符号扩展的下载)。

[命令 rde]

命令 rde 是读入外部寄存器的值、在读入失败时发生错误例外的命令。例如, 若为

rde C0: C1, Rb, (Ra, D5), 则处理器 1 通过 I/F 部 50 等, 将位移值 (D5) 加到寄存器 Ra 的值上的值作为外部寄存器序号, 将该外部寄存器 (扩展寄存器部 80) 的值读入寄存器 Rb 中, 同时,

将读入成功、失败输出到条件标志寄存器（CFR）32 的条件标志 C0 及 C1。在失败时，发生扩展寄存器错误例外。

这种命令作为控制硬件累加器的命令是有效的。硬件一返回错误，则发生例外，并反映到标志中。

另外，本处理器 1 不仅对从外部寄存器读入，即使写出到外部寄存器（命令 wte），也可执行与该读取命令一样的处理（标志的设置、例外的发生）。

[命令 addarvw]

命令 addarvw 是进行绝对值舍入（向离开零的方向舍入）相加的命令。例如，若为

addarvw Rc, Rb, Ra, 则处理器 1 通过算术逻辑、比较运算器 41 等，

将寄存器 Ra 的 32 比特与寄存器 Rb 的 32 比特相加，若结果为正，则进行进位舍去比特的处理，若结果为负，则进行舍去舍入比特的处理。具体而言，将寄存器 Ra 和 Rb 的值相加，在寄存器 Ra 的值为正的情况下，加 1。另外，在进行绝对值舍入的情况下，将由 1 埋置比舍入比特下位的比特的值加入寄存器 Rb 中。

这种命令对绝对值舍入（在离开零的方向上舍入）的加法 IDCT（Inverse Discrete Cosine Transform）有效。

发明效果

如上所述，可知，本发明的处理器是执行由 1 个命令来运算多个数据的 SIMD 型命令的处理器，不仅对两个数据排列中同一顺序的数据彼此、而且对处于交叉位置关系上的数据彼此或处于对称位置关系上的数据彼此并列执行运算。因此，可高速化对数字滤波器等处于对称位置关系上的数据实施相同运算的处理，实现适于多媒体处理等的处理器。

这里，在运算的种类是乘法、积和或积差的情况下，也可又仅输

出所得运算结果各自的下位部分、又仅输出上位部分、又仅输出部分运算结果。由此，对于复数的内积等，因为在运算的同时执行处理整数数据的情况或处理固定小数点数据情况下必需的比特舍去处理，所以高速化使用复数等 2 维数据的运算、例如使用 2 维坐标的图像处理或使用振幅和相位等二维表现的声音信号的处理等。

如上所述，本发明的处理器与通常的微型计算机相比，排列性高，AV 媒体类信号处理高速，可作为共同的芯片处理器用作便携电话、移动 AV 设备、数字 TV、DVD 等中，在期望高性能、高性价比的多媒体设备出现的今天，实用价值极高。

表1

寄存器名	位宽度	根数	用途
R0~R31	32位	32根	通用寄存器。用于数据存储器的指示器，在运算命令中的数据储存等。
TAR	32位	1根	分支用寄存器。用于分支时分支地址的储存。
LR	32位	1根	连结用寄存器。
SVR	16位	2根	保存用寄存器。保存条件标记(CRF)和多种模式。
M0~M1 (MH0:ML0~MH1:ML1)	64位	2根	运算用寄存器。用于在运算命令中的数据储存。

表2

C0~C7寄存器名	1位	8根	条件标记。表示条件成立、不成立。
VC0~VC3	1位	4根	媒体处理扩展命令用标志。表示条件成立·不成立
OVS	1位	1根	结束流程标记。检测运算时的结束流程。
CAS	1位	1根	进位标志。检测运算时的进位。
BPO	5位	1根	位的位置指定。屏蔽处理命令时指定作为处理对象的位的位置。
ALN	2位	1根	字节调节指定。
FXP	1位	1根	固定小数点演算模式
UDR	32位	1根	未定义寄存器

表3

种类	运算器	命令操作代码
存储器传输命令(取数)	M	ld, ldh, ldhu, ldb, dbu, ldp, ldhp, ldbp, ldbh, ld buh, ldbhp, ldbuhp
存储器传输命令(存储)	M	st, sth, stb, stp, sthp, stbp, stbh, stbhp
存储器传输命令(其他)	M	dpref, ldstb
外部寄存器的传输命令	M	rd, rde, wt, wte
分支命令	B	br, brl, call, jmp, jmpf, jmp, ret, jmpf, jloop, s etbb, setlr, settar
软件中断命令	B	rti, pi0, pi0l, pi1, pi1l, pi2, pi2l, pi3, pi3l, pi 4, pi4l, pi5, pi5l, pi6, pi6l, pi7, pi7l, sc0, sc1, sc2, sc3, sc4, sc5, sc6, sc7
VMP/中断控制命令	B	intd, inte, vmpsleap, vmpsus, vmpswd, vmpsw, vm pwait
算术运算命令	A	abs, absvh, absvw, add, addarvw, addc, addmsk, ad ds, addsr, addu, addvh, addvw, neg, negvh, negvw, rsub, s1add, s2add, sub, subc, submsk, subs, subv h, subvw, max, min
逻辑运算命令	A	and, andn, or, sethi, xor, not
比较命令	A	cmpCC, cmpCCa, cmpCCn, cmpCCo, tstn, tstna, tstn n, tstno, tstz, tstza, tstzn, tstzo
转换命令	A	mov, movcf, mvclcas, mvclovs, setlo, vcchk
NOP命令	A	nop
移位命令1	S1	asl, aslvh, aslvw, asr, asrvh, asrvw, lsl, lsr, rol, ror
移位命令2	S2	aslp, aslpvw, asrp, asrpvw, lsp, lsrp

表4

种类	运算器	命令操作代码
抽出命令	S2	ext, extb, extbu, exth, exthu, extr, extru, extu
屏蔽命令	C	msk, mskgen
饱和命令	C	sat12, sat9, satb, satbu, sath, satw
变换命令	C	valn, valn1, valn2, valn3, valnvc1, valnvc2, valnvc3, valnvc4, vhpkb, vhpkh, vhunpkb, vhunpkh, vintlhb, vintlhh, vintlhb, vintlhh, vlpkb, vlpkb u, vlpkh, vlpkhu, vlunpkb, vlunpkbu, vlunpkh, vlunpkhu, vstovb, vstovh, vunpk1, vunpk2, vxchngh, vexth
位计数命令	C	bcnt1, bseq, bseq0, bseq1
其他	C	byterev, extw, mskbrvb, mskbrvh, rndvh, movp
乘法命令1	X1	fmulhh, fmulhhr, fmulhw, fmulhww, hmul, lmul
乘法命令2	X2	fmulww, mul, mulu
积和命令1	X1	fmachh, fmachhr, fmachw, fmachww, hmac, lmac
积和命令2	X2	fmacww, mac
积差命令1	X1	fmsuhh, fmsuhhr, fmsuhw, fmsuww, hmsu, lmsu
积差命令2	X2	fmsuww, msu
除法命令	DIV	div, divu
调试器命令	DBGM	dbgm0, dbgm1, dbgm2, dbgm3

表5

种类	运算器	命令操作代码
SIMD 算术运算命令	A	vabshvh, vaddb, vaddh, vaddhvc, vaddhvh, vaddrhvc, vaddsb, vaddsh, vaddsrh, vaddsrh, vasubb, vchk, vhaddh, vhaddhvh, vsubh, vsubhvh, vladdh, vladdhvh, vlsubh, vlsubhvh, vnegb, vnegh, vneghvh, vsaddb, vsaddh, vsgh, vsrsubb, vsrsubh, vsubb, vssubh, vsubb, vsubh, vsubhvh, vsubsh, vsu mh, vsumh2, vsumrh2, vxaddh, vxaddhvh, vxsubh, vxsubhvh, vmaxb, vmaxh, vminb, vminh, vmovt, vsef
SIMD 比较命令	A	vcmpeqb, vcmpeqh, vcmpegb, vcmpegh, vcmptgb, vcmptgh, vcmpleb, vcmpleh, vcmpltb, vcmplth, vcmpneb, vcmpneh, vscmpeqb, vscmpeqh, vscmpgeb, vscmpgeh, vscmpgtb, vscmpgth, vscmpleb, vscmpleh, vscmpltb, vscmpltth, vscmpneb, vscmpneh
SIMD 移位命令1	S1	vaslb, vaslh, vaslvh, vasrb, vasrh, vasrvh, vlslb, vlslh, vlslrb, vlslrh, vrolb, vrolh, vrorb, vrorh
SIMD 移位命令2	S2	vasl, vaslvw, vasr, vasrvw, vlsl, vlslr
SIMD 其他命令	C	vsath, vsath12, vsath8, vsath8u, vsath9
SIMD 饱和命令	C	vabssumb, vrndvh
SIMD 乘法命令	X2	vfmulh, vfmulhr, vfmulw, vfmulh, vfmulhr, vfmulw, vhmul, vlfmulh, vlfmulhr, vlfmulw, vmul, vpmulhww, vxfmulh, vxfmulhr, vxfmulw, vxfmul
SIMD 积和命令	X2	vfmach, vfmachr, vfmacw, vfmach, vfmachr, vfmacw, vhmach, vlfmach, vlfmachr, vlfmacw, vlmach, vpmachww, vxfmach, vxfmachr, vxfmacw, vxfmach
SIMD 积差命令	X2	vfmsuh, vfmsuw, vhfmsuh, vhfmsuw, vhmsh, vlfmsuh, vlfmsuw, vlmsu, vmsu, vxfmsuh, vxfmsuw, vxmsu

图6

记号	含义
$X[i]$	X的位编号
$X[i:j]$	从X的位编号j到位编号i;
$X:Y$	X和Y的连结
$\{n\{X\}\}$	X的n次重复
$\text{sextM}(X, N)$	将X从N位幅到M位幅符号扩展。M的默认值为32。 N的默认值是X的所有位幅。
$\text{uextM}(X, N)$	将X从N位幅到M位幅零扩展。M的默认值为32。 N的默认值是X的所有位幅。
$\text{smul}(X, Y)$	有符号的乘法 $X*Y$
$\text{umul}(X, Y)$	没有符号的乘法 $X*Y$
$\text{sdiv}(X, Y)$	有符号的除法的商的整数部分 X/Y
$\text{smod}(X, Y)$	被除数和同符号的余
$\text{udiv}(X, Y)$	没有符号的除法的商 X/Y 。
$\text{umod}(X, Y)$	余
$\text{abs}(X)$	绝对值
$\text{bseq}(X, Y)$	<pre> for (i=0; i<32; i++) { if (X[31-i] != Y) break; } result = i; </pre>
$\text{bcnt}(X, Y)$	<pre> S = 0; for (i=0; i<32; i++) { if (X[i] == Y) S++; } result = S; </pre>
$\text{max}(X, Y)$	$\text{result} = (X > Y) ? X : Y$
$\text{min}(X, Y)$	$\text{result} = (X < Y) ? X : Y;$
$\text{tstz}(X, Y)$	$X \& Y == 0$
$\text{tstn}(X, Y)$	$X \& Y != 0$

表7

记号	含义		
Ra	Ra[31:0]	编号a的寄存器	(0 ≤ a ≤ 31)
Ra+1	R(a+1)[31:0]	编号a+1的寄存器	(0 ≤ a ≤ 30)
Rb	Rb[31:0]	编号b的寄存器	(0 ≤ b ≤ 31)
Rb+1	R(b+1)[31:0]	编号b+1的寄存器	(0 ≤ b ≤ 30)
Rc	Rc[31:0]	编号c的寄存器	(0 ≤ c ≤ 31)
Rc+1	R(c+1)[31:0]	编号c+1的寄存器	(0 ≤ c ≤ 30)
Ra2	Ra2[31:0]	编号a2的寄存器	(0 ≤ a2 ≤ 15)
Ra2+1	R(a2+1)[31:0]	编号a2+1的寄存器	(0 ≤ a2 ≤ 14)
Rb2	Rb2[31:0]	编号b2的寄存器	(0 ≤ b2 ≤ 15)
Rb2+1	R(b2+1)[31:0]	编号b2+1的寄存器	(0 ≤ b2 ≤ 14)
Rc2	Rc2[31:0]	编号c2的寄存器	(0 ≤ c2 ≤ 15)
Rc2+1	R(c2+1)[31:0]	编号c2+1的寄存器	(0 ≤ c2 ≤ 14)
Ra3	Ra3[31:0]	编号a3的寄存器	(0 ≤ a3 ≤ 7)
Ra3+1	R(a3+1)[31:0]	编号a3+1的寄存器	(0 ≤ a3 ≤ 6)
Rb3	Rb3[31:0]	编号b3的寄存器	(0 ≤ b3 ≤ 7)
Rb3+1	R(b3+1)[31:0]	编号b3+1的寄存器	(0 ≤ b3 ≤ 6)
Rc3	Rc3[31:0]	编号c3的寄存器	(0 ≤ c3 ≤ 7)
Rc3+1	R(c3+1)[31:0]	编号c3+1的寄存器	(0 ≤ c3 ≤ 6)
Rx	Rx[31:0]	编号x的寄存器	(0 ≤ x ≤ 3)

表8

记号	含义	
+	加法	
-	减法	
&	逻辑积	
	逻辑和	
!	逻辑否定	
<<	逻辑左移动(算术左移动)	
>>	算术右移动	
>>>	逻辑右移动	
^	排他性逻辑和	
~	逻辑否定	
==	等于(相等)	
!=	不等于(不相等)	
>	大于	有符号(左边、右边的MSB视为符号)
>=	大于等于	有符号(左边、右边的MSB视为符号)
>(u)	大于	无符号(左边、右边的MSB符号)
>=(u)	大于等于	无符号(左边、右边的MSB符号)
<	小于	有符号(左边、右边的MSB视为符号)
<=	小于等于	有符号(左边、右边的MSB视为符号)
<(u)	小于	无符号(左边、右边的MSB符号)
<=(u)	小于等于	无符号(左边、右边的MSB符号)

表9

记号	含义
D(addr)	存储器内的地址addr的双倍字数据
W(addr)	存储器内的地址addr的字数据
H(addr)	存储器内的地址addr的半数据
B(addr)	存储器内的地址addr的字节数据
B(addr, bus_lock)	锁住与存取存储器内的地址addr的字节数据的同时使用的总线。 (但是, 如果有不能锁住的总线, 则不锁)
B(addr, bus_unlock)	解除与存取存储器内的地址addr的字节数据同时使用的总线的锁住。 (但是, 如果有不能锁住的, 或没有锁住的总线, 则忽略锁住的解除)
EREG(num)	编号num的扩展寄存器
EREG_ERR	之前的扩展寄存器存取发生错误时为1; 如果没有发生错误, 则为零。
<-	结果的写入
=>	命令的同义词(编程转换)
reg#(Ra)	多用途寄存器Ra的寄存器编号(5位的简数)
0x	16进数的前置语
0b	2进数的前语
tmp	临时变量
UD	不定值(安装相关或动态变化值)
Dn	置换值(n为自然数表示的位数)
In	当前值(n为自然数表示的位数)

表10

记号	含义
○ 文字说明	
if (条件) {	条件成立时执行;
} else {	条件不成立时执行;
}	
条件A成立时执行, if(条件A);	※ 不成立时不执行
for (式1;式2;式3)	※ 同C语言
(式1)? 式2:式3	※ 同C语言
○ 用语说明	
定义说明使用的用语.	
整数乘法	用smul定义的乘法
固定小数点乘法	整数运算后,算术左移动。PSR.FXP为0时移动量为1 比特,为1时移动量为2比特。
SIMD示算	straight / cross / high / low / pair 半字向量数据的上位16比特为RH,下位16比特为RL。如果Ra寄存器与Rb 寄存器之间进行运算时,各运算如下定义。
straight	RHa和RHb, RLa和RLb 之间运算。
cross	RHa和RLb, RLa和RHb 之间运。
high	RHa和RHb, RLa和RHb 之间运算。
low	RHa和RLb, RLa和RLb 之间运算。
pair	RH和RHb, RH和RLb 之间运算。(RH为32位数据)

图1

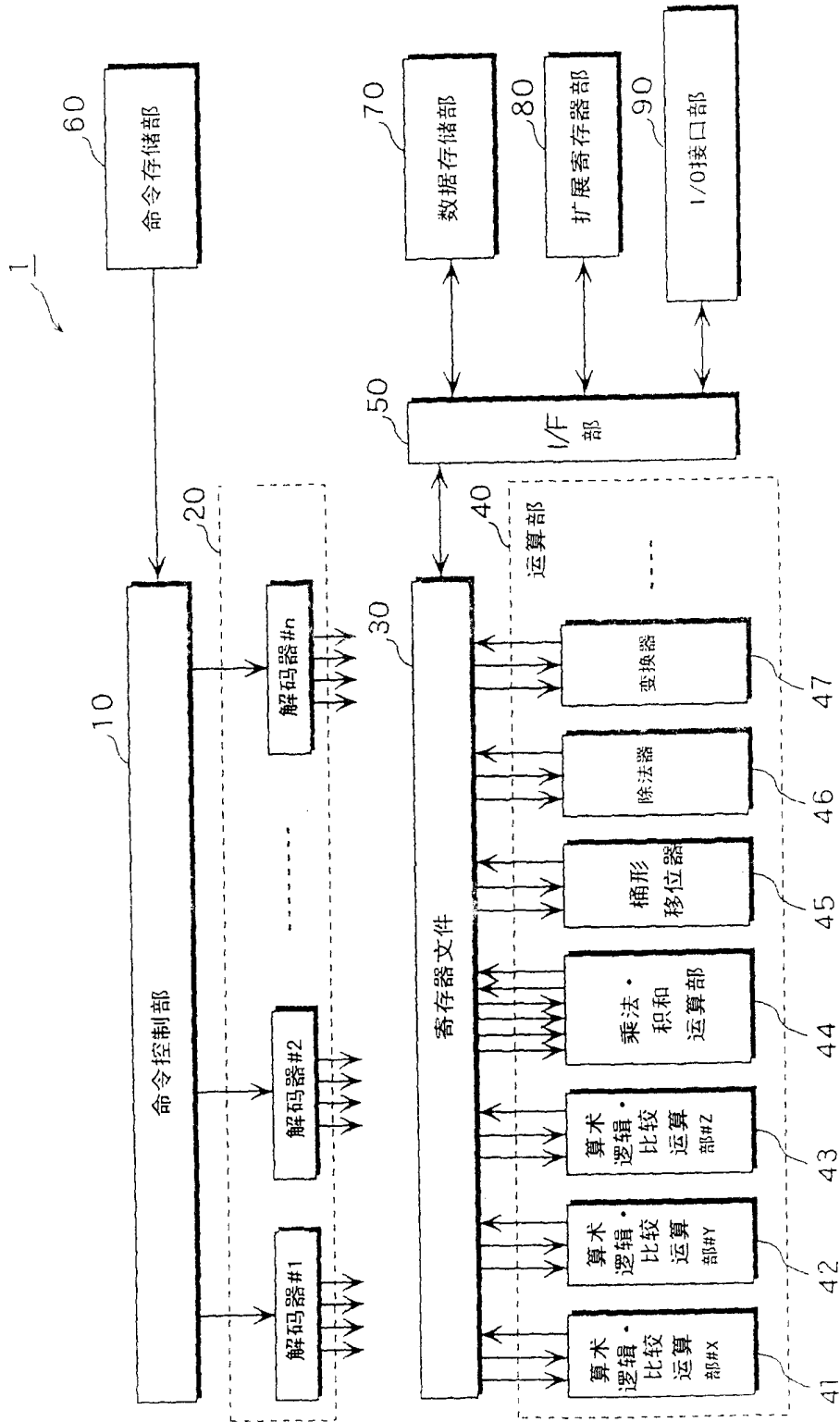


图2

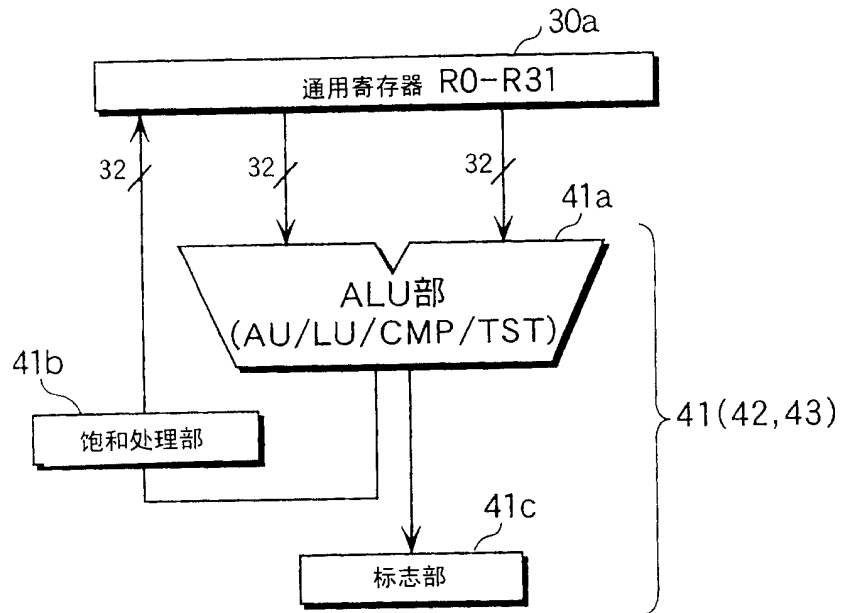


图3

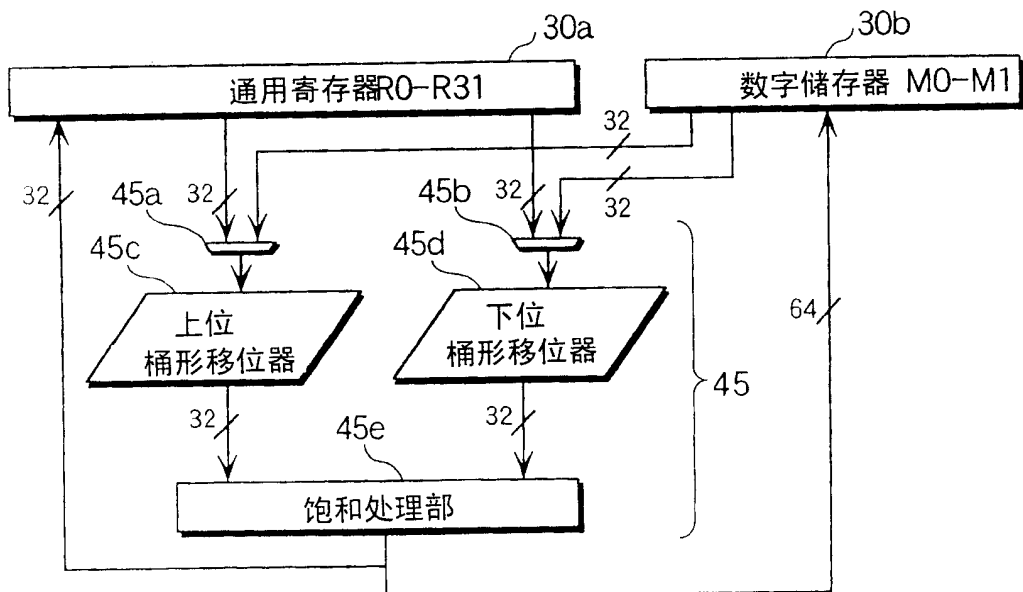


图4

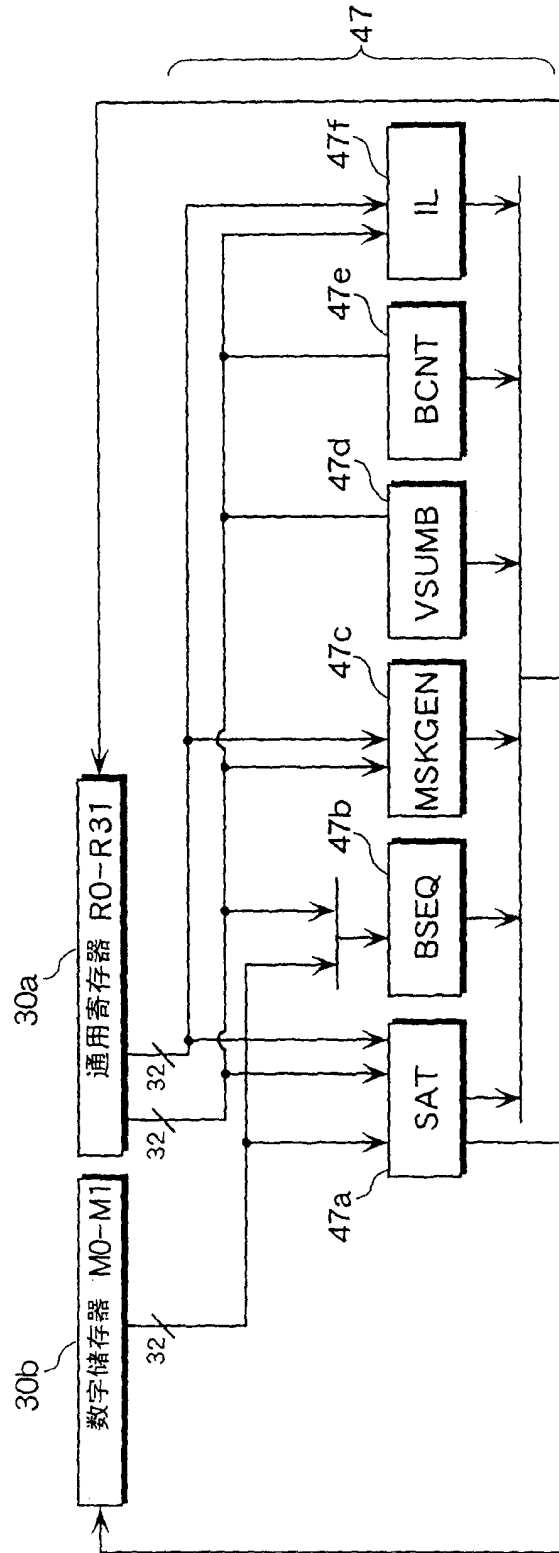


图5

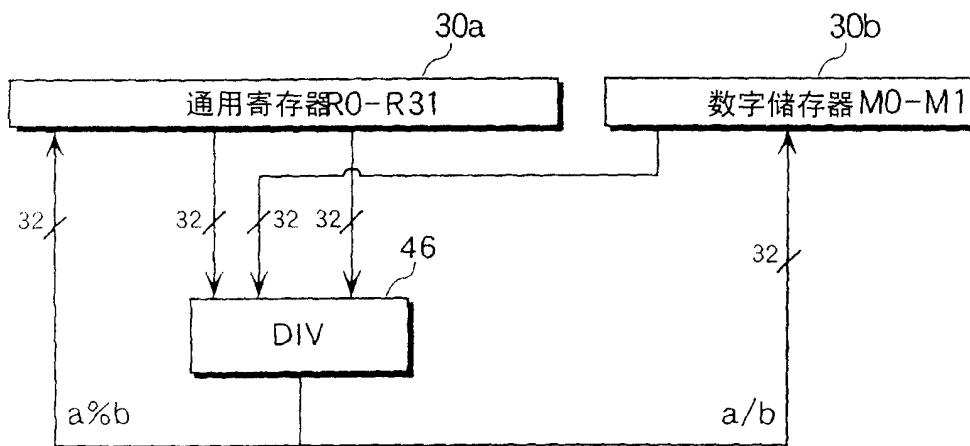


图6

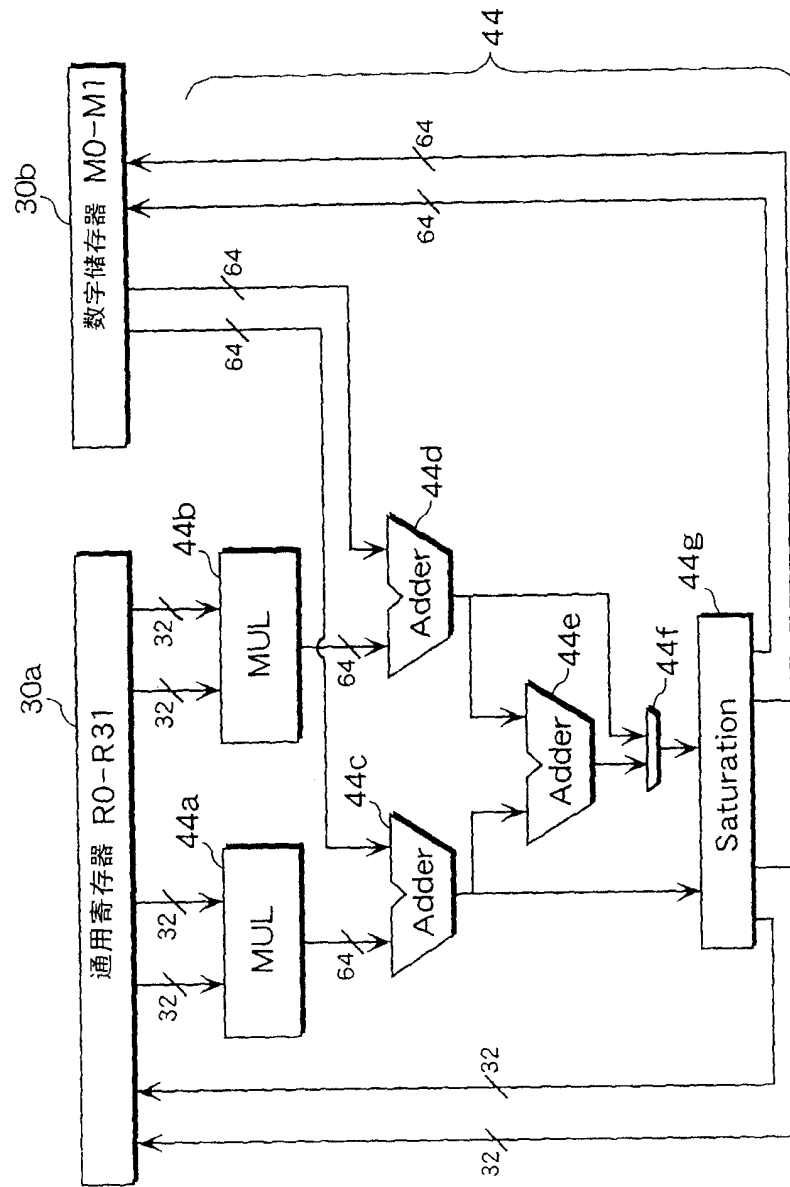


图7

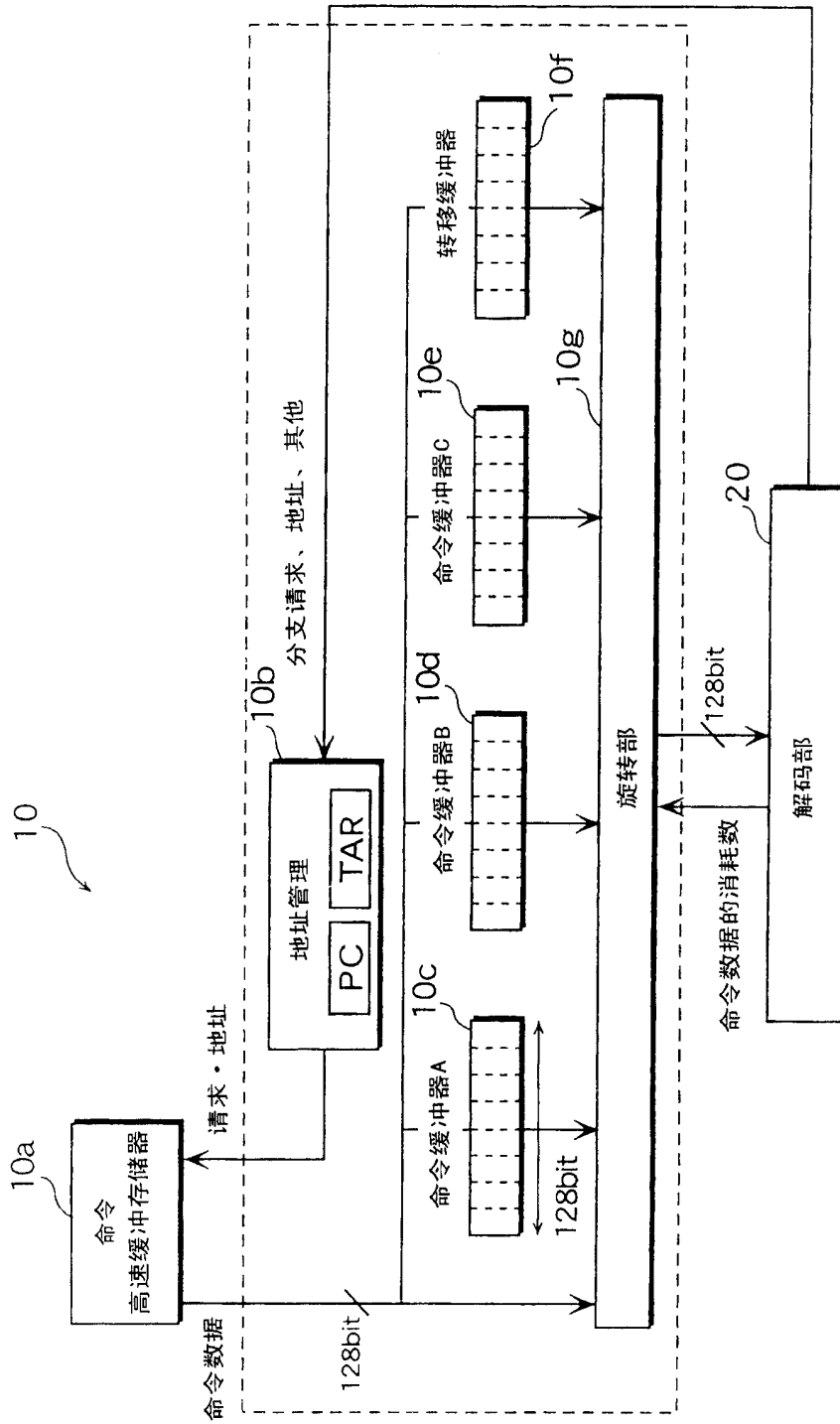


图8

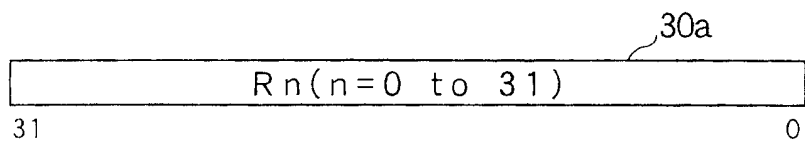


图9

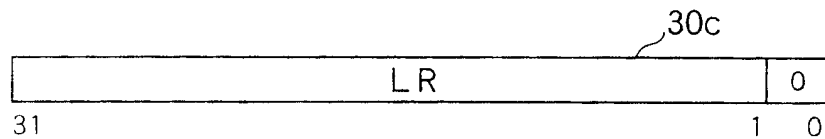


图10



图11

位	31	30	29	28	27	26	25	24
位名	reserved	SWE	FXP	reserved	IH	EH	PL	
位	23	22	21	20	19	18	17	16
位名	LPIE3	LPIE2	LPIE1	LPIE0	reserved	reserved	AEE	IE
位	15	14	13	12	11	10	9	8
位名	Reserved							
位	7	6	5	4	3	2	1	0
位名	IM[7:0]							

图12

32

位	31	30	29	28	27	26	25	24	
位名	ALN		reserved	BPO					
位	23	22	21	20	19	18	17	16	
位名	reserved		reserved		VC3	VC2	VC1	VC0	
位	15	14	13	12	11	10	9	8	
位名	reserved					OVS			CAS
位	7	6	5	4	3	2	1	0	
位名	C7	C6	C5	C4	C3	C2	C1	C0	

图13

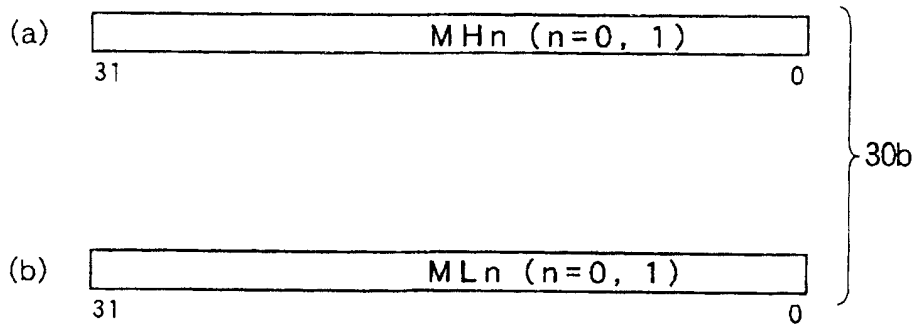


图14

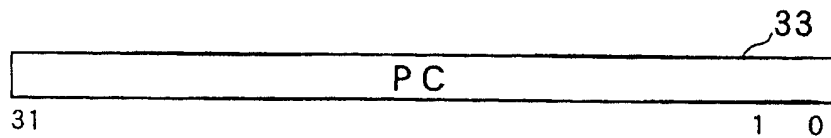


图15

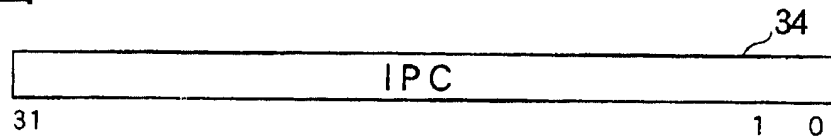


图16



图17

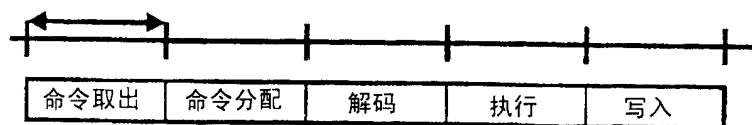


图18

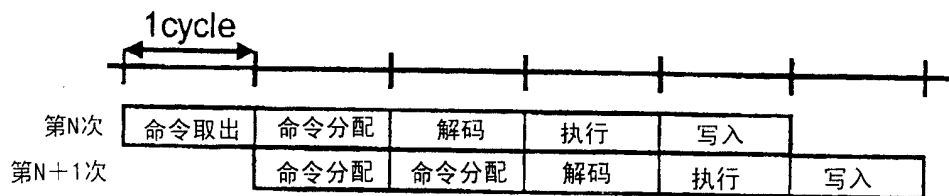


图19



图 20

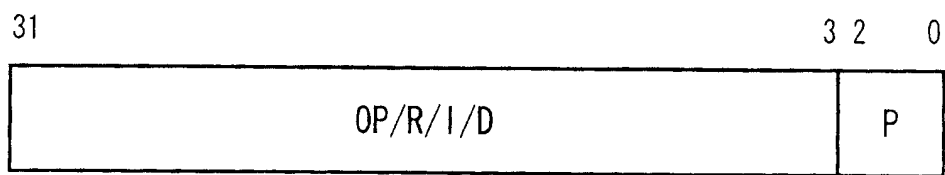


图21

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16	
ALU add 列	S I N G L E	字	add	Rc, Ra, Rb Rb, Ra, i12s SP, i19s Ra2, Rb2 Rc3, Ra3, Rb3 Ra2, i05s SP, i11s					32 16	
			addu	Rb, GP, i16u Rb, SP, i16u Ra3, SP, i08u					32 16	
			addc	Rc, Ra, Rb	W:cas, c0: c1		有进位的加法			
			addvw	Rc, Ra, Rb	W:ovs		有溢出的加法			
			adds	Rc, Ra, Rb			$Ra + Rb \rightarrow Rc$		32	
			addsr	Rc, Ra, Rb			$Ra + Rb + 1 \rightarrow Rc$			
			s1add	Rc, Ra, Rb Rc3, Ra3, Rb3			$Ra + Rb \rightarrow Rc$		16	
			s2add	Rc, Ra, Rb Rc3, Ra3, Rb3			$Ra + Rb \rightarrow Rc$ ($\gg 2$)		32 16	
		addmsk	Rc, Ra, Rb	R:BP0		$Ra \oplus Rb$				
		addarvw	Rc, Ra, Rb							
		SIMD	半字	faddvh	Rc, Ra, Rb	W:ovs				
				vaddh	Rc, Ra, Rb			$Ra + Rb \rightarrow Rc$		
				vaddhvh	Rc, Ra, Rb	W:ovs				
				vsaddh	Rb, Ra, i08s			$Ra + Rb \rightarrow Rb$ + 当前值 + 当前值		
	vaddsh			Rc, Ra, Rb			$Ra + 1, Rb + 1 \rightarrow Rc$ (+1) (+1) ($\gg 1$) ($\gg 1$) (有舍入)			
	vaddsrh			Rc, Ra, Rb						
	vaddhvc			Rc, Ra, Rb	R:VC				A 32	
	vaddrhvc			Rc, Ra, Rb						
	vxaddh		Rc, Ra, Rb			$Ra \oplus Rb$				
	vxaddhvh		Rc, Ra, Rb	W:ovs						
	vhaddh		Rc, Ra, Rb			$Ra \oplus Rb$				
	vhaddhvh		Rc, Ra, Rb	W:ovs						
	vladdh		Rc, Ra, Rb			$Ra \oplus Rb$				
	vladdhvh		Rc, Ra, Rb	W:ovs						
	字节	vaddb	Rc, Ra, Rb			$Ra + Rb \rightarrow Rc$				
		vsaddb	Rb, Ra, i08s			$Ra + Rb \rightarrow Rb$ (当前值)				
		vaddsb	Rc, Ra, Rb			$Ra + 1, Rb + 1 \rightarrow Rc$ (+1)(+1)(+1)(+1) ($\gg 1$) ($\gg 1$) ($\gg 1$) ($\gg 1$) (有舍入)				
		vaddsrb	Rc, Ra, Rb							

图22

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	
ALU sub 列	S I N G L E	字	sub	Rc,Rb,Ra Rb2,Ra2 Rc3,Rb3,Ra3				31 16	
			rsub	Rb,Ra,i08s Ra2,Rb2 Ra2,i04s			当前值 - Ra → Rb (Rb2)	32 16	
			subc	Rc,Rb,Ra	W:cas,c0:c1		有进位的加法		
			subvw	Rc,Rb,Ra	W:ovs		有溢出的加法		
			subs	Rc,Rb,Ra			Ra - Rb → Rb		
			submsk	Rc,Rb,Ra	R:BP0		Ra CFR:BP0 Rb		
		半字	fsubvh	Rc,Rb,Ra					
	S I M D	半字	vsubh	Rc,Rb,Ra			Ra Rb		
			vsubhvh	Rc,Rb,Ra	W:ovs		Ra Rb		
			vsrsubh	Rb,Ra,i08s			当前值 - 当前值		
			vsubsh	Rc,Rb,Ra			Ra Rb		
			vxsubh	Rc,Rb,Ra			Ra Rb		
			vxsubhvh	Rc,Rb,Ra	W:ovs		Ra Rb		
			vhsbuh	Rc,Rb,Ra			Ra Rb		
			vhsbuhvh	Rc,Rb,Ra	W:ovs		Ra Rb		
			vlsbuh	Rc,Rb,Ra			Ra Rb		
			vlsbuhvh	Rc,Rb,Ra	W:ovs		Ra Rb		
			字	vsubb	Rc,Rb,Ra			Ra Rb	
				vsrsubb	Rb,Ra,i08s			当前值 - 当前值	
				vasubb	Rc,Rb,Ra	R:VC		Ra Rb	

图23

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
ALU logic 列	S I N G L E	字	and	Rc,Ra,Rb Rb,Ra,i08u Ra2,Rb2			逻辑积	A	32 16 32 16 32 16
			andn	Rc,Ra,Rb Rb,Ra,i08u Ra2,Rb2					
			or	Rc,Ra,Rb Rb,Ra,i08u Ra2,Rb2			逻辑和		
			xor	Rc,Ra,Rb Rb,Ra,i08u Ra2,Rb2			排他性逻辑和		
ALU mov 列	S I N G L E	字	mov	Rb,Reg32 Reg32,Rb Rb2,Reg16 Reg16,Rb2 Ra2,Rb Ra,i16s Ra2,i08s			Reg32 = TAR LR SVR PSR CFR MH0 MHI ML0 ML1 EPSR IPC IPSR PC EPC PSR0 PSR1 PSR2 PSR3 CFR0 CFR1 CFR2 CFR3 Reg16 = TAR LR MH0 MHI	A	32 16 32 16 32 16
			movp	Rc:Rc+1,Ra,Rb			Rc←Ra; Rc+1←Rb;		
			movcf	Ck,Cj,Cm,Cn			Ci←Cj; Cm←Cn;		
			mvclovs	Cm:Cm+1	W:ovs		Cm:Cm+1←CFR.OVS; CFR.OVS; CFR.OVS←0;		
			mvclcas	Cm:Cm+1	W:cas		Cm:Cm+1←CFR.CAS; CFR.CAS; CFR.CAS←0;		
			sethi	Ra,i16s					
ALU max min 列	SIN GLE	字	max	Rc,Ra,Rb			Rc ← max(Ra,Rb)	A	32
	SIN GLE	半字	min	Rc,Ra,Rb	W:c:c1		Rc ← min(Ra,Rb)		
	SIMD	字节	vmaxh	Rc,Ra,Rb					
			vminh	Rc,Ra,Rb					
			vmaxb	Rc,Ra,Rb					
vminb	Rc,Ra,Rb								
ALU abs 列	SIN GLE	半字	abs	Rb,Ra			绝对值	A	32
	absvw		Rb,Ra	W:ovs		有溢出的加法			
	SIMD	半字	fabshv	Rb,Ra	W:ovs				
ALU neg 列	SIN GLE	半字	vabshvh	Rb,Ra	W:ovs			A	32
	negvw		Rb,Ra	W:ovs					
	SIMD	半字	fnegvh	Rb,Ra	W:ovs				
ALU sum 列	S I M D	半字	vnegvh	Rb,Ra	W:ovs			A	32
			vsumh	Rb,Ra					
		vsumh2	Rb,Ra						
		vsumrh2	Rb,Ra						
		字节	vabssumb	Rc,Ra,Rb				C	
ALU 其他	SIN GLE		fmvdvh	Rb,Ra	W:ovs		舍入	C	32
	SI MD		vmvdvh	Rb,Mn	W:ovs				
			vsel	Rc,Ra,Rb	R:VC				
			vsgnh	Rb,Ra					

图24

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16				
CMP	S I N G L E		cmpCCn	Cm,Ra,Rb,Cn Cm,Ra,i05s,Cn Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05s,Cn	W:CF		CC = eq, ne, gt, ge, gtu, geu, le, lt, leu, leu Cm ← result & Cn; (Cm+1 ← ~result & Cn)	A					
			cmpCCa	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05s,Cn	W:CF		Cm ← result & Cn; Cm+1 ← ~(result & Cn);						
			cmpCCo	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05s,Cn	W:CF		Cm ← result Cn; Cm+1 ← ~(result Cn);						
			cmpCC	C6,Ra2,Rb2 C6,Ra2,i04s	W:CF		CC = eq, ne, gt, ge, le, lt C6 ← result						
			tstzn	Cm,Ra,Rb,Cn Cm,Ra,i05u,Cn Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb == 0) & Cn; (Cm+1 ← ~(Ra & Rb == 0) & Cn)						
			tstza	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb == 0) & Cn; Cm+1 ← ~((Ra & Rb == 0) & Cn);						
			tstzo	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb == 0) Cn; Cm+1 ← ~((Ra & Rb == 0) Cn);						
			tstnn	Cm,Ra,Rb,Cn Cm,Ra,i05u,Cn Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb != 0) & Cn; (Cm+1 ← ~(Ra & Rb != 0) & Cn)						
			tstna	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb != 0) & Cn; Cm+1 ← ~((Ra & Rb != 0) & Cn);						
			tstno	Cm:Cm+1,Ra,Rb,Cn Cm:Cm+1,Ra,i05u,Cn	W:CF		Cm ← -(Ra & Rb != 0) Cn; Cm+1 ← ~((Ra & Rb != 0) Cn);						
			tstz	C6,Ra2,Rb2 C6,Ra2,i04u	W:CF		C6 ← (Ra2&Rb2 == 0)						
			tstn	C6,Ra2,Rb2 C6,Ra2,i04u	W:CF		C6 ← -(Ra2&Rb2 != 0)						
			S I M D	半字	vcmpCCh vscmpCCh	Ra,Rb Ra,i05s	W:CF				CC = eq, ne, gt, le, ge, lt		32
				字节	vcmpCCb vscmpCCb	Ra,Rb Ra,i05s	W:CF				CC = eq, ne, gt, le, ge, lt		

图25

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
mul 系列	SINGLE	字×字	mul	Mm,Rc,Ra,Rb Mm,Rb,Ra,i0Bs				X2	32
			mulu	Mm,Rc,Ra,Rb Mm,Rb,Ra,i0Bs			没有符号的乘法		
			fmulww	Mm,Rc,Ra,Rb	fxp	固定小数点运算			
		字×半字	hmul	Mm,Rc,Ra,Rb				X1	
			lmul	Mm,Rc,Ra,Rb					
			fmulhww	Mm,Rc,Ra,Rb	fxp				
		半字×半字	fmulhw	Mm,Rc,Ra,Rb	fxp		fxp	丸めあり	
			fmulhh	Mm,Rc,Ra,Rb	fxp				
			fmulhhr	Mm,Rc,Ra,Rb	fxp				
	SIMD	mul 系列	半字	vmul	Mm,Rc,Ra,Rb				X2
				vfmulw	Mm,Rc,Ra,Rb	fxp			
				vfmulh	Mm,Rc,Ra,Rb	fxp			
				vfmulhr	Mm,Rc,Ra,Rb	fxp	有舍入		
				vxmul	Mm,Rc,Ra,Rb				
				vxfmulw	Mm,Rc,Ra,Rb	fxp			
			半字	vxfmulh	Mm,Rc,Ra,Rb	fxp		有舍入	
				vxfmulhr	Mm,Rc,Ra,Rb	fxp			
				vhmul	Mm,Rc,Ra,Rb				
vhfmulw				Mm,Rc,Ra,Rb	fxp				
vhfmulh				Mm,Rc,Ra,Rb	fxp				
vhfmulhr				Mm,Rc,Ra,Rb	fxp	有舍入			
字×半字	vimul	Mm,Rc,Ra,Rb				有舍入			
	vfmulw	Mm,Rc,Ra,Rb	fxp						
	vfmulh	Mm,Rc,Ra,Rb	fxp						
	vfmulhr	Mm,Rc,Ra,Rb	fxp	有舍入					
	vpfmulhww	Mm,Rc,Rc+1,Ra,Rb Mm,Rc,Rc+1,Ra,Rb	fxp	fxp					

图26

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16		
mac 系列	S I N G L E	字×字	mac	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用mul的积和运算	X2	32		
			fmacww	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用fmulww的积和运算				
		字×半字	hmac	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用hmul的积和运算		X1	
			lmac	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用lmul的积和运算			
		半字×半字	fmachww	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用fmulhww的积和运算			
			fmachw	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用vfmulw的积和运算			
			fmachh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用vxfmulh的积和运算			
		S I M D	半字	半字	fmachhr	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				fxp	有舍入
					vmac	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx					使用vmul的积和运算
				vfmacw	Mm,Rc,Ra,Rb,Mn			fxp		使用vfmulw的积和运算	
	vvmac			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用vhmul的积和运算			
	vxfmacw			Mm,Rc,Ra,Rb,Mn			fxp	使用vfmulw的积和运算			
	vxfmach			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用vxfmulh的积和运算			
	vxfmachr			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	有舍入			
	vhmac			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用vhmul的积和运算			
	vhfmacw			Mm,Rc,Ra,Rb,Mn			fxp	使用vfmulw的积和运算			
	vhfmach			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用vfmulh的积和运算			
	半字	vhfmachr	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx					X2			
		vlmac	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用vmul的积和运算				
		vlfmacw	Mm,Rc,Ra,Rb,Mn			fxp	使用vfmulh的积和运算				
vfmach		Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	使用vfmulh的积和运算					
vfmachr		Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	有舍入					
vlfmach		Mm,Rc,Ra,Rb,Mn Mm,Rc,Ra,Rb,Mn			fxp	使用vfmulh的积和运算					
vlfmachr		Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			fxp	有舍入					
vpfmachww		Mm,Rc:Rc+1,Ra,Rb,Mn			fxp						
字×半字											

图27

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16		
msu 系列	S I N G L E	字×字	msu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用mul的积和运算	X2	32		
			fmsuww	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用fmulww的积和运算				
		字×半字	hmsu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用hmul的积和运算	X1			
			lmsu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用lmul的积和运算				
		半字×半字	fmsuhww	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用fmulhww的积和运算	X1			
			fmsuhw	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vfmulw的积和运算				
			fmsuhh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vxfmulh的积和运算				
		半字	半字	vmsu	vmsu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx				使用vmul的积和运算	X2
					vfmsuw	Mm,Rc,Ra,Rb,Mn		fxp		使用vfmul的积和运算	
				vfmsuh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vfmulh的积和运算			
	vxmsu			Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用xfmulw的积和运算				
	vxfmsuw			Mm,Rc,Ra,Rb,Mn		fxp	使用vxfmulh的积和运算				
	半字		vxfrmsuh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vhmul的积和运算				
			vhmsu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用vhmul的积和运算				
			vhfmsuw	Mm,Rc,Ra,Rb,Mn		fxp	使用vhfmuhw的积和运算				
			vhfmsuh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vfmulh的积和运算				
			vimsu	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx			使用vmul的积和运算				
	半字	vfmsuw	Mm,Rc,Ra,Rb,Mn		fxp	使用vifmulw的积和运算					
		vifmsuh	Mm,Rc,Ra,Rb,Mn M0,Rc,Ra,Rb,Rx		fxp	使用vifmulh的积和运算					

图 28

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
MEM Id 系列	单	字	ld	Rb,(Ra,d10u)				M	32
				Rb,(GP,d13u)					16
				Rb,(SP,d13u)					16
		半字	ldh	Rb,(Ra,d09u)				32	
				Rb,(GP,d12u)				16	
				Rb,(SP,d12u)				16	
		字节	ldb	Rb,(Ra,d08u)				32	
				Rb,(GP,d11u)					
		字节	ldbu	Rb,(Ra,d08u)					
				Rb,(GP,d11u)					
		字节→半字	ldbh	Rb,(Ra,d08u)				32	
				Rb,(GP,d11u)					
字节→半字	ldbuh	Rb,(Ra,d08u)							
		Rb,(GP,d11u)							
双	字	ldp	Rb,Rb+1,(Ra,d11u)				M	32	
			LR:SVR,(Ra,d11u)					32	
			TAR:UDR,(Ra,d11u)					32	
			Rb,Rb+1,(GP,d14u)						
			LR:SVR,(GP,d14u)						
			TAR:UDR,(GP,d14u)						
半字	ldhp	Rb,Rb+1,(Ra,d10u)				32			
		Rb,Rb+1,(GP,d14u)				16			
字节	ldbp	Rb,Rb+1,(Ra,d09u)				32			
		Rb,Rb+1,(GP,d14u)				16			
字节→半字	ldbhp	Rb,Rb+1,(Ra,d09u)				32			
		Rb,Rb+1,(GP,d14u)							
字节→半字	ldbuhp	Rb,Rb+1,(Ra,d09u)							
		Rb,Rb+1,(GP,d14u)							

图29

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16		
MEM store 系列	SINGLE	字	st	(Ra,d10u),Rb (GP,d13u),Rb (SP,d13u),Rb (Ra+);i10s,Rb ----- (Ra2),Rb2 (Ra2,d05u),Rb2 (GP,d06u),Rb2 (SP,d06u),Rb2 (Ra2+),Rb2				M	32		
							16				
		半字	sth	(Ra,d09u),Rb (GP,d12u),Rb (SP,d12u),Rb (Ra+);i09s,Rb ----- (Ra2),Rb2 (Ra2,d04u),Rb2 (GP,d05u),Rb2 (SP,d05u),Rb2 (Ra2+),Rb2				32			
								16			
		字节	stb	(Ra,d08u),Rb (GP,d11u),Rb (SP,d11u),Rb (Ra+);i08s,Rb				M	32		
					字节→半字	stbh	(Ra+);i07s,Rb				
		双	stp	(Ra,d11u),Rb:Rb+1 (Ra,d11u),LR:SVR (Ra,d11u),TAR:UDR (GP,d14u),Rb:Rb+1 (GP,d14u),LR:SVR (GP,d14u),TAR:UDR (SP,d14u),Rb:Rb+1 (SP,d14u),LR:SVR (SP,d14u),TAR:UDR (Ra+);i11s,Rb:Rb+1 ----- (SP,d07u),Rb:Re (SP,d07u),LR:SVR (Ra2+),Rb2:Re2				32			
					半字	sthp	(Ra,d10u),Rb:Rb+1 (Ra+);i10s,Rb:Rb+1 ----- (Ra2+),Rb2:Re2				32
								字节	stbp	(Ra,d09u),Rb:Rb+1 (Ra+);i09s,Rb:Rb+1	
					字节→半字	stbhp	(Ra+);i07s,Rb:Rb+1				

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
BRA			setlr	d09s C5,d09s			设定LR 从LR取出的命令存储在分支用缓冲器中	32	
			settar	d09s C6,d09s C6,C2:C4,d09s C6,Cm,d09s C6,C4,d09s	W:c6 W:c2:c4,c6 W:c6,cm W:c6		设定TAR 从TAR取出的命令存储在分支用缓冲器中		
			setbb	LR TAR			从LR取出的命令存储在分支用缓冲器中 从TAR取出的命令存储在分支用缓冲器中	16	
			jloop	C5,LR,Ra,i08s C6,TAR,Ra,i08s C6,C2:C4,TAR,Ra,i08s C6,Cm,TAR,Ra,i08s C6,TAR,Ra2 C6,C2:C4,TAR,Ra2 C6,Cm,TAR,Ra2	W:c5 W:c6 W:c2:c4,c6 W:c6,cm W:c6 W:c2:c4,c6 W:c6		只是判定[c5] 只是判定[c6]	32	B
			jmp	TAR LR				16	
			jmpI	TAR LR	R:CF				
			jmpf	TAR LR Cm,TAR C6,C2:C4,TAR					
			jmpR	LR					
			br	d20s d09s			只是判定[c6][c7]	32 16	
			brI	d20s d09s	R:CF			32 16	
			rti			W:PSR R:eh			

图 30

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
BS asl 系列	S I N G L E	字	asl	Rc,Ra,Rb Rb,Ra,i05u Ra2,i04u			左移位 $\ll Ra[0:4]$ 	S1	32 16
			faslvw	Rc,Ra,Rb Rb,Ra,i05u Rc,Ra,Rb Rb,Ra,i05u	W:ovs		$\ll Ra[0:4]$ 		
		双字	aslp	Mm,Ra,Mn,Rb Mm,Rb,Mn,i06u Mm,Rc,MHn,Ra,Rb Mm,Rb,MHn,Ra,i06u			$\ll Ra[0:4]$ 	S2	32
			faslpww	Mm,Ra,Mn,Rb Mm,Rb,Mn,i06u	W:ovs		$\ll Ra[0:4]$ 		
	S I M D	字	vasl	Mm,Ra,Mn,Rb Mm,Rb,Mn,i05u				S2	32
			vfaslvw	Mm,Ra,Mn,Rb Mm,Rb,Mn,i05u	W:ovs				
		半字	vaslh	Rc,Ra,Rb Rb,Ra,i04u			$\ll Ra[0:3]$ $\ll Ra[0:3]$ 	S1	16
			vfaslvh	Rc,Ra,Rb Rb,Ra,i04u	W:ovs		$\ll Ra[0:3]$ $\ll Ra[0:3]$ 		
		字节	vaslb	Rc,Ra,Rb Rb,Ra,i03u			$\ll Ra[0:2]$ $\ll Ra[0:2]$ 	S1	16
		BS asr 系列	S I N G L E	字	asr	Rc,Ra,Rb Rb,Ra,i05u Ra2,i04u			算术右移位 $\gg Ra[0:4]$
asrp	Mm,Ra,Mn,Rb Mm,Rb,Mn,i06u Mm,Rc,MHn,Ra,Rb Mm,Rb,MHn,Ra,i06u						$\gg Ra[0:4]$ 		
S I M D	字		vasr	Mm,Ra,Mn,Rb Mm,Rb,Mn,i05u				S2	32
			vasrth	Rc,Ra,Rb Rb,Ra,i04u			$\gg Ra[0:3]$ $\gg Ra[0:3]$ 		
	半字		vasrb	Rc,Ra,Rb Rb,Ra,i03u			$\gg Ra[0:2]$ $\gg Ra[0:2]$ 	S1	16
			vasrb	Rc,Ra,Rb Rb,Ra,i03u			$\gg Ra[0:2]$ $\gg Ra[0:2]$ 		

图31

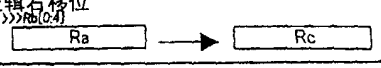
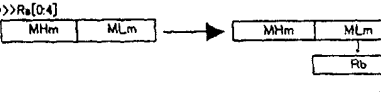
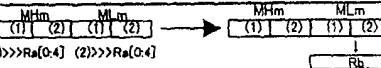
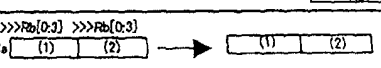
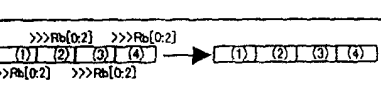
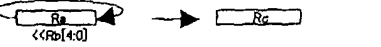
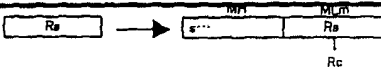
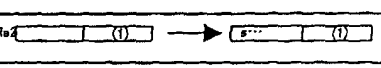
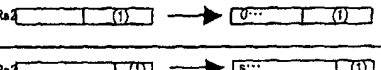
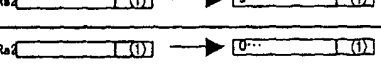
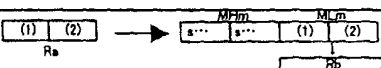
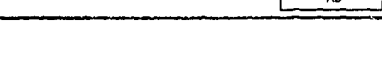
种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
BS lsr 系列	SINGLE	字	lsr	Rc,Ra,Rb Rb,Ra,i05u			逻辑右移位 	S1	32
		双字	lsrp	Mm,Ra,Mn,Rb Mm,Rb,Mn,i06u Mm,Rc,MHn,Ra,Rb Mm,Rb,MHn,Ra,i06u				S2	
	SIMD	字	visr	Mm,Ra,Mn,Rb Mm,Rb,Mn,i05u				S1	
		半字	visrh	Rc,Ra,Rb Rb,Ra,i04u					
		字节	visrb	Rc,Ra,Rb Rb,Ra,i03u					
BS rotate 系列	SINGLE	字	rol	Rc,Ra,Rb Rb,Ra,i05u				S1	32
	SIMD	半字	vrolh	Rc,Ra,Rb Rb,Ra,i04u					
		字节	vrolb	Rc,Ra,Rb Rb,Ra,i03u					
BS ext 系列	SINGLE	字	extw	Mm,Rb,Ra				C	32
		半字	exth	Ra2				S2	
			exthu	Ra2					
		字节	extb	Ra2				S2	
	extbu		Ra2						
	SIMD	半字	vexth	Mm,Rb,Ra				C	32

图32

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
CNV valn 系列	SIMD		valn	Rc,Ra,Rb	R:aln[1:0]		$\ll (CFR.ALN[1:0] \ll 3)$ 	C	32
			valn1	Rc,Ra,Rb					
			valn2	Rc,Ra,Rb					
			valn3	Rc,Ra,Rb					
			valnvc1	Rc,Ra,Rb	R:VC0				
			valnvc2	Rc,Ra,Rb	R:VC0				
			valnvc3	Rc,Ra,Rb	R:VC0				
			valnvc4	Rc,Ra,Rb	R:VC0				

图 33

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16		
CNV	SINGLE		bcnt1	Rb,Ra			计算1的数	C	32		
			bseq0	Rb,Ra			计算从MSB到最初变为0的数				
			bseq1	Rb,Ra			计算从MSB到最初变为1的数				
			bseq	Rb,Ra			计算从MSB-1到最初变为MSB的数				
			mskbrvh	Rc,Ra,Rb	R:BP0						
			byterev	Rb,Ra							
			mskbrvb	Rc,Ra,Rb	R:BP0						
			SIMD	半字	vintlh	Rc,Ra,Rb					
					vintlh	Rc,Ra,Rb					
				字节	vintl	Rc,Ra,Rb					
	vintl	Rc,Ra,Rb									
	半字	vhunpkh		Rb:Rb+1,Ra							
		vhunpkb		Rb:Rb+1,Ra							
		viunpkh		Rb:Rb+1,Ra							
		viunpkb		Rb:Rb+1,Ra							
		字节	vlunpkb	Rb:Rb+1,Ra							
			vlunpkbu	Rb:Rb+1,Ra							
	半字	vunpk1	Rb,Mn								
		vunpk2	Rb,Mn								
		vstovh	Rb,Ra								
	字节	vstovb	Rb,Ra								
		vhpkb	Rc,Ra,Rb								

图34

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
SAT vlpk 系列	S I M D	字→半字	vlpkh	Rc,Ra,Rb				C	32
			vlpkhu	Rc,Ra,Rb					
		半字→字节	vlpkb	Rc,Ra,Rb					
			vlpkbu	Rc,Ra,Rb					
SAT sat 系列	S I N G L E	字	satw	Mm,Rb,Mn			字饱和	C	32
			sath	Rb,Ra			半字饱和		
			satb	Rb,Ra			字节饱和		
			satbu	Rb,Ra			没有字节符号饱和		
			sat9	Rb,Ra			9比特饱和		
			sat12	Rb,Ra			12比特饱和		
	S I M D	半字	vsath	Mm,Rb,Mn					
			vsath8	Rb,Ra			有符号8比特饱和		
			vsath8u	Rb,Ra			没有符号8比特饱和		
			vsath9	Rb,Ra			9比特饱和		
		vsath12	Rb,Ra			12比特饱和			

图35

种类	SIMD	大小	命令	操作数	CFR	PSR	代表性的操作	运算器	31 16
MSK			mskgen	Rc,Rb Rb,i05U,i05u			生成屏蔽 	S2	32
			msk	Rc,Ra,Rb Rb,Ra,i05U,i05u					
EXTR			extr	Rc,Ra,Rb Rb,Ra,i05U,i05u			符号扩展 	S2	32
			extru	Rc,Ra,Rb Rb,Ra,i05U,i05u			(无符号扩展)		
DIV			div divu	MHm,Rc,MHn,Ra,Rb MHm,Rc,MHn,Ra,Rb	W:ovs		除法	DIV	32
			piNl			W:ih,ie,fi,pl R:PSR	软件中断N=0~7	B	32
			piN			W:ih,ie,pl R:PSR	软件中断N=0~7		16
			scN			W:ih,ie,pl R:PSR	系统需求N=0~7		
			ldstb	Rb,(Ra)			load总线锁定	M	32
			rd	Rb,(Ra) Rb,(d11u) Rb2,(Ra2)		R:eee	外部寄存器读数		16
			wt	(Ra),Rb (d11u),Rb (Ra2),Rb2		R:eee	外部寄存器写数		32
			dpref	(Ra,d11u)			预先取出		16
			dbgmn	i18u			N=0~3	DBGM	32
			vcchk		W:CF R:VC		VC标记检查		
			vmpsw				切换VMP		
			vmpsw	LR					
			vmpintd1			Wie	禁止切换VMP		
			vmpintd2			Wie			
			vmpintd3			Wie			
		vmpinte1			Wie	许可切换VMP			
		vmpinte2							
		vmpinte3							
		nop				no operation	A	16	

图36

ld Rb,(Ra,D10)

操作	$Rb \leftarrow W[Ra + (\text{uext}(D10[9:2]) \ll 2)];$ <p>从Ra与显示 值 (D10) 相加后的地址处下载的数据</p>		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(Ra,D10)	32bit	—	PSR.AEE
备注			
<p>D10是没有符号的值，下位2比特请变为0。 $Ra + (\text{uext}(D10[9:2]) \ll 2)$ 不能调节到4个字节则发生错误调节例外。 D10为0时，有时不产生Ra的正向损失。</p>			

图37

ld Rb3,(Ra3,D5)

操作	$Rb3 \leftarrow W[Ra3 + \text{uext}(D5[4:2]) \ll 2];$ <p>从Ra与置换值 (D10) 相加后的地址处下载的数据</p>		
汇编·助记	格式	影响标记	受影响的标记
ld Rb3,(Ra3,D5)	16bit	—	PSR.AEE
备注			
<p>D5是没有符号的值，下位2比特请变为0。 $Ra3 + (\text{uext}(D5[4:2]) \ll 2)$ 不能调节到14个字节则发生错误调节例外。 D5为0时，有时不产生Ra3的正向损失。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图38

ld Rb,(GP,D13)

操作	$Rb \leftarrow W[GP + (\text{uext}(D13[12:2]) \ll 2)];$ <p>从Ra与置换值 (D10) 相加后的地址处下载的数据</p>		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(GP,D13)	32bit	—	PSR.AEE
备注			
<p>D12是没有符号的值，下位1比特请变为0。 $GP + (\text{uext}(D13[12:2]) \ll 2)$ 不能调节到4个字节则发生错误调节例外。 D13为0时，有时不产生GP的正向罚函数。</p>			

图39**ld Rb2,(GP,D6)**

操作	$Rb2 \leftarrow W(GP + (\text{uext}(D6[5:2]) \ll 2));$ <p>从Ra与置换值 (D10) 相加后的地址处下载的数据</p>		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(GP,D6)	16bit	—	PSR.AEE
备注			
<p>D6是没有符号的值，下位2比特请变为0。 $GP + (\text{uext}(D6[5:2]) \ll 2)$ 不能调节到4个字节则发生错误调节例外。 D6为0时，有时不产生GP的正向损失。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图40

ld Rb2,(GP)

操作	\Rightarrow ld Rb2,(GP,0) Rb2 <- W(GP); 从GP所示的地址下载的数据。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(GP)	相当于16bit	—	PSR.AEE
备注			
GP不能调节到4个字节则发生错误调节例外。 *有时不产生GP的正向损失。 可能使用的判定只是C7。(不可进行条件执行)			

图41

ld Rb,(SP,D13)

操作	$Rb <- W[SP+(uext(D13[12:2]) \ll 2)];$ 从SP与置换值(D13)相加后的地址下载数据。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(SP,D13)	32bit	—	—
备注			
D13是没有符号的值，下位2位特请变为0。 GP + (uext(D13[12:2]) << 2) 不能调节到4个字节则发生错误调节例外。 D13为0时，有时不产生SP的正向损失。			

图42

ld Rb2,(SP,D6)

操作	Rb2 <- W(SP+(uext(D6[5:2]) << 2)); 从SP与置换值(D6)相加后的地址处下载字数据。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(SP,D6)	16bit	—	PSR.AEE
备注			
D6是没有符号的值，下位2比特为0。 SP+(uext(D6[5:2]) << 2) 不能调节到4个字节则发生错误调节例外。 D6为0时，有时不产生SP的正向罚函数。 可能使用的判定只是C7。(不可进行条件执行)			

图 43

ld Rb2,(SP)

操作	=> ld Rb2,(SP,0) Rb2 <- W(SP); 从SP所示的地址下载的数据。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(SP)	相当于16bit	—	PSR.AEE
备注			
SP不能调节到4个字节则发生错误调节例外。 有时不产生SP的正向损失。 可能使用的判定只是C7。(不可进行条件执行)			

图 44

图45

ld Rb,(Ra+)I10

操作	Rb <- W(Ra); Ra <- Ra+(sext(I10[9:2]) << 2); 从Ra所示的地址下载字数据。 另外, Ra与定值(I10)相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(Ra+)I10	32bit	—	PSR.AEE
备注			
I10是有符号的值, 下位2比特为0。 I10如果为0时, 不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I10≠0时)。 Ra不能调节到4个字节时, 则发生错误调节例外。			

图46

ld Rb,(Ra+)

操作	=> ld Rb,(Ra+)4 Rb <- W(Ra); Ra <- Ra+sext(4); 从Ra所示的地址下载字数据。 另外, Ra与定值(I10)相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(Ra+)	相当于32bit	—	PSR.AEE
备注			
Rb和Ra相等时的操作是不定的(I10≠0时)。 Ra不能调节到4个字节时, 则发生错误调节例外。			

图47

ld Rb2,(Ra2+)

操作	Rb2 <- W(Ra2); Ra2 <- Ra2+sext(4); 从Ra2所示的地址将字数据符号扩展并下载。 另外, Ra2与4相加, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(Ra2+)	16bit	—	PSR.AEE
备注			
Ra2不能调节到4个字节则发生错误调节例外。 可能使用的判定只是C7。(不可进行条件执行) Rb2和Ra2相等时的操作是不定的。			

图48

ld Rb,(Ra)

操作	=> ld Rb,(Ra+0) Rb <- W(Ra); 从Ra所示的地址下载字数据。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb,(Ra)	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到4个字节则发生错误调节例外。 有时不产生Ra的正向损失。			

ld Rb2,(Ra2)

操作	$Rb2 \leftarrow W(Ra2);$ 从Ra2所示的地址将字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ld Rb2,(Ra2)	16bit	--	PSR.AEE
备注			
Ra2不能调节到4个字节则发生错误调节例外。 可能使用的判定只是C7。(不可进行条件执行)			

图49

ldh Rb,(Ra,D9)

操作	$Rb \leftarrow sext(H[Ra+(uext(D9[8:1]) \ll 1)]);$ 从Ra与置换值(D10)相加后的地址将半字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb,(Ra,D9)	32bit	--	PSR.AEE
备注			
D9是无符号的值，将下位1位为0。 (Ra + (uext(D9[8:1]) << 1) 不能调节到2个字节则发生错误调节例外。 D9为0时，有时不产生Ra的正向损失。			

图50

ldh Rb3,(Ra3,D4)

操作	$Rb3 \leftarrow H[Ra3 + (uext(D4[3:1]) \ll 1)];$ <p>从Ra3与置换值(D4)相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb3,(Ra3,D4)	16bit	—	PSR.AEE
备注			
<p>D4是没有符号的值，下位1比特请变为0。 $Ra3 + (uext(D4[3:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D4为0时，有时不产生Ra3的正向损失。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图51

ldh Rb,(GP,D12)

操作	$Rb \leftarrow sext(H[GP + (uext(D12[11:1]) \ll 1)]);$ <p>从GP与置换值(D12)相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb,(GP,D12)	32bit	—	PSR.AEE
备注			
<p>D12是没有符号的值，下位1比特请变为0。 $GP + (uext(D12[11:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D12为0时，有时不产生GP的正向损失。</p>			

图52

ldh Rb2,(GP,D5)

操作	$Rb2 \leftarrow sext(H(GP+(uext(D5[4:1]) \ll 1)));$ <p>从Ra3与置换值(D10)相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb2,(GP,D5)	16bit	—	PSR.AEE
备注			
<p>D5是没有符号的值，下位1比特请变为0。 $GP+(uext(D5[4:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D5为0时，有时不产生GP的正向损失。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图53

ldh Rb2,(GP)

操作	$\Rightarrow ldh Rb2,(GP,0)$ $Rb2 \leftarrow sext(H(GP));$ <p>从GP与置换值(D10)相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb2,(GP)	相当于16bit	—	PSR.AEE
备注			
<p>GP未调节到2个字节时发生错误调节例外。 有时不产生GP的正向罚函数。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图54

ldh Rb,(SP,D12)

操作	$Rb \leftarrow \text{uext}(H[SP + (\text{uext}(D12[11:1]) \ll 1)]);$ <p>从SP与置换值 (D12) 相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb,(SP,D12)	32bit	—	PSR.AEE
备注			
<p>D12是没有符号的值，下位1比特请变为0。 $SP + (\text{uext}(D12[11:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D12为0时，有时不产生SP的正向损失。</p>			

图55

ldh Rb2,(SP,D5)

操作	$Rb2 \leftarrow \text{sxt}(H[SP + (\text{uext}(D5[4:1]) \ll 1)]);$ <p>从SP与置换值 (D5) 相加后的地址将半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb2,(SP,D5)	16bit	—	PSR.AEE
备注			
<p>D5是没有符号的值，下位1比特请变为0。 $SP + (\text{uext}(D5[4:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D5为0时，有时不产生Ra3的正向损失。 可能使用的判定只是C7。(不可进行条件执行)</p>			

图56

ldh Rb2,(SP)

操作	=> ldh Rb2,(SP,0)		
	Rb2 <- sext(H(SP)); 从SP所示地址将半字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb2,(SP)	相当于16bit	—	PSR.AEE
备注			
SP未调节到2字节时不产生错误排列例外。 ※ 有时不产生SP的正向损失。 可使用的判定只是C7。(不可进行条件执行)			

图57

ldh Rb,(Ra+)I9

操作	Rb <- sext(H(Ra)); Ra <- Ra+(sext(I9[8:1]) << 1);		
	从Ra所示的地址符号扩展半字数据下载。 另外，Ra与当前值(I9)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb,(Ra+)I9	32bit	—	PSR.AEE
备注			
I9是有符号的值，下位1比特为0。 I9如果为0时，不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I9≠0时)。 Ra不能调节到2个字节时，则发生错误调节例外。			

图58

ldh Rb,(Ra+)

操作	=> ldh Rb,(Ra+) Rb <- sext(H(Ra)); Ra <- Ra+sext(2); 从Ra所示的地址将半字数据符号扩展并下载。 另外, Ra与2相加, 存储于Ra。		
	汇编·助记	格式	影响标记
ldh Rb,(Ra+)	相当于32bit	—	PSR.AEE
备注			
Rb和Ra相等时的操作是不定的。 Ra不能调节到2个字节则发生错误调节例外。			

图59

ldh Rb2,(Ra2+)

操作	Rb2 <- sext(H(Ra2)); Ra2 <- Ra2+sext(2); 从SP与置换值(D10)相加后的地址将半字数据符号扩展并下载。		
	汇编·助记	格式	影响标记
ldh Rb2,(Ra2+)	16bit	—	PSR.AEE
备注			
Ra2未调节到2个字节时, 产生错误调节例外。 可能使用的判定只是C7。(不可有条件执行) Rb2和Ra2相等时的操作是不定的。			

图60

ldh Rb,(Ra)

操作	=> ldh Rb,(Ra+0)		
	Rb <- sext(H(Ra)); 从Ra所示地址将半字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldh Rb,(Ra)	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到2个字节则发生错误调节例外。 有时不产生Ra的正向罚函数。			

图61

ldh Rb2,(Ra2)

操作	Rb2 <- sext(H(Ra2)); 从Ra与所示地址将半字数据符号扩展并下载。		
	汇编·助记	格式	影响标记
ldh Rb2,(Ra2)	16bit	—	PSR.AEE
备注			
Ra2不能调节到2个字节则发生错误调节例外。 可能使用的判定只是C7。(不可进行条件执行)			

图62

ldhu Rb,(Ra,D9)

操作	$Rb \leftarrow \text{uext}(H[Ra + (\text{uext}(D9[8:1]) \ll 1)]);$ <p>从Ra与置换值(D9)相加后的地址不进行符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhu Rb,(Ra,D9)	32bit	—	PSR.AEE
备注			
<p>D9是没有符号的值，下位1比特请变为0。 $Ra + (\text{uext}(D9[8:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D9为0时，有时不产生Ra的正向损失。</p>			

图63

ldhu Rb,(GP,D12)

操作	$Rb \leftarrow \text{uext}(H[GP + (\text{uext}(D12[11:1]) \ll 1)]);$ <p>从GP与置换值(D12)相加后的地址将半字数据不进行符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhu Rb,(GP,D12)	32bit	—	PSR.AEE
备注			
<p>D12是没有符号的值，下位1比特请变为0。 $(GP + (\text{uext}(D12[11:1]) \ll 1)$ 不能调节到2个字节则发生错误调节例外。 D12为0时，有时不产生GP的正向损失。</p>			

图64

ldhu Rb,(SP,D12)

操作	$Rb \leftarrow \text{uext}(H[SP+(\text{uext}(D12[11:1]) \ll 1)]);$ 从SP与置换值(D12)相加后的地址下载半字数据， 不进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldhu Rb,(SP,D12)	32bit	—	PSR.AEE
备注			
D12是没有符号的值，下位1比特请变为0。 SP + (uext(D12[11:1]) << 1) 不能调节到2个字节则发生错误调节例外。 D12为0时，有时不产生SP的正向损失。			

图65

ldhu Rb,(Ra+)I9

操作	$Rb \leftarrow \text{uext}(H(Ra)); Ra \leftarrow Ra+(\text{sext}(I9[8:1]) \ll 1);$ 从Ra所示的地址不进行符号扩展。 另外，Ra与定值(I9)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldhu Rb,(Ra+)I9	32bit	—	PSR.AEE
备注			
I9是有符号的值，下位1比特请变为0。 I9如果为0时，不发生向R0的写入。 Rb和Ra相等时的操作是不定的(I9≠0时)。 Ra不能调节到2个字节时，则发生错误调节例外。			

图66

ldhu Rb,(Ra+)

操作	=> ldhu Rb,(Ra+) Rb <- uext(H(Ra)); Ra <- Ra+sext(2); 从Ra所示的地址下载半字数据，不进行符号扩展。 另外，Ra与2相加，存储于Ra。		
	汇编·助记	格式	影响标记
	ldhu Rb,(Ra+)	相当于32bit	—
备注			
Rb2和Ra相等时的操作是不定的。 Ra不能调节到2个字节则发生错误调节例外。			

图67

ldhu Rb,(Ra)

操作	=> ldhu Rb,(Ra+) Rb <- uext(H(Ra)); 从Ra所示的地址下载半字数据，不进行符号扩展。		
	汇编·助记	格式	影响标记
	ldhu Rb,(Ra)	相当于32bit	—
备注			
Ra不能调节到2个字节则发生错误调节例外。 有时不产生Ra的正向损失。			

图68

ldb Rb,(Ra,D8)

操作	Rb <- sext(B[Ra+uext(D8)]); 从Ra与置换值(D8)相加的地址下载字节数据进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(Ra,D8)	32bit	—	—
备注			
D8为无符号的值。 D8为0时，有时不产生Ra的正向损失。			

图69

ldb Rb,(GP,D11)

操作	Rb <- sext(B[GP+uext(D11)]); 从GP与置换值(D11)的地址下载字节数据进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(GP,D11)	32bit	—	—
备注			
D11为无符号的值。 D11为0时，有时不产生GP的正向损失。			

图70

ldb Rb,(SP,D11)

操作	Rb <- sext(B[SP+uext(D11)]); 从SP与置换值(D11)的地址下载字节数据进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(SP,D11)	32bit	—	—
备注			
D11为无符号的值。 D11为0时，有时不产生SP的正向损失。			

图71**ldb Rb,(Ra+)I8**

操作	Rb <- sext(B(Ra)); Ra <- Ra+sext(I8); 从Ra所示的地址下载字节数据，不进行符号扩展。 另外，Ra与当前值(I8)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(Ra+)I8	32bit	—	—
备注			
I8是有符号的值， I8如果为0时，不发生向R4的写入。 Rb和Ra相等时的操作是不定的(I8≠0时)。			

图72

ldb Rb,(Ra+)

操作	=> ldb Rb,(Ra+)1 Rb <- sext(B(Ra)); Ra <- Ra+sext(1); 从Ra所示的地址将字节数据符号扩展并下载。 另外, Ra与1相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(Ra+)	相当于32bit	—	—
备注			
Rb和Ra相等时的操作是不定的。			

图73

ldb Rb,(Ra)

操作	=> ldb Rb,(Ra)0 Rb <- sext(B(Ra)); 从Ra所示的地址下载字节数据进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldb Rb,(Ra)	相当于32bit	—	—
备注			
有时不产生Ra的正向损失。			

图74

ldbu Rb,(Ra,D8)

操作	$Rb \leftarrow \text{uext}(B[Ra + \text{uext}(D8)]);$ 从Ra所示的置换值(D8)相加后的地址下载字节数据, 不进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(Ra,D8)	32bit	—	—
备注			
D8为无符号的值。 D8为0时, 有时不产生Ra的正向损失。			

图75

ldbu Rb,(GP,D11)

操作	$Rb \leftarrow \text{uext}(B[GP + \text{uext}(D11)]);$ 从GP所示的置换值(D11)相加后的地址下载字节数据, 不进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(GP,D11)	32bit	—	—
备注			
D11为有符号的值, 下位2比特变为0。 D11为0时, 有时不产生GP的正向损失。			

图76

ldbu Rb,(SP,D11)

操作	Rb <- uext(B[SP+uext(D11)]); 从SP与置换值(D8)相加后的地址下载字节数据， 不进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(SP,D11)	32bit	—	—
备注			
D11为无符号的值。 D11为0时，有时不产生SP的正向损失。			

图77

ldbu Rb,(Ra+)I8

操作	Rb <- uext(B(Ra)); Ra <- Ra+sext(I8); 从Ra所示的地址下载字节数据，不进行符号扩展。 另外，Ra与当前值(I8)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(Ra+)I8	32bit	—	—
备注			
I8是有符号的值， I8如果为0时，不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I8≠0时)。			

图78

ldbu Rb,(Ra+)

操作	=> ldbu Rb,(Ra+)		
	Rb <- uext(B(Ra)); Ra <- Ra+sext(1); 从Ra所示的地址下载字节数据，不进行符号扩展。 另外，Ra与1相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(Ra+)	相当于32bit	—	PSR.AEE
备注			
Rb和Ra相等时的操作是不定的。			

图79

ldbu Rb,(Ra)

操作	=> ldbu Rb,(Ra+0)		
	Rb <- uext(B(Ra)); 从Ra所示的地址下载字节数据，不进行符号扩展。		
汇编·助记	格式	影响标记	受影响的标记
ldbu Rb,(Ra)	相当于32bit	—	—
备注			
有时不产生Ra的正向损失。			

图80

ldp Rb:Rb+1,(Ra,D11)

操作	$Rb:Rb+1 \leftarrow D(Ra+(uext(D11[10:3]) \ll 3));$ 从Ra与替换值(D11)相加后的地址将2个数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(Ra,D11)	32bit	—	PSR.AEE
备注			
<p>D11为无符号的值，下位3比特变为0。 Rb变为偶数编号。 $Ra+(uext(D11[10:3]) \ll 3)$ 不能调节到8个字节时则发生错误调节例外。 D11为0时，有时不产生Ra的正向损失。</p> <p>$Rb:Rb+1 \leftarrow D(Ra+(uext(D11[10:3]) \ll 3))$ 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- Ra + (uext(D11[10:3]) << 3) Rb <- W(effective_address) Rb+1 <- W(effective_address+4) </pre> <p>与优先顺序无关地固定。</p>			

图 81

ldp Rb:Rb+1,(GP,D14)

操作	$Rb:Rb+1 \leftarrow D(GP+(uext(D14[13:3]) \ll 3));$ 从GP与替换值(D14)相加后的地址将2个字数据符号扩展并下载。		
汇程·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(GP,D14)	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。 Rb变为偶数编号。 $GP+(uext(D14[13:3]) \ll 3)$ 不能调节到8个字节时则发生错误调节例外。 D14为0时，有时不产生GP的正向损失。</p> <p>$Rb:Rb+1 \leftarrow D(GP+(uext(D14[13:3]) \ll 3))$ 的正确含义如下。</p> <p style="padding-left: 40px;">$effective_address \leftarrow GP + (uext(D14[13:3]) \ll 3)$ $Rb \leftarrow W(effective_address)$ $Rb+1 \leftarrow W(effective_address+4)$ 与优先顺序无关地固定。</p>			

图82

ldp Rb:Rb+1,(SP,D14)

操作	Rb:Rb+1 <- D(SP+(uext(D14[13:3]) << 3)); 从SP与替换值 (D14) 相加后的地址将2个字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(SP,D14)	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。</p> <p>Rb变为偶数编号。</p> <p>SP+(uext(D14[13:3]) << 3): 不能调节到8个字节时则发生错误调节例外。</p> <p>D14为0时，有时不产生SP的正向损失。</p> <p>Rb:Rb+1 <- D(SP+(uext(D14[13:3]) << 3)) 的正确含义如下。</p> <p style="padding-left: 40px;">effective_address <- SP + (uext(D14[13:3]) << 3)</p> <p style="padding-left: 40px;">Rb <- W(effective_address)</p> <p style="padding-left: 40px;">Rb+1 <- W(effective_address+4)</p> <p style="padding-left: 40px;">与优先顺序无关地固定。</p>			

图83

ldp Rb:Rb+1,(SP,D7)

操作	Rb:Rb+1 <- D(SP+(uext(D7[6:3]) << 3)); 从SP与替换值(D7)相加后的地址将2个字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(SP,D7)	16bit	—	PSR.AEE
备注			
<p>D7为无符号的值，下位3比特变为0。 Rb变为偶数编号。 SP+(uext(D7[6:3]) << 3) 不能调节到8个字节时则发生错误调节例外。 D7为0时，有时不产生SP的正向损失。</p> <p>Rb:Rb+1 <- D(SP+(uext(D7[6:3]) << 3)) 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- SP+(uext(D7[6:3]) << 3); Rb <- W(effective_address); Rb+1 <- W(effective_address+4); </pre> <p>与优先顺序无关地固定。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图84

ldp Rb:Rb+1,(SP)

操作	<p>=> ldp Rb:Rb+1,(SP,0)</p> <p>Rb:Rb+1 <- D(SP);</p> <p>从SP所示的地址将2个字节数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(SP)		—	PSR.AEE
备注			
<p>Rb变为偶数编号。</p> <p>SP不能调节到8个字节时则发生错误调节例外。</p> <p>有时不产生SP的正向损失。</p> <p>Rb:Rb+1 <- D(SP) 的正确含义如下。</p> <p style="padding-left: 40px;">effective_address <- SP;</p> <p style="padding-left: 40px;">Rb <- W(effective_address);</p> <p style="padding-left: 40px;">Rb+1 <- W(effective_address+4);</p> <p style="padding-left: 40px;">与优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图85

ldp Rb:Rb+1,(Ra+)I11

操作	$Rb:Rb+1 \leftarrow D(Ra);$ $Ra \leftarrow Ra + (sxt(I11[10:3]) \ll 3);$ 从Ra所示的地址将2个半数据符号扩展并下载。 另外, Ra与当前值(I11)相加, 存储于Ra中。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(Ra+)I11	32bit	—	PSR.AEE
备注			
<p>I11为有符号的值, 下位3比特变为0。 Rb变为偶数编号。 I11为0时, 不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I11≠0时)。 Rb+1和Ra相等时的操作是不定的(I11≠0时)。 Ra不能调节到8个字节时, 则发生错误调节例外。 $Rb:Rb+1 \leftarrow D(Ra)$ 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address ← Ra Rb ← W(effective_address) Rb+1 ← W(effective_address+4) </pre> <p>与优先顺序无关地固定。</p>			

图86

ldp Rb2:Rb2+1,(Ra2+)

操作	Rb2:Rb2+1 <- D(Ra2); Ra2 <- Ra2 + sext(8); 从Ra2所示的地址将2个字数据符号扩展并下载。 另外，Ra2与8相加，存储于Ra中。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb2:Rb2+1,(Ra2+)	16bit	—	PSR.AEE
备注			
Rb2变为偶数编号。 Rb2和Ra2相等时的操作是不定的。 Rb2+1和Ra2相等时的操作是不定的。 Ra2不能调节到8个字节时，则发生错误调节例外。 Rb2:Rb2+1 <- D(Ra2) 的正确含义如下。 effective_address <- Ra2 Rb2 <- W(effective_address) Rb2+1 <- W(effective_address+4) 与优先顺序无关地固定。 可能使用的判定只是C7。(不能进行条件执行)			

图87

ldp Rb:Rb+1,(Ra+)

操作	=> ldp Rb:Rb+1,(Ra+) 8		
	Rb:Rb+1 <- D(Ra); Ra <- Ra+sext(8); 从Ra所示的地址将2个数据符号扩展并下载。 另外, Ra与8相加, 存储于Ra中。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(Ra+)	相当于32bit	—	PSR.AEE
备注			
Rb变为偶数编号。 Ra不能调节到8个字节时, 则发生错误调节例外。			

图 88

ldp Rb:Rb+1,(Ra)

操作	\Rightarrow ldp Rb:Rb+1,(Ra+0) Rb:Rb+1 <- D(Ra); 从Ra所示的地址将2个字数据符号扩展并下载。		
汇编·助记	格式	影响标记	受影响的标记
ldp Rb:Rb+1,(Ra)	相当于32bit	—	PSR.AEE
备注			
<p>※: Rb变为偶数编号。 Ra不能调节到8个字节时则发生错误调节例外。 有时不产生Ra的正向损失。 Rb:Rb+1 <- D(Ra) 的正确含义如下。</p> <p style="text-align: center;"> $\text{effective_address} \leftarrow \text{Ra}$ $\text{Rb} \leftarrow \text{W}(\text{effective_address})$ $\text{Rb+1} \leftarrow \text{W}(\text{effective_address}+4)$ 与优先顺序无关地固定。 </p>			

图89

ldp LR:SVR,(Ra,D14)

操作	$\text{tmp}[63:0] \leftarrow D(\text{Ra} + (\text{uext}(\text{D11}[10:3]) \ll 3));$ $\text{LR} \leftarrow \text{tmp}[63:32];$ $\text{SVR} \leftarrow \text{tmp}[15:0];$ <p>从Ra与替换值(D11)相加的地址下载2个字数据。存储于LR和SVR。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(Ra,D14)	32bit	—	PSR.AEE
备注			
<p>D11为无符号的值，使下3比特为0。 废弃tmp[31:16]。 $\text{Ra} + (\text{uext}(\text{D11}[10:3]) \ll 3)$ 未调节成8字节时产生错误调节例外。 D11为0时，有时不产生Ra的正向损失。 LR:SVR $\leftarrow D(\text{Ra} + (\text{uext}(\text{D11}[10:3]) \ll 3))$ 的正确含义如下。</p> $\text{effective_address} \leftarrow \text{Ra} + (\text{uext}(\text{D11}[10:3]) \ll 3)$ $\text{LR} \leftarrow W(\text{effective_address})$ $\text{tmp} \leftarrow W(\text{effective_address} + 4)$ $\text{SVR} \leftarrow \text{tmp}[15:0];$ <p>与优先顺序无关地固定。</p>			

图90

ldp LR:SVR,(Ra)

操作	<p>=> ldp LR:SVR,(Ra,0)</p> <p>tmp[63:0] <- D(Ra); LR <- tmp[63:32]; SVR <- tmp[15:0];</p> <p>从Ra所示的地址下载2个字数据，存储于LR和SVR。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(Ra)	相当于32bit	—	PSR.AEE
备注			
<p>※ 废弃tmp[31:16]。</p> <p>Ra未调节成8字节时产生错误调节例外。</p> <p>有时不产生Ra的正向损失。</p> <p>LR:SVR <- D(Ra) 的正确含义如下。</p> <p style="padding-left: 40px;">effective_address <- Ra</p> <p style="padding-left: 40px;">LR <- W(effective_address)</p> <p style="padding-left: 40px;">tmp <- W(effective_address+4)</p> <p style="padding-left: 40px;">SVR <- tmp[15:0];</p> <p style="padding-left: 40px;">与优先顺序无关地固定。</p>			

图91

ldp LR:SVR,(GP,D14)

操作	$tmp[63:0] \leftarrow D(GP + (uext(D14[13:3]) \ll 3));$ $LR \leftarrow tmp[63:32];$ $SVR \leftarrow tmp[15:0];$ 从GP与替换值(D14)相加的地址下载2个字数据。存储于LR和SVR。		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(GP,D14)	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，使下位3比特为0。 废弃tmp[31:16]。 $GP + (uext(D14[13:3]) \ll 3)$，未调节成8字节时产生错误调节例外。 D14为0时，有时不产生GP的正向损失。</p> <p>LR:SVR $\leftarrow D(GP + (uext(D14[13:3]) \ll 3))$ 的正确含义如下。</p> $effective_address \leftarrow GP + (uext(D14[13:3]) \ll 3)$ $LR \leftarrow W(effective_address)$ $tmp \leftarrow W(effective_address + 4)$ $SVR \leftarrow tmp[15:0];$ 与优先顺序无关地固定。			

图92

ldp LR:SVR,(SP,D14)

操作	<pre>tmp[63:0] <- D(SP+(uext(D14[13:3]) << 3)); LR <- tmp[63:32]; SVR <- tmp[15:0];</pre> <p>从SP与替换值(D14)相加的地址下载2个字数据，存储于LR和SVR。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(SP,D14)	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，使下位3比特为0。</p> <p>废弃tmp[31:16]。</p> <p>SP+(uext(D14[13:3]) << 3)未调节成8字节时产生错误调节例外。</p> <p>D14为0时，有时不产生SP的正向损失。</p> <p>LR:SVR <- D(SP+(uext(D14[13:3]) << 3)) 的正确含义如下。</p> <pre>effective_address <- SP + (uext(D14[13:3]) << 3) LR <- W(effective_address) tmp <- W(effective_address+4) SVR <- tmp[15:0];</pre> <p>与优先顺序无关地固定。</p>			

图93

ldp LR:SVR,(SP,D7)

操作	$tmp[63:0] \leftarrow D(SP+(uext(D7[6:3]) \ll 3));$ $LR \leftarrow tmp[63:32];$ $SVR \leftarrow tmp[15:0];$ 从SP与替换值(D7)相加的地址下载2个字数据, 存储于LR和SVR。		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(SP,D7)	16bit	SVR	PSR.AEE
备注			
<p>D7为无符号的值, 使下位3比特为0。 废弃tmp[31:16]。 $SP+(uext(D7[6:3]) \ll 3)$ 未调节成8字节时产生错误调节例外。 D7为0时, 有时不产生SP的正向损失。</p> <p>LR:SVR $\leftarrow D(SP+(uext(D7[6:3]) \ll 3))$ 的正确含义如下。</p> $effective_address \leftarrow SP+(uext(D7[6:3]) \ll 3);$ $LR \leftarrow W(effective_address);$ $tmp[31:0] \leftarrow W(effective_address+4);$ $SVR \leftarrow tmp[15:0];$ <p>与优先顺序无关地固定。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图94

ldp LR:SVR,(SP)

操作	=> ldp LR:SVR,(SP,0) tmp[63:0] <- D(SP); LR <- tmp[63:32]; SVR <- tmp[15: 0]; 从SP与置换值(D7)相加的地址下载2个字数据, 存储于LR和SVR。		
汇编·助记	格式	影响标记	受影响的标记
ldp LR:SVR,(SP)	相当于16bit	SVR	PSR.AEE
备注			
<p>※ 废弃tmp[31:16]。</p> <p>SP未调节成8字节时产生错误调节例外。</p> <p>有时不产生SP的正向损失。</p> <p>LR:SVR <- D(SP) 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- SP; LR <- W(effective_address); tmp[31:0] <- W(effective_address+4); SVR <- tmp[15:0]; </pre> <p>与优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图95

ldp TAR:UDR,(Ra,D11)

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
ldp TAR:UDR,(Ra,D11)	32bit	—	PSR.AEE
备注			

图96**ldp TAR:UDR,(GP,D14)**

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
ldp TAR:UDR,(GP,D14)	32bit	—	—
备注			

图97

ldp TAR:UDR,(SP,D14)

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
ldp TAR:UDR,(SP,D14)	32bit	—	—
备注			

图98

ldhp Rb:Rb+1,(Ra,D10)

操作	<pre>tmp <- W(Ra+(uext(D10[9:2]) << 2)); Rb <- sext(tmp[31:16]); Rb+1 <- sext(tmp[15: 0]);</pre> <p>从Ra与替换值(D10)相加后的地址将2个半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhp Rb:Rb+1,(Ra,D10)	32bit	—	PSR.AEE
备注			
<p>D10为无符号的值，下位2比特变为0。 Rb变为偶数编号。 $Ra+(uext(D10[9:2]) << 2)$ 不能调节到8个字节时则发生错误调节例外。 D10为0时，有时不产生Ra的正向损失。</p>			

图99

ldhp Rb:Rb+1,(Ra+)I10

操作	<pre>tmp <- W(Ra); Rb <- sext(tmp[31:16]); Rb+1 <- sext(tmp[15: 0]); Ra <- Ra+(sext(I10[9:2]) << 2);</pre> <p>从Ra所示的地址将2个半字数据符号扩展并下载。</p> <p>另外，Ra与当前值(I10)相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhp Rb:Rb+1,(Ra+)I10	32bit	—	PSR.AEE
备注			
<p>I10为有符号的值，下位2比特变为0。</p> <p>Rb变为偶数编号。</p> <p>I10为0时，不发生向Ra的写入。</p> <p>Rb和Ra相等时的操作是不定的(I10≠0时)。</p> <p>Rb+1和Ra相等时的操作是不定的(I10≠0时)。</p> <p>Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图100

ldhp Rb2:Rb2+1,(Ra2+)

操作	<pre> tmp <- W(Ra2); Rb2 <- sext(tmp[31:16]); Rb2+1 <- sext(tmp[15: 0]); Ra2 <- Ra2 + sext(4); </pre> <p>从Ra2所示的地址将2个半字数据符号扩展并下载。 另外，Ra与4相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhp Rb2:Rb2+1,(Ra2+)	16bit	—	PSR.AEE
备注			
<p>Rb2变为偶数编号。 Rb2和Ra2相等时的操作是不定的。 Rb2+1和Ra2相等时的操作是不定的。 Ra2不能调节到4个字节时，则发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图101

图 102

ldhp Rb:Rb+1, (Ra+)

操作	=> ldhp Rb:Rb+1,(Ra+)		
	<pre> tmp <- W(Ra); Rb <- sext(tmp[31:16]); Rb+1 <- sext(tmp[15: 0]); Ra <- Ra+sext(4); </pre> <p>从Ra所示的地址将2个半字数据符号扩展并下载。</p> <p>另外, Ra与4相加, 存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhp Rb:Rb+1,(Ra+)		—	PSR.AEE
备注			
<p>Rb变为偶数编号。</p> <p>Ra不能调节到4个字节时, 则发生错误调节例外。</p>			

图 103

ldhp Rb:Rb+1,(Ra)

操作	=> ldhp Rb:Rb+1,(Ra)		
	<pre> tmp <- W(Ra); Rb <- sext(tmp[31:16]); Rb+1 <- sext(tmp[15: 0]); </pre> <p>从Ra所示的地址将2个半字数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldhp Rb:Rb+1,(Ra)	相当于32bit	—	PSR.AEE
备注			
<p>Rb变为偶数编号。</p> <p>Ra不能调节到4个字节时, 则发生错误调节例外。</p> <p>有时不产生Ra的正向损失。</p>			

图104

ldbp Rb:Rb+1,(Ra,D9)

操作	$\text{tmp} \leftarrow \text{H}(\text{Ra} + (\text{uext}(\text{D9}[8:1]) \ll 1))$ $\text{Rb} \leftarrow \text{sext}(\text{tmp}[15:8]);$ $\text{Rb}+1 \leftarrow \text{sext}(\text{tmp}[7:0]);$ <p>从Ra与替换值(D9)相加后的地址将2个字节数据符号扩展并下载。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbp Rb:Rb+1,(Ra,D9)	32bit	—	PSR.AEE
备注			
<p>D9为无符号的值，下位1比特变为0。 Rb变为偶数编号。 $\text{Ra} + (\text{uext}(\text{D9}[8:1]) \ll 1)$ 不能调节到2个字节时则发生错误调节例外。 D9为0时，有时不产生Ra的正向损失。</p>			

图105

ldbp Rb:Rb+1,(Ra+)I9

操作	$\text{tmp} \leftarrow \text{H}(\text{Ra});$ $\text{Rb} \leftarrow \text{sext}(\text{tmp}[15:8]);$ $\text{Rb}+1 \leftarrow \text{sext}(\text{tmp}[7:0]);$ $\text{Ra} \leftarrow \text{Ra} + (\text{sext}(\text{I9}[8:1]) \ll 1);$ <p>从Ra所示的地址将2个字节数据符号扩展并下载。 另外，Ra与当前值(I9)相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbp Rb:Rb+1,(Ra+)I9	32bit	—	PSR.AEE
备注			
<p>I9为有符号的值，下位1比特变为0。 Rb变为偶数编号。 I9为0时，不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I9≠0时)。 Rb+1和Ra相等时的操作是不定的(I9≠0时)。 Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图106

ldbp Rb:Rb+1,(Ra+)

操作	=> ldbp Rb:Rb+1,(Ra+2)		
	tmp <- H(Ra); Rb <- sext(tmp[15: 8]); Rb+1 <- sext(tmp[7: 0]); Ra <- Ra+sext(2);		
从Ra所示的地址将2个字节数据符号扩展并下载。			
另外，Ra与2相加，存储于Ra中。			
汇编·助记	格式	影响标记	受影响的标记
ldbp Rb:Rb+1,(Ra+)	相当于32bit	—	PSR.AEE
备注			
Rb变为偶数编号。			
Ra不能调节到3个字节时，则发生错误调节例外。			

图107

ldbp Rb:Rb+1,(Ra)

操作	=> ldbp Rb:Rb+1,(Ra+0)		
	tmp <- H(Ra); Rb <- sext(tmp[15: 8]); Rb+1 <- sext(tmp[7: 0]);		
从Ra所示的地址将2个字节数据符号扩展并下载。			
汇编·助记	格式	影响标记	受影响的标记
ldbp Rb:Rb+1,(Ra,D9)		—	PSR.AEE
备注			
Rb变为偶数编号。			
Ra不能调节到2个字节时则发生错误调节例外。			
有时不产生Ra的正向损失。			

ldbh Rb,(Ra+)I7

操作	<pre>tmp <- H(Ra); Rb[31:16] <- sext(tmp[15: 8]); Rb[15: 0] <- sext(tmp[7: 0]); Ra <- Ra+(sext(I7[6:1]) << 1);</pre> <p>从Ra所示的地址下载2个字节数据。数据的上位8比特下位8比特分别进行符号扩展，存储于Rb的16比特·下位16比特。</p> <p>另外，Ra与当前值(I7)相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbh Rb,(Ra+)I7	32bit	—	PSR.AEE
备注			
<p>I7为有符号的值，下位1比特变为0。</p> <p>I7为0时，不发生向Ra的写入。</p> <p>Rb和Ra相等时的操作是不定的(I7≠0时)。</p> <p>Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图108

图109

ldbh Rb,(Ra+)

操作	=> ldh Rb,(Ra+)		
	<pre> tmp <- H(Ra); Rb[31:16] <- sext(tmp[15: 8]); Rb[15: 0] <- sext(tmp[7: 0]); Ra <- Ra+sext(2); </pre> <p>从Ra所示的地址下载2个字节数据。分别进行符号扩展， 存储于Rb的上位16比特·下位16比特。</p> <p>另外，Ra与2相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbh Rb,(Ra+)	相当于32bit	—	PSR.AEE
备注			
<p>Rb和Ra相等时的操作是不定的。</p> <p>Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图110

ldbh Rb,(Ra)

操作	=> ldh Rb,(Ra+0)		
	<pre> tmp <- H(Ra); Rb[31:16] <- sext(tmp[15: 8]); Rb[15: 0] <- sext(tmp[7: 0]); </pre> <p>从Ra所示的地址下载2个字节数据，分别进行符号扩展， 存储于Rb的上位16比特·下位16比特。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbh Rb,(Ra)	相当于32bit	—	PSR.AEE
备注			
<p>Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图 111

ldbuh Rb,(Ra+)I7

操作	<pre>tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); Rb+1[31:16] <- uext(tmp[15: 8]); Rb+1[15: 0] <- uext(tmp[7: 0]); Ra <- Ra+(sext(I7[6:2] << 2);</pre> <p>从Ra所示的地址下载2个字节数据，数据的上位8比特·下位8比特分别进行符号扩展，</p> <p>存储于Rb的上位16比特·下位16比特。</p> <p>另外，Ra与当前值(I7)相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbuh Rb,(Ra+)I7	32bit	—	PSR.AEE
备注			
<p>I7为有符号的值，下位1比特变为0。</p> <p>I7为0时，不发生向Ra的写入。</p> <p>Rb和Ra相等时的操作是不定的(I7≠0)。</p> <p>Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图112

ldbuh Rb,(Ra+)

操作	=> ldbuh Rb,(Ra+) 2		
	tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); Ra <- Ra+(sext(2)); 从Ra所示的地址下载2个字节数据，分别不进行符号扩展， 存储于Rb的上位16比特·下位16比特。 另外，Ra与2相加，存储于Ra中。		
汇编·助记	格式	影响标记	受影响的标记
ldbuh Rb,(Ra+)	相当于32bit	—	PSR.AEE
备注			
* Rb和Ra相等时的操作是不定的。 Ra不能调节到2个字节时，则发生错误调节例外。			

图113

ldbuh Rb,(Ra)

操作	=> ldbuh Rb,(Ra) 0		
	tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); 从Ra所示的地址下载2个字节数据，分别不进行符号扩展， 存储于Rb的上位16比特·下位16比特。		
汇编·助记	格式	影响标记	受影响的标记
ldbuh Rb,(Ra)	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到2个字节时，则发生错误调节例外。			

ldbhp Rb:Rb+1,(Ra+)I7

操作	<pre> tmp <- W(Ra); Rb [31:16] <- sext(tmp[31:24]); Rb [15: 0] <- sext(tmp[23:16]); Rb+1[31:16] <- sext(tmp[15: 8]); Rb+1[15: 0] <- sext(tmp[7: 0]); Ra <- Ra+(sext(I7[6:2]) << 2); </pre> <p>从Ra所示的地址下载4个字节数据，数据的上位8比特·下位8比特分别进行符号扩展， 存储于Rb和Rb+1的上位比特。下位16比特。 另外，Ra与当前值(I7)相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbhp Rb:Rb+1,(Ra+)I7	32bit	—	PSR.AEE
备注			
<p>I7为有符号的值，下位1比特变为0。 请将Rb变为偶数编号。 I7为0时，不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(I7≠0时)。 Rb+1和Ra相等时的操作是不定的(I7≠0时)。 Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图114

ldbhp Rb:Rb+1,(Ra+)

操作	=> ldbhp Rb:Rb+1,(Ra+)		
	<pre> tmp <- W(Ra); Rb [31:16] <- sext(tmp[31:24]); Rb [15: 0] <- sext(tmp[23:16]); Rb+1[31:16] <- sext(tmp[15: 8]); Rb+1[15: 0] <- sext(tmp[7: 0]); Ra <- Ra+sext(4); </pre> <p>从Ra所示的地址下载4个字节数据，分别进行符号扩展， 存储于Rb和Rb+1的上位16比特·下位16比特。 另外，Ra与4相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbhp Rb:Rb+1,(Ra+)	相当于32bit	—	PSR.AEE
备注			
<p>请将Rb变为偶数编号。 Rb和Ra相等时的操作是不定的。 Rb+1和Ra相等时的操作是不定的。 Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图115

ldbhp Rb:Rb+1,(Ra)

操作	=> ldbhp Rb:Rb+1,(Ra+0)		
	<pre> tmp <- W(Ra); Rb [31:16] <- sext(tmp[31:24]); Rb [15: 0] <- sext(tmp[23:16]); Rb+1[31:16] <- sext(tmp[15: 8]); Rb+1[15: 0] <- sext(tmp[7: 0]); Ra <- Ra+(sext(l[7[6:2] << 2); </pre> <p>从Ra所示的地址下载4个字节数据，分别进行符号扩展， 存储于Rb和Rb+1的上位16比特·下位16比特。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbhp Rb:Rb+1,(Ra)	相当于32bit	—	PSR.AEE
备注			
<p>※ 请将Rb变为偶数编号。</p> <p>Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图116

ldbuhp Rb:Rb+1,(Ra+)*I*7

操作	<pre> tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); Rb+1[31:16] <- uext(tmp[15: 8]); Rb+1[15: 0] <- uext(tmp[7: 0]); Ra <- Ra+(sext(<i>I</i>7[6:2]) << 2); </pre> <p>从Ra所示的地址下载4个字节数据，2个半字数据的上位8比特· 下位8比特分别进行符号扩展， 存储于Rb和Rb+1的上位16比特。下位16比特。 另外，Ra与2相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbuhp Rb:Rb+1,(Ra+)<i>I</i>7	32bit	—	PSR.AEE
备注			
<p><i>I</i>7为有符号的值，下位2比特变为0。 <i>I</i>7为0时，不发生向Ra的写入。 Rb和Ra相等时的操作是不定的(<i>I</i>7≠0时)。 Rb+1和Ra相等时的操作是不定的(<i>I</i>7≠0时)。 Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图117

ldbuhp Rb:Rb+1,(Ra+)

操作	<pre> => ldbuhp Rb:Rb+1,(Ra+4) tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); Rb+1[31:16] <- uext(tmp[15: 8]); Rb+1[15: 0] <- uext(tmp[7: 0]); Ra <- Ra+sxt(4); </pre> <p>从Ra所示的地址下载2个字节数据，分别不进行符号扩展，存储于Rb的上位16比特·下位16比特。</p> <p>另外，Ra与4相加，存储于Ra中。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbuhp Rb:Re,(Ra+)	相当于32bit	—	PSR.AEE
备注			
<p>※ Rb和Ra相等时的操作是不定的。</p> <p>※ Rb+1和Ra相等时的操作是不定的。</p> <p>Ra不能调节到4个字节时，则发生错误调节例外。</p>			

图118

图119

ldbuhp Rb:Rb+1,(Ra)

操作	=> ldbuhp Rb:Rb+1,(Ra)+0		
	<pre> tmp <- W(Ra); Rb [31:16] <- uext(tmp[31:24]); Rb [15: 0] <- uext(tmp[23:16]); Rb+1[31:16] <- uext(tmp[15: 8]); Rb+1[15: 0] <- uext(tmp[7: 0]); </pre> <p>从Ra所示的地址下载2个字节数据，分别不进行符号扩展， 存储于Rb的上位16比特·下位16比特。</p>		
汇编·助记	格式	影响标记	受影响的标记
ldbuhp Rb:Rb+1,(Ra)	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到4个字节时，则发生错误调节例外。			

图120

st (Ra,D10),Rb

操作	W[Ra+(uext(D10[9:2]) << 2)] <- Rb;		
	Ra与置换值(D10)相加后的地址上存储存储于Rb的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra,D10),Rb	32bit	—	PSR.AEE
备注			
D10为无符号的值，下位2比特变为0。			
Ra+(uext(D10[9:2]) << 2) 不调节到4个字节时，则发生错误调节例外。			
D10为0时，不产生Ra的正向损失。			

st (Ra3,D5),Rb3

操作	$W[Ra3+(uext(D5[4:2]) \ll 2)] \leftarrow Rb3;$ <p>Ra3置换值 (D5) 相加后的地址上存储存储于Rb的字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
st (Ra3,D5),Rb3	16bit	—	PSR.AEE
备注			
<p>D5为无符号的值，下位2比特变为0。</p> <p>$Ra3+(uext(D5[4:2]) \ll 2)$ 不调节到4个字节时，则发生错误调节例外。</p> <p>D5为0时，不产生Ra3的正向损失。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图121

st (GP,D13),Rb

操作	$W[GP+(uext(D13[12:2]) \ll 2)] \leftarrow Rb;$ <p>GP与置换值 (D13) 相加后的地址上存储存储于Rb的半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
st (GP,D13),Rb	32bit	—	PSR.AEE
备注			
<p>D13为无符号的值，下位2比特变为0。</p> <p>$GP+(uext(D13[12:2]) \ll 2)$ 不调节到4个字节时，则发生错误调节例外。</p> <p>D13为0时，不产生GP的正向损失。</p>			

图122

st (GP,D6),Rb2

操作	$W(GP+(uext(D6[5:2]) \ll 2)) \leftarrow Rb2;$ GP与置换值(D6)相加后的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (GP,D6),Rb2	16bit	—	PSR.AEE
备注			
D6为无符号的值，下位2比特变为0。 GP+(uext(D6[5:2]) << 2): 不调节到4个字节时，则发生错误调节例外。 D6为0时，不产生GP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图123

st (GP),Rb2

	$\Rightarrow st (GP,0),Rb2$ $W(GP) \leftarrow Rb2;$ GP所示的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (GP),Rb2	16bit シノニム	—	PSR.AEE
备注			
GP不能调节到4个字节时，发生错误调节例外。 不产生GP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图124

st (SP,D13),Rb

操作	$W[SP+(uext(D13[12:2]) \ll 2)] \leftarrow Rb;$ SP与置换值(D13)相加后的地址上存储存储于Rb的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (SP,D13),Rb	32bit	—	PSR.AEE
备注			
D13为无符号的值，下位2比特变为0。 SP+(uext(D13[12:2]) << 2) 不调节到4个字节时，则发生错误调节例外。 D13为0时，不产生SP的正向损失。			

图125

st (SP,D6),Rb2

操作	$W(SP+(uext(D6[5:2]) \ll 2)) \leftarrow Rb2;$ SP与置换值(D6)相加后的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (SP,D6),Rb2	16bit	—	PSR.AEE
备注			
SP+(uext(D6[5:2]) << 2) 不调节到4个字节时，则发生错误调节例外。 D6为0时，不产生SP的正向损失。 D6为无符号的值，下位2比特变为0。 可能使用的判定只是C7。(不能进行条件执行)			

图126

st (SP),Rb2

操作	=> st (SP,0),Rb2		
	W(SP) <- Rb2; SP所示的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (SP),Rb2	相当于16bit	—	PSR.AEE
备注			
SP不能调节到4个字节时，发生错误调节例外。 ※ 不产生SP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图127

st (Ra+)I10,Rb

操作	W[Ra] <- Rb; Ra <- Ra+(sext(I10[9:2]) << 2);		
	Ra2所示的地址上存储存储于Rb2的字数据。 另外，Ra2与当前值(I10)相加存储于R2a。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra+)I10,Rb	32bit	—	PSR.AEE
备注			
I10为有符号的值，下位2比特变为0。 I10为不发生向Ra的写入。 Ra不能调节到4个字节时，发生错误调节例外。			

图128

st (Ra+),Rb

操作	=> st (Ra+)4,Rb		
	W(Ra) <- Rb; Ra <- Ra+sext(4); Ra所示的地址上存储存储于Rb的字数据。 另外, Ra与4相加存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra+),Rb		—	PSR.AEE
备注			
Ra不能调节到4个字节时, 发生错误调节例外。			

图129

st (Ra2+),Rb2

操作	W(Ra2) <- Rb2; Ra2 <- Ra2 + sext(4);		
	Ra2所示的地址上存储存储于Rb2的字数据。 另外, Ra2与4相加存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra2+),Rb2	16bit	—	PSR.AEE
备注			
Ra2不能调节到4个字节时, 发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)			

图130

st (Ra),Rb

操作	\Rightarrow st (Ra+0),Rb W(Ra) <- Rb; Ra <- Ra; Ra所示的地址上存储存储于Rb的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra),Rb	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到4个字节时，发生错误调节例外。 不产生Ra的正向损失。			

图131

st (Ra2),Rb2

操作	W(Ra2) <- Rb2; Ra2所示的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
st (Ra2),Rb2	16bit	—	PSR.AEE
备注			
Ra2不能调节到4个字节时，发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)			

图132

sth (Ra,D9),Rb

操作	$H[Ra+(uext(D9[8:1]) \ll 1)] \leftarrow Rb[15:0];$ <p>Ra与置换值(D9)相加后的地址上存储存储于Rb的半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra,D9),Rb	32bit	—	PSR.AEE
备注			
<p>D9为无符号的值，下位1比特变为0。 $Ra+(uext(D9[8:1]) \ll 1)$ 不调节到2个字节时，则发生错误调节例外。 D9为0时，不产生Ra的正向损失。</p>			

图133

sth (Ra3,D4),Rb3

操作	$H[Ra3+(uext(D4[3:1]) \ll 1)] \leftarrow Rb3;$ <p>Ra3与置换值(D4)相加后的地址上存储存储于Rb3的半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra3,D4),Rb3	16bit	—	PSR.AEE
备注			
<p>D4为无符号的值，下位1比特变为0。 $Ra3+(uext(D4[3:1]) \ll 1)$ 不调节到2个字节时，则发生错误调节例外。 D4为0时，不产生Ra3的正向损失。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图134

sth (GP,D12),Rb

操作	$H[GP+(uext(D12[11:1]) \ll 1)] \leftarrow Rb[15:0];$ <p>GP与置换值(D12)相加后的地址上存储存储于Rb的半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
sth (GP,D12),Rb	32bit	—	PSR.AEE
备注			
<p>D12为无符号的值，下位1比特变为0。 $GP+(uext(D12[11:1]) \ll 1)$ 不调节到2个字节时，则发生错误调节例外。 D12为0时，不产生GP的正向损失。</p>			

图135

sth (GP,D5),Rb2

操作	$H[GP+(uext(D5[4:1]) \ll 1)] \leftarrow Rb2[15:0];$ <p>GP与置换值(D5)相加后的地址上存储存储于R2的半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
sth (GP,D5),Rb2	16bit	—	PSR.AEE
备注			
<p>D5为无符号的值，下位1比特变为0。 $GP+(uext(D5[4:1]) \ll 1)$ 不调节到2个字节时，则发生错误调节例外。 D5为0时，不产生GP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图136

sth (GP),Rb2

操作	\Rightarrow sth (GP,0),Rb2 H(GP) <- Rb2[15:0]; GP所示的地址上存储存储于Rb2的字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (GP),Rb2	相当于16bit	—	PSR.AEE
备注			
GP不能调节到2个字节时，发生错误调节例外。 ※ 不产生GP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图137

sth (SP,D12),Rb

操作	$\mathbf{H[SP+(uext(D12[11:1]) \ll 1)] \leftarrow Rb[15:0];}$ SP与置换值(D12)相加后的地址上存储存储于Rb的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (SP,D12),Rb	32bit	—	PSR.AEE
备注			
D12为无符号的值，下位1比特变为0。 SP+(uext(D12[11:1]) << 1)不调节到2个字节时，则发生错误调节例外。 D12为0时，不产生SP的正向损失。			

图138

sth (SP,D5),Rb2

操作	$H(SP+(uext(D5[4:1]) \ll 1)) \leftarrow Rb2[15:0];$ SP与置换值(D5)相加后的地址上存储存储于Rb2的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (SP,D5),Rb2	16bit	—	PSR.AEE
备注			
D5为无符号的值，下位1比特变为0。 SP+(uext(D5[4:1]) << 1) 不调节到2个字节时，则发生错误调节例外。 D5为0时，不产生SP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图 139

sth (SP),Rb2

操作	$\Rightarrow sth (SP,0),Rb2$ $H(SP) \leftarrow Rb2[15:0];$ SP所示的地址上存储存储于Rb2的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (SP),Rb2	相当于16bit	—	PSR.AEE
备注			
SP不能调节到2个字节时，发生错误调节例外。 ※ 不产生SP的正向损失。 可能使用的判定只是C7。(不能进行条件执行)			

图 140

sth (Ra+)I9,Rb

操作	$H[Ra] \leftarrow Rb[15:0]; Ra \leftarrow Ra + (\text{sext}(I9[8:1]) \ll 1);$ Ra所示的地址上存储存储于Rb的半字数据。 另外, Ra与当前值(I9)相加存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra+)I9,Rb	32bit	—	PSR.AEE
备注			
I9为时, 不发生向Ra的写入。 Ra不能调节到2个字节时, 发生错误调节例外。 I9为有符号的值, 下位1比特变为0。			

图 141

sth (Ra+),Rb

操作	$\Rightarrow \text{sth } (Ra+),Rb$ $H(Ra) \leftarrow Rb[15:0]; Ra \leftarrow Ra + \text{sext}(2);$ Ra所示的地址上存储存储于Rb的半字数据。 另外, Ra与2相加存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra+),Rb	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到2个字节时, 发生错误调节例外。			

图 142

sth (Ra2+),Rb2

操作	H(Ra2) <- Rb2[15:0]; Ra2 <- Ra2 + sext(2); Ra2所示的地址上存储存储于Rb2的半字数据。 另外，Ra2与2相加存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra2+),Rb2	16bit	—	PSR.AEE
备注			
Ra2不能调节到2个字节时，发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)			

图143

sth (Ra),Rb

操作	=> sth (Ra+),Rb H(Ra) <- Rb[15:0]; 在Ra所示的地址存储在Rb中存储的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra),Rb	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到2个字节时，发生错误调节例外。 不产生Ra的正向损失。			

图144

sth (Ra2),Rb2

操作	H(Ra2) <- Rb2[15:0]; Ra2所示地址上存储存储于Rb2的半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sth (Ra2),Rb2	16bit	—	PSR.AEE
备注			
Ra2不能调节到2个字节时，发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)			

图145

stb (Ra,D8),Rb

操作	B[Ra+uext(D8)] <- Rb[7:0]; Ra与置换值(D8)相加后的地址上存储存储于Rb的字节数据。		
汇编·助记	格式	影响标记	受影响的标记
stb (Ra,D8),Rb	32bit	—	—
备注		备注	
D8为无符号的值。 D8为0时，不产生Ra的正向损失。			

图146

stb (GP,D11),Rb

操作	B[GP+uext(D11)] <- Rb[7:0]; GP与置换值(D11)相加后的地址上存储存储于Rb的字节数据。		
汇编·助记	格式	影响标记	受影响的标记
stb (GP,D11),Rb	32bit	—	—
备注			
D11为无符号的值。 D11为0时, 不产生GP的正向损失。			

图147

stb (SP,D11),Rb

操作	B[SP+uext(D11)] <- Rb[7:0]; SP与置换值(D11)相加后的地址上存储存储于Rb的字节数据。		
汇编·助记	格式	影响标记	受影响的标记
stb (SP,D11),Rb	32bit	—	—
备注			
D11为无符号的值。 D11为0时, 不产生SP的正向损失。			

图148

stb (Ra+)I8,Rb

操作	B[Ra] <- Rb[7:0]; Ra <- Ra+sext(I8); Ra所示的地址上存储存储于Rb的字节数据。 另外, Ra与当前值(I8)相加存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stb (Ra+)I8,Rb	32bit	—	—
备注			
I8为有符号的值。 I8为0时, 不发生向Ra的写入。			

图149

stb (Ra+),Rb

操作	=> stb (Ra+)1,Rb B(Ra) <- Rb[7:0]; Ra <- Ra+sext(1); Ra所示的地址上存储存储于Rb的字节数据。 另外, Ra与1相加存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stb (Ra+),Rb	相当于32bit	—	—
备注			

图150

图151

stb (Ra),Rb

操作	$\Rightarrow \text{stb (Ra+0),Rb}$ $\text{B(Ra)} \leftarrow \text{Rb}[7:0];$ 在Ra所示的地址上存储在Rb中存储的字节数据。		
汇编·助记	格式	影响标记	受影响的标记
stb (Ra),Rb	相当于32bit	—	—
备注			
不产生Ra的正向损失。			

图152

stp (Ra,D11),Rb:Rb+1

操作	$\text{D(Ra+(uext(D11[10:3]) \ll 3))} \leftarrow \text{Rb:Rb+1};$ Ra与置换值(D11)相加的地址上存储存储于Rb和Rb+1的2个字数据。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra,D11),Rb:Rb+1	32bit	—	PSR.AEE
备注			
D11为无符号的值，下位3比特变为0。 请将Rb变为偶数编号。 $\text{Ra+(uext(D11[10:3]) \ll 3)}$ 不能调节到8个字节时，发生错误调节例外。 D11为0时，不产生Ra的正向损失。 $\text{D(Ra+(uext(D11[10:3]) \ll 3))} \leftarrow \text{Rb:Rb+1}$ 的正确含义如下。 $\text{effective_address} \leftarrow \text{Ra+(uext(D11[10:3]) \ll 3);}$ $\text{W(effective_address)} \leftarrow \text{Rb};$ $\text{W(effective_address+4)} \leftarrow \text{Rb+1};$ 和优先顺序无关地固定。			

图 153

stp (GP,D14),Rb:Rb+1

操作	$D(GP+(uext(D14[13:3]) \ll 3)) \leftarrow Rb:Rb+1;$ GP与置换值(D14)相加的地址上存储存储于Rb和Rb+1的2个字数据。		
汇编·助记	格式	影响标记	受影响的标记
stp (GP,D14),Rb:Rb+1	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。 请将Rb变为偶数编号。 $GP+(uext(D14[13:3]) \ll 3)$: 不能调节到8个字节时，发生错误调节例外。 D14为0时，不产生GP的正向损失。</p> <p>$D(GP+(uext(D14[13:3]) \ll 3)) \leftarrow Rb:Rb+1$ 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address ← GP+(uext(D14[13:3]) << 3); W(effective_address) ← Rb; W(effective_address+4) ← Rb+1; 和优先顺序无关地固定。 </pre>			

图154

stp (SP,D14),Rb:Rb+1

操作	$D(SP+(uext(D14[13:3]) \ll 3)) \leftarrow Rb:Rb+1;$ SP与置换值(D14)相加的地址上存储存储于Rb和Rb+1的2个字数据。		
汇编·助记	格式	影响标记	受影响的标记
stp (SP,D14),Rb:Rb+1	32bit	—	PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。 请将Rb变为偶数编号。 $SP+(uext(D14[13:3]) \ll 3)$; 不能调节到8个字节时，发生错误调节例外。 D14为0时，不产生SP的正向损失。</p> <p>$D(SP+(uext(D14[13:3]) \ll 3)) \leftarrow Rb:Rb+1$ 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address ← SP+(uext(D14[13:3]) << 3); W(effective_address) ← Rb; W(effective_address+4) ← Rb+1; </pre> 和优先顺序无关地固定。			

图155

stp (SP,D7),Rb:Rb+1

操作	$D(SP+(uext(D7[6:3]) \ll 3)) \leftarrow Rb:Rb+1;$ <p>SP与置换值(D7[6:3])相加的地址上存储存储于Rb和Rb+1的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (SP,D7),Rb:Rb+1	16bit	—	PSR.AEE
备注			
<p>D7为无符号的值，下位3比特变为0。</p> <p>请将Rb变为偶数编号。</p> <p>$SP+(uext(D7[6:3]) \ll 3)$ 不能调节到8个字节时，发生错误调节例外。</p> <p>D7为0时，不产生SP的正向损失。</p> <p>$D(SP+(uext(D7[6:3]) \ll 3)) \leftarrow Rb:Rb+1$ 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address ← SP+(uext(D7[6:3]) << 3); W(effective_address) ← Rb; W(effective_address+4) ← Rb+1; </pre> <p>和优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图156

stp (SP),Rb:Rb+1

操作	<p>=> stp (SP,0),Rb:Rb+1</p> <p>D(SP) <- Rb:Rb+1;</p> <p>SP所示的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (SP),Rb:Rb+1	相当于16bit	—	PSR.AEE
备注			
<p>※ 请将Rb变为偶数编号。</p> <p>SP不能调节到8个字节时，则发生错误调节例外。</p> <p>* 不产生SP的正向损失。</p> <p>D(SP) <- Rb:Rb+1 的正确含义如下。</p> <p style="padding-left: 40px;">effective_address <- SP;</p> <p style="padding-left: 40px;">W(effective_address) <- Rb;</p> <p style="padding-left: 40px;">W(effective_address+4) <- Rb+1;</p> <p style="padding-left: 40px;">和优先顺序无关地固定。</p> <p>可能使用的判定只是07。(不能进行条件执行)</p>			

图157

stp (Ra+)I11,Rb:Rb+1

操作	D(Ra) <- Rb:Rb+1; Ra <- Ra+(sxt(I11[10:3]) << 3); Ra所示的地址上存储存储于Rb和Rb+1的2个字数据。 另外，Ra与当前值(I11)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra+)I11,Rb:Rb+1	32bit	—	PSR.AEE
备注			
I11为有符号的值，下位3比特变为0。 请将Rb变为偶数编号。 I11为0时，不发生向Ra的写入。 Ra不能调节到8个字节时，则发生错误调节例外。 D(Ra) <- Rb:Rb+1 的正确含义如下。 effective_address <- Ra W(effective_address) <- Rb W(effective_address+4) <- Rb+1 和优先顺序无关地固定。			

图 158

stp (Ra+),Rb:Rb+1

操作	=> stp (Ra+),Rb:Rb+1		
	D(Ra) <- Rb:Rb+1; Ra <- Ra+(8); Ra所示的地址上存储存储于Rb和Rb+1的2个字数据。 另外，Ra与8相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra+),Rb:Rb+1	相当于32bit	—	PSR.AEE
备注			
※ 请将Rb变为偶数编号。 Ra不能调节到8个字节时，则发生错误调节例外。 D(Ra) <- Rb:Rb+1 的正确含义如下。 effective_address <- Ra W(effective_address) <- Rb W(effective_address+4) <- Rb+1 和优先顺序无关地固定。			

图 159

stp (Ra2+),Rb2:Rb2+1

操作	D(Ra2) <- Rb2:Rb2+1; Ra2 <- Ra2 + sext(8); Ra所示的地址上存储存储于Rb和Rb+1的2个字数据。 另外，Ra与8相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra2+),Rb2:Rb2+1	16bit	—	PSR.AEE
备注			
<p>请将Rb2变为偶数编号。</p> <p>Ra2不能调节到8个字节时，则发生错误调节例外。</p> <p>D(Ra2) <- Rb2:Rb2+1 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- Ra2 W(effective_address) <- Rb2 W(effective_address+4) <- Rb2+1 </pre> <p>和优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图 160

stp (Ra),Rb:Rb+1

操作	\Rightarrow stp (Ra+0),Rb:Rb+1 D(Ra) <- Rb:Rb+1; Ra所示的地址上存储存储于Rb和Rb+1的2个字数据。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra),Rb:Rb+1	相当于32bit	—	PSR.AEE
备注			
<p>※ 请将Rb变为偶数编号。</p> <p>Ra不能调节到8个字节时，则发生错误调节例外。</p> <p>不产生Ra的正向损失。</p> <p>D(Ra) <- Rb:Rb+1 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- Ra W(effective_address) <- Rb W(effective_address+4) <- Rb+1 和优先顺序无关地固定。 </pre>			

图161

stp (Ra,D11),LR:SVR

操作	<pre>tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(Ra+(uext(D11[10:3]) << 3)) <- tmp[63:0];</pre> <p>Ra与置换值(D11)相加后的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra,D11),LR:SVR	32bit	—	SVR,PSR.AEE
备注			
<p>D11为无符号的值，下位3比特变为0。</p> <p>Ra+(uext(D11[10:3]) << 3) 不能调节到8个字节时，发生错误调节例外。</p> <p>D11为0时，不产生Ra的正向损失。</p> <p>D(Ra+(uext(D11[10:3]) << 3)) <- LR:SVR 的正确含义如下。</p> <pre>effective_address <- Ra+(uext(D11[10:3]) << 3); W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp[31:0];</pre> <p>和优先顺序无关地固定。</p>			

图 162

stp (Ra),LR:SVR

操作	<p>=> stp (Ra,0),LR:SVR</p> <p>tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(Ra) <- tmp[63:0];</p> <p>Ra所示的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra),LR:SVR	相当于32bit	—	PSR.AEE
备注			
<p>Ra不能调节到8个字节时，则发生错误调节例外。 不产生Ra的正向损失。 D(Ra) <- LR:SVR 的正确含义如下。</p> <pre style="margin-left: 40px;"> effective_address <- Ra; W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp[31:0]; </pre> <p>和优先顺序无关地固定。</p>			

图 163

stp (GP,D14),LR:SVR

操作	<pre>tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(GP+(uext(D14[13:3]) << 3)) <- tmp[63:0];</pre> <p>GP与置换值(D14)相加的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (GP,D14),LR:SVR	32bit	—	SVR,PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。</p> <p>GP+(uext(D14[13:3]) << 3) 不能调节到8个字节时，则发生错误调节例外。</p> <p>D14为0时，不产生GP的正向损失。</p> <p>D(GP+(uext(D14[13:3]) << 3)) <- LR:SVR 的正确含义如下。</p> <pre>effective_address <- GP+(uext(D14[13:3]) << 3); W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp[31:0];</pre> <p>和优先顺序地固定。</p>			

图164

stp (SP,D14),LR:SVR

操作	<pre> tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(SP+(uext(D14[13:3]) << 3)) <- tmp[63:0]; </pre> <p>SP与置换值(D14)相加的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (SP,D14),LR:SVR	32bit	—	SVR,PSR.AEE
备注			
<p>D14为无符号的值，下位3比特变为0。</p> <p>SP+(uext(D14[13:3]) << 3) 不能调节到8个字节时，则发生错误调节例外。</p> <p>D14为0时，不产生SP的正向损失。</p> <p>D(SP+(uext(D14[13:3]) << 3)) <- LR:SVR 的正确含义如下。</p> <pre> effective_address <- SP+(uext(D14[13:3]) << 3); W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp[31:0]; </pre> <p>和优先顺序无关地固定。</p>			

图 165

stp (SP,D7),LR:SVR

操作	<pre>tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(SP+(uext(D7[6:3]) << 3)) <- tmp[63:0];</pre> <p>SP与置换值 (D14) 相加的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (SP,D7),LR:SVR	16bit	—	SVR,PSR.AEE
备注			
<p>D7为无符号的值，下位3比特变为0。</p> <p>SP+(uext(D7[6:3]) << 3) 不能调节到8个字节时，则发生错误调节例外。</p> <p>D7为0时，不产生SP的正向损失。</p> <p>D(SP+(uext(D7[6:3]) << 3)) <- LR:SVR 的正确含义如下。</p> <pre>effective_address <- Ra+(uext(D7[6:3]) << 3); W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp;</pre> <p>和优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图 166

stp (SP),LR:SVR

操作	<p>=>stp (SP,0),LR:SVR</p> <p>tmp[63:32] <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; D(SP) <- tmp[63:0];</p> <p>SP所示的地址上存储存储于LR和SVR的2个字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stp (SP),LR:SVR	相当于16bit	—	SVR,PSR.AEE
备注			
<p>SP不能调节到8个字节时，则发生错误调节例外。 ※ 不产生SP的正向损失。</p> <p>D(SP) <- LR:SVR 的正确含义如下。</p> <p style="padding-left: 40px;">effective_address <- SP; W(effective_address) <- LR; tmp[31:16] <- 0x00; tmp[15: 0] <- SVR; W(effective_address+4) <- tmp;</p> <p style="padding-left: 40px;">和优先顺序无关地固定。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图167

stp (Ra,D11),TAR:UDR

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
stp (Ra,D11),TAR:UDR	32bit	—	—
备注			

图168

stp (GP,D14),TAR:UDR

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
stp (GP,D14),TAR:UDR	32bit	—	—
备注			

图 169

stp (SP,D14),TAR:UDR

操作	未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
stp (SP,D14),TAR:UDR	32bit	—	—
备注			

图 170

sthp (Ra,D10),Rb:Rb+1

操作	$W(Ra+(uext(D10[9:2]) \ll 2)) \leftarrow Rb[15:0]:Rb+1[15:0];$ <p>Ra与置换值(D10)相加的地址上存储存储于Rb和Rb+1的2个半字数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
sthp (Ra,D10),Rb:Rb+1	32bit	—	PSR.AEE
备注			
<p>D10为无符号的值，下位2比特变为0。</p> <p>请将Rb变为偶数编号。</p> <p>$Ra+(uext(D10[9:2]) \ll 2)$ 不能调节到4个字节时，发生错误调节例外。</p> <p>D10为0时，不产生Ra的正向损失。</p>			

图171

sthp (Ra+)I10,Rb:Rb+1

操作	W(Ra) <- Rb[15:0]:Rb+1[15:0]; Ra <- Ra+(sext(I10[9:2]) << 2); Ra所示的地址上存储存储于Ra和Rb+1的2个半字数据。 另外, Ra与当前值(I10)相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
sthp (Ra+)I10,Rb:Rb+1	32bit	—	PSR.AEE
备注			
I10为有符号的值, 下位2比特变为0。 请将Rb变为偶数编号。 I10为0时, 不发生向Ra的写入。 Ra不能调节到4个字节时, 则发生错误调节例外。			

图172

sthp (Ra+),Rb:Rb+1

操作	=> sthp (Ra+)4,Rb:Rb+1 W(Ra) <- Rb[15:0]:Rb+1[15:0]; Ra <- Ra+sext(4); Ra所示的地址上存储存储于Rb和Rb+1的2个半字数据。 另外, Ra与4相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
sthp (Ra+),Rb:Rb+1	相当于32bit	—	PSR.AEE
备注			
※ 请将Rb变为偶数编号。 Ra不能调节到4个字节时, 则发生错误调节例外。			

sthp (Ra2+),Rb2:Rb2+1

操作	W(Ra2) <- Rb2[15:0]:Rb2+1[15:0]; Ra2 <- Ra2 + sext(4); Ra2所示的地址上存储存储于Rb和Rb+1的2个半字数据。 另外，Ra与4相加，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
sthp (Ra2+),Rb2:Rb2+1	16bit	—	PSR.AEE
备注			
请将Rb变为偶数编号。 Ra2不能调节到4个字节时，则发生错误调节例外。 可能使用的判定只是C7。(不能进行条件执行)			

图173

sthp (Ra),Rb:Rb+1

操作	=> sthp (Ra+)0,Rb:Rb+1 W(Ra) <- Rb[15:0]:Rb+1[15:0]; Ra所示的地址上存储于Rb和Rb+1的2个半字数据。		
汇编·助记	格式	影响标记	受影响的标记
sthp (Ra),Rb:Rb+1		—	PSR.AEE
备注			
※ 请将Rb变为偶数编号。 Ra不能调节到4个字节时，则发生错误调节例外。 不产生Ra的正向损失。			

图174

stbp (Ra,D9),Rb:Rb+1

操作	$H(Ra+(uext(D9[8:1]) \ll 1)) \leftarrow Rb[7:0]:Rb+1[7:0];$ <p>Ra与置换值(D9)相加的地址上存储存储于Rb和Rb+1的2个字节数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
stbp (Ra,D9),Rb:Rb+1	32bit	—	PSR.AEE
备注			
<p>D9为无符号的值，下位1比特变为0。 请将Rb变为偶数编号。 $Ra+(uext(D9[8:1]) \ll 1)$ 不能调节到2个字节时，发生错误调节例外。 D9为0时，不产生Ra的正向损失。</p>			

图175

stbp (Ra+I9),Rb:Rb+1

操作	$H(Ra) \leftarrow Rb[7:0]:Rb+1[7:0];$ $Ra \leftarrow Ra+(sext(I9[8:1]) \ll 1);$ <p>Ra所示的地址上存储存储于Rb和Rb+1的2个字节数据。</p> <p>另外，Ra与当前值(I9)相加，存储于Ra。</p>		
汇编·助记	格式	影响标记	受影响的标记
stbp (Ra+I9),Rb:Rb+1	32bit	—	PSR.AEE
备注			
<p>I9为有符号的值，下位1比特变为0。 请将Rb变为偶数编号。 I9为0时，不发生向Ra的写入。 Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图176

stbp (Ra+),Rb:Rb+1

操作	=> stbp (Ra+),Rb:Rb+1		
	H(Ra) <- Rb[7:0]:Rb+1[7:0]; Ra <- Ra+(sxt(2)); Ra所示的地址上存储存储于Rb和Rb+1的2个字节数据。 另外, Ra与2相加, 存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stbp (Ra+),Rb:Rb+1		—	PSR.AEE
备注			
※ 请将Rb变为偶数编号。 Ra不能调节到2个字节时, 则发生错误调节例外。			

图 177

stbp (Ra),Rb:Rb+1

操作	=> stbp (Ra),Rb:Rb+1		
	H(Ra) <- Rb[7:0]:Rb+1[7:0]; Ra所示的地址上存储存储于Rb和Rb+1的2个字节数据。		
汇编·助记	格式	影响标记	受影响的标记
stbp (Ra),Rb:Rb+1		—	PSR.AEE
备注			
※ 请将Rb变为偶数编号。 Ra不能调节到2个字节时, 则发生错误调节例外。 不产生Ra的正向损失。			

图 178

stbh (Ra+)I7,Rb

操作	$H(Ra) \leftarrow Rb[23:16]:Rb[7:0];$ $Ra \leftarrow Ra + (\text{sext}(I7[6:1]) \ll 1);$		
<p>Ra所示的地址上存储存储于Rb和Rb+1的2个字节数据。 2个字节数据存储于Rb的16~23比特及0~7比特中。</p> <p>另外，Ra与当前值(I7)相加，存储于Ra。</p>			
汇编·助记	格式	影响标记	受影响的标记
stbh (Ra+)I7,Rb	32bit	—	PSR.AEE
备注			
<p>I7为有符号的值，下位1比特变为0。 I7为0时，不发生向Ra的写入。 Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图179

stbh (Ra+),Rb

操作	$\Rightarrow \text{stbh (Ra+)2,Rb}$		
$H(Ra) \leftarrow Rb[23:16]:Rb[7:0];$ $Ra \leftarrow Ra + \text{sext}(2);$			
<p>Ra所示的地址上存储存储于Rb和Rb+1的2个字节数据。 2个字节数据存储于Rb的16~23比特及0~7比特中。</p> <p>另外，Ra与2相加，存储于Ra。</p>			
汇编·助记	格式	影响标记	受影响的标记
stbh (Ra+),Rb		—	PSR.AEE
备注			
<p>Ra不能调节到2个字节时，则发生错误调节例外。</p>			

图180

stbh (Ra),Rb

操作	=> stbh (Ra+0),Rb		
	H(Ra) <- Rb[23:16]:Rb[7:0]; 所示的地址上存储存储于Rb和Rb+1的2个字节数据。 2个字节数据存储于Rb的16~23比特及0~7比特中。		
汇编·助记	格式	影响标记	受影响的标记
stbh (Ra),Rb	相当于32bit	—	PSR.AEE
备注			
Ra不能调节到2个字节时，则发生错误调节例外。			

图 181**stbhp (Ra+)I7,Rb:Rb+1**

操作	W(Ra) <- Rb[23:16]:Rb[7:0]:Rb+1[23:16]:Rb+1[7:0]; Ra <- Ra+(sext(I7[6:2]) << 2);		
	Ra所示的地址上存储存储于Rb和Rb+1的4个字节数据。 4个字节数据存储于Rb的16~23比特及0~7比特中。 另外，Ra与当前值(I7)相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stbhp (Ra+)I7,Rb:Rb+1	32bit	—	PSR.AEE
备注			
I7为有符号的值，下位1比特变为0。			
I7为0时，不发生向Ra的写入。			
Ra不能调节到4个字节时，则发生错误调节例外。			

图 182

stbhp (Ra+),Rb:Rb+1

操作	=> stbhp (Ra+),4,Rb:Rb+1		
	W(Ra) <- Rb[23:16]:Rb[7:0]:Rb+1[23:16]:Rb+1[7:0]; Ra <- Ra+sext(4); Ra所示的地址上存储存储于Rb和Rb+1的4个字节数据。 4个字节数据存储于Rb的16~23比特及0~7比特中。 另外，Ra与4相加，存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
stbhp (Ra+),Rb:Rb+1		—	PSR.AEE
备注			
Ra不能调节到2个字节时，则发生错误调节例外。			

图183

stbhp (Ra),Rb:Rb+1

操作	=> stbhp (Ra+),0,Rb:Rb+1		
	W(Ra) <- Rb[23:16]:Rb[7:0]:Rb+1[23:16]:Rb+1[7:0]; Ra所示的地址上存储存储于Rb和Rb+1的4个字节数据。 4个字节数据存储于Rb的16~23比特及0~7比特中。		
汇编·助记	格式	影响标记	受影响的标记
stbhp (Ra),Rb:Rb+1		—	PSR.AEE
备注			
Ra不能调节到4个字节时，则发生错误调节例外。			

图184

dpref (Ra,D8)

操作	从Ra与置换值(D8)相加后的地址取出数据。 未安装命令。		
汇编·助记	格式	影响标记	受影响的标记
dpref (Ra,D8)	32bit	—	—
备注			
★ 研究中。			

图 185

ldstb Rb,(Ra)

操作	<pre>tmp[31:0] <- sext(B(Ra,bus_lock)); B(Ra,bus_unlock) <- Rb; Rb <- tmp[31:0];</pre> 锁住总线，交换Rb的值和Ra所示的地址存储器的值。 完成后解除总线的锁住。。		
汇编·助记	格式	影响标记	受影响的标记
ldstb Rb,(Ra)	32bit	—	—
备注			
命令执行时，禁止VMP切换。			

图 186

rd C0:C1,Rb,(D11)

操作	<pre> if (EREG(D11) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(D11) if (EREG_ERR == 1) { Rb <- UD; C0:C1 <- 1:0; } else { Rb <- tmp; C0:C1 <- 0:1 } } </pre> <p>当前值(D11)作为外部寄存器编号。读取其外部寄存器的值。 读入的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C0:C1,Rb,(D11)	32bit	C0,C1	—
备注			
D11为无符号的值。			

图187

rd C0:C1,Rb,(Ra,D5)

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra+uext(D5)) if (EREG_ERR == 1) { Rb <- UD; C0:C1 <- 1:0; } else { Rb <- tmp; C0:C1 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号。读入其外部寄存器的值至Rb。 读入的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C0:C1,Rb,(Ra,D5)	32bit	C0,C1	—
备注			
<p>D5为无符号的值。 Ra+uext(D5) 的内容是寄存器编号。</p>			

图188

rd C0:C1,Rb,(Ra)

操作	<pre> => rd C0:C1,Rb,(Ra,0) if (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra) if (EREG_ERR == 1) { Rb <- UD; C0:C1 <- 1:0; } else { Rb <- tmp; C0:C1 <- 0:1 } } </pre> <p>Ra作为外部寄存器编号，其外部寄存器的值读入Rb。 读入的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C0:C1,Rb,(Ra)	相当于32bit	C0,C1	—
备注			

图189

rd C0:C1,Rb2,(Ra2)

操作	<pre> if (EREG(Ra2) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra2) if (EREG_ERR == 1) { Rb2 <- UD; C0:C1 <- 1:0; } else { Rb2 <- tmp; C0:C1 <- 0:1; } } </pre> <p>Ra2作为外部寄存器编号，其外部寄存器的值读入Rb2。</p> <p>读入的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C0:C1,Rb2,(Ra2)	16bit	C0,C1	—
备注			
<p>Ra2的内容是寄存器编号。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图190

rd C2:C3,Rb,(Ra,D5)

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra+uext(D5)) if (EREG_ERR == 1) { Rb <- UD; C2:C3 <- 1:0; } else { Rb <- tmp; C2:C3 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号。读入其外部寄存器的值至Rb。</p> <p>读入的成功·失败输入至C2和C3。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C2:C3,Rb,(Ra,D5)	32bit	C2,C3	—
备注			
<p>D5为无符号的值。</p> <p>Ra+uext(D5) 的内容是寄存器编号。</p>			

图191

rd C2:C3,Rb,(Ra)

操作	<pre> => rd C2:C3,Rb,(Ra,0) If (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra) If (EREG_ERR == 1) { Rb <- UD; C2:C3 <- 1:0; } else { Rb <- tmp; C2:C3 <- 0:1; } } </pre> <p>Ra作为外部寄存器编号，其外部寄存器的值读入Rb。 读入的成功·失败输入至C2和C3。</p>		
汇编·助记	格式	影响标记	受影响的标记
rd C2:C3,Rb,(Ra)	相当于32bit	C2,C3	—
备注			
※ Ra的内容是寄存器编号。			

图192

rde C0:C1,Rb,(Ra,D5)

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra+uext(D5)) if (EREG_ERR == 1) { Rb <- UD; C0:C1 <- 1:0; 扩展寄存器错误例外 } else { Rb <- tmp; C0:C1 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号。读入其外部寄存器的值至Rb。</p> <p>读入的成功·失败输入至C0和C1。失败时发生扩展寄存器错误例外。</p>		
汇编·助记	格式	影响标记	受影响的标记
rde C0:C1,Rb,(Ra,D5)	32bit	C0,C1	—
备注			
<p>D5为无符号的值。</p> <p>Ra+uext(D5) 的内容是寄存器编号。</p>			

图193

rde C0:C1,Rb,(Ra)

操作	<pre> => rde C0:C1,Rb,(Ra,0) If (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra); If (EREG_ERR == 1) { Rb <- UD; C0:C1 <- 1:0; 扩展寄存器错误例外 } else { Rb <- tmp; C0:C1 <- 0:1; } } Ra作为外部寄存器编号，其外部寄存器的值读入Rb。 读入的成功·失败输入至C2和C3。失败时发生扩展寄存器 错误例外。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
rde C0:C1,Rb,(Ra)	相当于32bit	C0,C1	-
备注			
※ Ra的内容是寄存器编号。			

图194

rde C2:C3,Rb,(Ra,D5)

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra+uext(D5)) if (EREG_ERR == 1) { Rb <- UD; C2:C3 <- 1:0; 扩展寄存器错误例外 } else { Rb <- tmp; C2:C3 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号。读入其外部寄存器的值至Rb。</p> <p>读入的成功·失败输入至C2和C3。失败时发生扩展寄存器错误例外。</p>		
汇编·助记	格式	影响标记	受影响的标记
rde C2:C3,Rb,(Ra,D5)	32bit	C2,C3	—
备注			
<p>D5为无符号的值。 Ra+uext(D5) 的内容是寄存器编号。</p>			

图195

rde C2:C3,Rb,(Ra)

操作	<pre> => rde C2:C3,Rb,(Ra,0) if (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { tmp <- EREG(Ra); if (EREG_ERR == 1) { Rb <- UD; C2:C3 <- 1:0; 扩展寄存器错误例外 } else { Rb <- tmp; C2:C3 <- 0:1; } } Ra作为外部寄存器编号，其外部寄存器的值读入Rb。 读入的成功·失败输入至C2和C3。失败时发生扩展寄存器 错误例外。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
rde C2:C3,Rb,(Ra)	相当于32bit	C2,C3	—
备注			
※ Ra的内容是寄存器编号。			

图196

wt C0:C1,(D11),Rb

操作	<pre> if (EREG(D11) 未安装) { 扩展寄存器存取例外 } else { EREG(D11) <- Rb; if (EREG_ERR == 1){ C0:C1 <- 1:0; } else { C0:C1 <- 0:1; } } </pre> <p>当前值(D11)作为外部寄存器编号，写出Rb的值至其外部寄存器。 写出的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C0:C1,(D11),Rb	32bit	C0,C1	—
备注			
<p>D11为无符号的值。</p> <p>错误时的扩展寄存器的更新的一方依赖于扩展寄存器的安装。</p>			

图197

wt C0:C1,(Ra,D5),Rb

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra+uext(D5)) <- Rb; if (EREG_ERR == 1) { C0:C1 <- 1:0; } else { C0:C1 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号， 写出Rb的值至其外部寄存器。 读入的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C0:C1,(Ra,D5),Rb	32bit	C0,C1	—
备注			
<p>D5为无符号的值。 Ra+uext(D5) 的内容是寄存器编号。 错误时的扩展寄存器的更新的一方依赖于扩展寄存器的安装。</p>			

图198

wt C0:C1,(Ra),Rb

操作	<pre> => wt C0:C1,(Ra,0),Rb if (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra) <- Rb; If (EREG_ERR == 1){ C0:C1 <- 1:0; } else { C0:C1 <- 0:1; } } </pre> <p>Ra作为外部寄存器编号。写出Rb的值至其外部寄存器。 写出的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C0:C1,(Ra),Rb	相当于32bit	C0,C1	—
备注			
※ 错误时的扩展寄存器的更新一方依赖于扩展寄存器的安装。			

图199

wt C0:C1,(Ra2),Rb2

操作	<pre> if (EREG(Ra2) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra2) <- Rb2; if (EREG_ERR == 1) { C0:C1 <- 1:0; } else { C0:C1 <- 0:1; } } </pre> <p>Ra2作为外部寄存器编号，写出Rb2的值至其外部寄存器。 写出的成功·失败输入至C0和C1。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C0:C1,(Ra2),Rb2	16bit	C0,C1	—
备注			
<p>Ra2的内容是寄存器编号。 错误时的扩展寄存器的更新一方依赖于扩展寄存器的安装。 可能使用的判定只是C7。(不能进行条件执行)</p>			

图200

wt C2:C3,(Ra,D5),Rb

操作	<pre> if (EREG(Ra+uext(D5)) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra+uext(D5)) <- Rb; if (EREG_ERR == 1) { C2:C3 <- 1:0; } else { C2:C3 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号， 写出Rb的值至其外部寄存器。 读入的成功·失败输入至C2和C3。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C2:C3,(Ra,D5),Rb	32bit	C2,C3	—
备注			
<p>D5为无符号的值。 Ra+uext(D5)的内容是寄存器编号。 错误时的扩展寄存器的更新的一方依赖于扩展寄存器的安装。</p>			

图201

wt C2:C3,(Ra),Rb

操作	<pre> => wt C2:C3,(Ra,0),Rb if (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra) <- Rb; if (EREG_ERR == 1) { C2:C3 <- 1:0; } else { C2:C3 <- 0:1; } } </pre> <p>Ra作为外部寄存器编号。写出Rb的值至其外部寄存器。 写出的成功·失败输入至C2和C3。</p>		
汇编·助记	格式	影响标记	受影响的标记
wt C2:C3,(Ra),Rb	相当于32bit	C2,C3	—
备注			
<p>※ Ra的内容是寄存器编号。</p> <p>※ 错误时的扩展寄存器的更新一方依赖于扩展寄存器的安装。</p>			

图202

wte C0:C1,(Ra,D5),Rb

操作	<pre> if (EREG(Ra+uext(D5)) (未安装) { 扩展寄存器存取例外 } else { EREG(Ra+uext(D5)) <- Rb; if (EREG_ERR == 1) { C0:C1 <- 1:0; 扩展寄存器错误例外 } else { C0:C1 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号， 写出Rb的值至其外部寄存器。 读入的成功·失败输入至C2和C3。失败时发生扩展寄存器 错误例外。</p>		
汇编·助记	格式	影响标记	受影响的标记
wte C0:C1,(Ra,D5),Rb	32bit	C0,C1	—
备注			
<p>D5为无符号的值。 Ra+uext(D5)的内容是寄存器编号。 错误时的扩展寄存器的更新的一方依赖于扩展寄存器的安装。</p>			

图203

wte C0:C1,(Ra),Rb

操作	<pre> => wte C0:C1,(Ra,0),Rb if (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra) <- Rb; if (EREG_ERR == 1) { C0:C1 <- 1:0; 扩展寄存器错误例外 } else { C0:C1 <- 0:1; } } </pre> <p>Ra作为外部寄存器编号。写出Rb的值至其外部寄存器。 写出的成功·失败输入至C0和C1。失败时发生扩展寄存器错误例外。</p>		
	汇编·助记	格式	影响标记
wte C0:C1,(Ra),Rb		C0,C1	—
备注			
<p>※ Ra的内容是寄存器编号。 ※ 错误时的扩展寄存器的更新一方依赖于扩展寄存器的安装。</p>			

图204

wte C2:C3,(Ra,D5),Rb

操作	<pre> if (EREG(Ra+uext(D5)) (未安装) { 扩展寄存器存取例外 } else { EREG(Ra+uext(D5)) <- Rb; if (EREG_ERR == 1){ C2:C3 <- 1:0; 扩展寄存器错误例外 } else { C2:C3 <- 0:1; } } </pre> <p>Ra和置换值(D5)相加后的值作为外部寄存器编号， 写出Rb的值至其外部寄存器。</p>		
汇编·助记	格式	影响标记	受影响的标记
wte C2:C3,(Ra,D5),Rb	32bit	C2,C3	—
备注			
<p>D5为无符号的值。 Ra+uext(D5) 的内容是寄存器编号。 错误时的扩展寄存器的更新的一方依赖于扩展寄存器的安装。</p>			

图205

wte C2:C3,(Ra),Rb

操作	<pre> => wte C2:C3,(Ra,0),Rb If (EREG(Ra) 未安装) { 扩展寄存器存取例外 } else { EREG(Ra) <- Rb; If (EREG_ERR == 1) { C2:C3 <- 1:0; 扩展寄存器错误例外 } else { C2:C3 <- 0:1; } } </pre> <p>Ra作为外部寄存器编号。写出Rb的值至其外部寄存器。 写出的成功·失败输入至C2和C3。失败时发生扩展寄存器错误例外。</p>		
汇编·助记	格式	影响标记	受影响的标记
wte C2:C3,(Ra),Rb		C2,C3	—
备注			
<p>※ Ra的内容是寄存器编号。 ※ 错误时的扩展寄存器的更新一方依赖于扩展寄存器的安装。</p>			

图206

图207

br D20

操作	$PC \leftarrow PC + (\text{sext}(D20[19:1]) \ll 1);$ 现在的PC与置换值(D20)相加, 分支于该地址。		
汇编·助记	格式	影响标记	受影响的标记
br D20	32bit	—	—
备注			
D20为有符号的值, 下位1比特变为0。			

图208

br D9

操作	$PC \leftarrow PC + (\text{sext}(D9[8:1]) \ll 1);$ 现在的PC与置换值(D9)相加, 分支于该地址。		
汇编·助记	格式	影响标记	受影响的标记
br D9	16bit	—	—
备注			
D9为有符号的值, 下位1比特变为0。 可能使用的判定是C6和C7。			

brl D20

操作	$PC \leftarrow PC + (\text{sext}(D20[19:1]) \ll 1);$ $LR \leftarrow \text{next_PC};$ $SVR.CF \leftarrow CFR.CF;$		
	<p>子程序呼叫用分支命令。</p> <p>现在的PC与置换值(D20)相加，分支于该地址。</p> <p>返回地址(下一命令的PC)存储于LR、将CFR·CF保存于SVR、CF。</p>		
汇编·助记	格式	影响标记	受影响的标记
brl D20	32bit	SVR.CF	CFR.CF
备注			
D20为有符号的值，下位1比特变为0。			

图209

call D20

操作	=> brr D20		
	<pre>PC <- PC + (sext(D20[19:1]) << 1); LR <- next_PC; SVR.CF <- CFR.CF;</pre> <p>子程序呼叫用分支命令。</p> <p>现在的PC与置换值(D20)相加, 分支于该地址。</p> <p>返回地址(下一命令的PC)存储于LR、将CFR·CF保存于SVR·CF。</p>		
汇编·助记	格式	影响标记	受影响的标记
call D20		SVR.CF	CFR.CF
备注			
D20为有符号的值, 下位1比特变为0。			

图210

brl D9

操作	$PC \leftarrow PC + (\text{sext}(D9[8:1]) \ll 1);$ $LR \leftarrow \text{next_PC};$ $SVR.CF \leftarrow CFR.CF;$		
	<p>子程序呼叫用分支命令。</p> <p>现在的PC与置换值(D9)相加，分支于该地址。</p> <p>返回地址(下一命令的PC)存储于LR、将CFR·CF保存于SVR·CF。</p>		
汇编·助记	格式	影响标记	受影响的标记
brl D9	16bit	SVR.CF	CFR.CF
备注			
<p>D9为有符号的值，下位1比特变为0。</p> <p>可能使用的判定只是C7。(不能进行条件执行)</p>			

图211

图212

call D9

操作	=> brl D9		
	<pre>PC <- PC + (sext(D9[8:1]) << 1); LR <- next_PC; SVR.CF <- CFR.CF;</pre> <p>子程序呼叫用分支命令。</p> <p>现在的PC与置换值(D9)相加, 分支于该地址。</p> <p>返回地址(下一命令的PC)存储于LR、将CFR·CF保存于SVR·CF。</p>		
汇编·助记	格式	影响标记	受影响的标记
call D9		SVR.CF	CFR.CF
备注			

图213

jmp LR

操作	PC <- LR;		
	在LR存储的地址处分支。		
汇编·助记	格式	影响标记	受影响的标记
jmp LR	16bit	—	—
备注			

jmp TAR

操作	PC <- TAR; 在TAR存储的地址处分支。		
汇编·助记	格式	影响标记	受影响的标记
jmp TAR	16bit	—	—
备注			

图214

jmp LR

操作	PC <- LR; LR <- next_PC; SVR.CF <- CFR.CF; 子程序呼叫用分支命令。 在LR存储的地址处分支。返回地址(下一命令的PC) 存储于LR、将CFR·CF保存于SVR·CF。		
汇编·助记	格式	影响标记	受影响的标记
jmp LR	16bit	SVR.CF	CFR.CF
备注			

图215

图216

call LR

操作	=> jmpI LR		
	PC <- LR; LR <- next_PC; SVR.CF <- CFR.CF; 子程序呼叫用分支命令。 在LR存储的地址处分支。返回地址(下一命令的PC) 存储于LR、将CFR·CF保存于SVR·CF。		
汇编·助记	格式	影响标记	受影响的标记
call LR	相当于16bit	SVR.CF	CFR.CF
备注			

图217

jmpI TAR

操作	PC <- TAR; LR <- next_PC; SVR.CF <- CFR.CF; 子程序呼叫用分支命令。 在TAR存储的地址处分支。返回地址(下一命令的PC) 存储于LR、将CFR·CF保存于SVR·CF。		
	汇编·助记	格式	影响标记
jmpI TAR	16bit	SVR.CF	CFR.CF
备注			

图218

call TAR

操作	=> jmp_l TAR		
	PC <- TAR; LR <- next_PC; SVR.CF <- CFR.CF; 子程序呼叫用分支命令。 在TAR存储的地址处分支，返回地址(下一命令的PC)。 存储于LR、将CFR·CF保存于SVR·CF。		
汇编·助记	格式	影响标记	受影响的标记
call TAR		SVR.CF	CFR.CF
备注			

图219

jmp_r LR

操作	PC <- LR; CFR.CF <- SVR.CF; 子程序呼叫用分支命令。 在LR存储的地址处分支。SVR、CF转送至CFR、CF。		
	汇编·助记	格式	影响标记
jmp_r LR	16bit	CFR.CF	SVR.CF
备注			

图220

ret

操作	=> jmp_r LR		
	PC <- LR; CFR.CF <- SVR.CF; 子程序呼叫用分支命令。 在LR存储的地址处分支。SVR、CF转送至CFR、CF。		
汇编·助记	格式	影响标记	受影响的标记
jmp_r LR	相当于16bit	CFR.CF	SVR.CF
备注			

图221

jmp_f LR

操作	PC <- LR; 子程序呼叫用分支命令。 在LR存储的地址处分支。分支用命令缓冲器不充充分支地址的命令时，充充分支地址命令。		
	汇编·助记	格式	影响标记
jmp_f LR	16bit	—	—
备注			

jmpf C6,C2:C4,TAR

操作	PC ← TAR; C2 ← C3; C3 ← C4; C6 ← C4; C3转送至C2、C4转送至C3和C4后，在TAR存储的地址处分支。 分支用命令缓冲器不填充分支地址的命令时，填充分支地址命令。		
汇编·助记	格式	影响标记	受影响的标记
jmpf C6,C2:C4,TAR	16bit	C2,C6,C3	C4,C3
备注			

图222**jmpf Cm,TAR**

	PC ← TAR; Cm ← 1; Cm写入1在TAR存储的地址处分支。 分支用命令缓冲器不填充分支地址的命令时，填充分支地址命令。		
汇编·助记	格式	影响标记	受影响的标记
jmpf Cm,TAR	16bit	Cm	—
备注			

图223

jmpf TAR

操作	=> jmpf C7,TAR PC <- TAR; 在TAR存储的地址分支。		
汇编·助记	格式	影响标记	受影响的标记
jmpf TAR	相当于16bit	—	—
备注			

图224

jloop C6,TAR,Ra,I8

操作	=> jloop C6,C7,TAR,Ra,I8 PC <- TAR; C6 <- (Ra >= 0)? 1:0; Ra <- Ra + sext(I8); 回路使用。进行以下处理。 (1) Ra与当前值(I8)相加，存储于Ra。如果Ra比0小时，C6处设为0。 (2) 在TAR所示的地址处转移。		
编程·助记	格式	影响标记	受影响的标记
jloop C6,TAR,Ra,I8	相当于32bit	C6	—
备注			
I8的有符号的值。 ※ I8处设定0以上的数时有必要注意。			

图225

jloop C6,TAR,Ra

操作	=> jloop C6,C7,TAR,Ra,-1		
	PC <- TAR; C6 <- (Ra >= 0)? 1:0; Ra <- Ra - sext(1); 在循环中使用。进行以下处理。 (1) Ra与-1相加，存储于Ra。如果Ra比0小时，C6处设为0。 (2) 在TAR所示的地址处分支。在分支用命令缓冲器上未填充分支地址的命令时，填充分支地址命令。		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,TAR,Ra		C6	—
备注			

图226

jloop C6,TAR,Ra2

操作	<p>=> jloop C6,C7,TAR,Ra2</p> <p>PC <- TAR; C6 <- (Ra2 >= 0)? 1 : 0;</p> <p>Ra2 <- Ra2 -1;</p> <p>在循环中使用。进行以下处理。</p> <p>(1) Ra2与-1相加，存储于Ra2。如果Ra2比0小时，C6处设为0。</p> <p>(2) 在TAR所示的地址处转移。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,TAR,Ra2		C6	—
备注			
可能使用的判定只是C6。(一般进行条件执行)			

图227

jloop C6,TAR,Ra2,-1

操作	<pre> => jloop C6,C7,TAR,Ra2 PC <- TAR; C6 <- (Ra2 >= 0)? 1 : 0; Ra2 <- Ra2 -1; </pre> <p>在循环中使用。进行以下处理。</p> <p>(1) Ra2与-1相加，存储于Ra2。如果Ra2比0小时，C6处设为0。</p> <p>(2) 在TAR所示的地址处转移。</p>		
	汇编·助记	格式	影响标记
jloop C6,TAR,Ra2,-1	相当于16bit	C6	—
备注			
可能使用的判定只是C6。(一般进行条件执行)			

图228

jloop C6,Cm,TAR,Ra,I8

操作	PC <- TAR; C6 <- (Ra >= 0)? 1:0; Cm <- 1; Ra <- Ra + sext(I8);		
	<p>在循环中使用。进行以下处理。</p> <p>(1) cm处设为0。</p> <p>(2) Ra与当前值(I8)相加，存储于Ra。如果Ra比0小时，C6处设为0。</p> <p>(3) 在TAR所示的地址处。在分支用命令缓冲器上未填充分支地址的命令时，填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,Cm,TAR,Ra,I8	32bit	Cm,C6	—
备注			
<p>I8为有符号的值。</p> <p>I8处设定了0以上的数时应注意。</p> <p>Cm=C6时的操作是不定的。</p>			

图229

jloop C6,Cm,TAR,Ra

操作	\Rightarrow jloop C6,Cm,TAR,Ra,-1 PC <- TAR; C6 <- (Ra >= 0)? 1:0; Cm <- 1; Ra <- Ra - sext(1);		
	<p>在循环中使用。进行以下处理。</p> <p>(1) cm处设为1。</p> <p>(2) Ra与1相加，存储于Ra。如果Ra比0小时，C6处设为0。</p> <p>(3) 在TAR所示的地址处分支。在分支用命令缓冲器上未充分分支地址的命令时，填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,Cm,TAR,Ra	相当于32bit	Cm,C6	—
备注			
※ Cm=C6时的操作是不定的。			

图230

jloop C6,Cm,TAR,Ra2

操作	PC <- TAR; C6 <- (Ra2 >= 0)? 1 : 0; Cm <- 1; Ra2 <- Ra2 - 1;		
	<p>在循环中使用。进行以下处理。</p> <p>(1) cm处设为1。</p> <p>(2) Ra2与1相加，存储于Ra2。如果Ra2比0小时，C6处设为0。</p> <p>(3) 在TAR所示的地址处分支。在分支用命令缓冲器上未充分分支地址的命令时，充充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,Cm,TAR,Ra2	16bit	Cm,C6	—
备注			
<p>Cm=C6时的操作是不定的。</p> <p>可能使用的判定只是C6。(一般进行条件执行)</p>			

图231

jloop C6,Cm,TAR,Ra2,-1

操作	<p>=> jloop C6,Cm,TAR,Ra2</p> <p>PC <- TAR; C6 <- (Ra2 >= 0)? 1 : 0; Cm <- 1; Ra2 <- Ra2 -1;</p> <p>在循环中使用。进行以下处理。</p> <p>(1) cm处设为1。</p> <p>(2) Ra2与-1相加，存储于Ra2。如果Ra2比0小时，C6处设为0。</p> <p>(3) 在TAR所示的地址处转移。</p>		
	汇编·助记	格式	影响标记
jloop C6,Cm,TAR,Ra2,-1	16bit	Cm,C6	—
备注			
<p>※ Cm=C6时的操作是不定的。</p> <p>可能使用的判定只是C6。(一般进行条件执行)</p>			

图232

jloop C6,C2:C4,TAR,Ra,I8

操作	PC <- TAR; C2 <- C3, C3 <- C4, C6 <- C4; C4 <- (Ra >= 0)? 1 : 0; Ra2 <- Ra + sext(I8);		
	<p>在循环中使用。进行以下处理。</p> <p>(1) C3转送至C2, C4转送至C3和C6。</p> <p>(2) Ra2与当前值(I8)相加, 存储于Ra。如果Ra比0小时, C4处设为0。</p> <p>(3) 在TAR所示的地址处分支。在分支用命令缓冲器上未充分分支地址的命令时, 填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,C2:C4,TAR,Ra,I8	32bit	C2,C4,C6,C3	--
备注			
<p>I8为有符号的值。</p> <p>I8处设定了0以上的数时应注意。</p>			

图233

jloop C6,C2:C4,TAR,Ra

操作	<p>=> jloop C6,C2:C4,TAR,Ra,-1</p> <p>PC <- TAR; C2 <- C3, C3 <- C4, C6 <- C4; C4 <- (Ra >= 0)? 1 : 0; Ra <- Ra - sext(1);</p> <p>在循环中使用。进行以下处理。</p> <p>(1) C3转送至C2, C4转送至C3和C6。 (2) Ra与-1相加, 存储于Ra。如果Ra比0小时, C4处设为0。 (3) 在TAR所示的地址处分支。在分支用命令缓冲器上未填充分支地址的命令时, 填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,C2:C4,TAR,Ra	相当于32bit	C2,C4,C6,C3	—
备注			

图 234

jloop C6,C2:C4,TAR,Ra2

操作	PC <- TAR; C2 <- C3; C3 <- C4; C6 <- C4; C4 <- (Ra2 >= 0)? 1 : 0; Ra2 <- Ra2 -1;		
	<p>在循环中使用。进行以下处理。</p> <p>(1) C3转送至C2, C4转送至C3和C6。</p> <p>(2) Ra2与-1相加, 存储于Ra2。如果Ra2比0小时, C4处设为0。</p> <p>(3) 在TAR所示的地址处分支。在分支用命令缓冲器上未填充分支地址的命令时, 填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,C2:C4,TAR,Ra2	16bit	C2,C4,C6,C3	C4,C3
备注			
可能使用的判定只是C6。(一般进行条件执行)			

图235

jloop C6,C2:C4,TAR,Ra2,-1

操作	=> jloop C6,C2:C4,TAR,Ra2		
	<p>PC <- TAR; C2 <- C3; C3 <- C4; C6 <- C4; C4 <- (Ra2 >= 0)? 1 : 0; Ra2 <- Ra2 -1;</p> <p>在循环中使用。进行以下处理。</p> <p>(1) C3转送至C2, C4转送至C3和C6。</p> <p>(2) Ra2与-1相加, 存储于Ra2。如果Ra2比0小时, C4处设为0。</p> <p>(3) 在TAR所示的地址处分支。在分支用命令缓冲器上未充分分支地址的命令时, 填充分支地址命令。</p>		
汇编·助记	格式	影响标记	受影响的标记
jloop C6,C2:C4,TAR,Ra2,-1	相当于16bit	C2,C4,C6,C3	C4,C3
备注			
可能使用的判定只是C6。(一般进行条件执行)			

图236

图237

jloop C5,LR,Ra,I8

<pre>PC <- LR; C5 <- (Ra >= 0)? 1:0; Ra <- Ra + sext(I8);</pre> <p>在循环中使用。进行以下处理。 (1)Ra与当前值(I8)相加,存储于Ra。如Ra小于0时,C5处设为0。 (2)在LR所示的地址处分支。在分支用命令缓冲器上未填充分支地址的命令时,填充分支地址的命令</p>			
汇程·助记	格式	影响标记	受影响的标记
jloop C5,LR,Ra,I8	32bit	C5	-
备注			
I8为有符号的值。 I8处设定了0以上的数时应注意。			

图238

jloop C5,LR,Ra

<pre>⇒ jloop C5,LR,Ra,-1</pre> <pre>PC ← LR;</pre> <pre>C5 ← (Ra ≥ 0)? 1:0;</pre> <pre>Ra ← Ra - sext(1);</pre> <p>在循环中使用。进行以下处理。 (1) Ra与-1相加，存储于Ra。如Ra小于0时，C5处设为0。 (2) 在LR所示的地址处分支。在分支用命令缓冲器上未填充分支地址的命令时，填充分支地址的命令</p>			
汇编·助记	格式	影响标记	受影响的标记
jloop C5,LR,Ra	相当于32bit	C5	-
备注			
18为有符号的值。 18处设定了0以上的数时应注意。			

图239

settar D9

<pre>TAR ← PC + (sext(D9[8:1]) << 1);</pre> <p>进行以下处理。 (1) PC和置换值（D9）相加后的地址存储于TAR。 (2) 读取上述地址命令，存储于分支用命令缓冲器。</p>			
汇编·助记	格式	影响标记	受影响的标记
settar D9	32bit	-	-
备注			
D9为有符号的值，下位1比特变为0。			

图 240

settar C6,Cm,D9

操作	<p>TAR <- PC + (sext(D9[8:1]) <<1); C6 <- 1; Cm <- 0;</p> <p>进行以下处理。 (1)PC和置换值 (D9) 相加后的地址存储于TAR。 (2) 读取上述地址命令, 存储于分支用命令缓冲器。 (3) C6设为1, Cm设为0。</p>		
汇编·助记	格式	影响标记	受影响的标记
settar C6,Cm,D9	32bit	Cm,C6	-
备注			
<p>D9为有符号的值, 下位1比特变为0。 Cm=C6时的操作是不定的。</p>			

图241

settar C6,D9

操作	=> settar C6,C7,D9		
	TAR <- PC + (sext(D9[8:1]) <<1); C6 <- 1; 进行以下处理。 (1)PC和置换值（D9）相加后的地址存储于TAR。 (2)读取上述地址命令，存储于分支用命令缓冲器。 (3) C6设为1。		
汇编·助记	格式	影响标记	受影响的标记
settar C6,D9	相当于32bit	C6	—
备注			
D9为有符号的值，下位1比特变为0。			

图 242

settar C6,C2:C4,D9

操作	$\text{TAR} \leftarrow \text{PC} + (\text{sext}\{\text{D9}[8:1]\} \ll 1);$ $\text{C2} \leftarrow 0; \text{C3} \leftarrow 0; \text{C4} \leftarrow 1, \text{C6} \leftarrow 1;$ <p>进行以下处理。</p> <p>(1) PC和置换值 (D9) 相加后的地址存储于TAR。</p> <p>(2) 读取上述地址命令, 存储于分支用命令缓冲器。</p> <p>(3) C4和C6设为1, C2和C3设为0。</p>			
	汇编·助记	格式	影响标记	受影响的标记
	settar C6,C2:C4,D9	32bit	C2,C4,C6,C3	-
备注				
D9为有符号的值, 下位1比特变为0。				

图243

settar C6,C4,D9

操作	$TAR \leftarrow PC + (\text{sext}(D9[8:1]) \ll 1);$ $C4 \leftarrow 0; C6 \leftarrow 1;$ 进行以下处理。 (1)PC和置换值(D9)相加后的地址存储于TAR。 (2)读取上述地址命令,存储于分支用命令缓冲器。 (3)C6设为1,C4设为0。		
	汇编·助记	格式	影响标记
	settar C6,C4,D9	16bit	C4,C6
备注			
可能使用的判定只是C7(不能进行条件执行) D9为有符号的值,下位1比特变为0。			

图244

setlr D9

操作	$LR \leftarrow PC + (\text{sext}(D9[8:1]) \ll 1);$ 进行以下处理。 (1)PC和置换值(D9)相加后的地址存储于TAR。 (2)读取上述地址命令,存储于分支用命令缓冲器。		
	汇编·助记	格式	影响标记
	setlr D9	32bit	-
备注			
D9为有符号的值,下位1比特变为0。			

图245

setlr C5,D9

操作	$LR \leftarrow PC + (\text{sext}(D9[8:1]) \ll 1);$ $C5 \leftarrow 1;$ 进行以下处理。 (1) PC和置换值 (D9) 相加后的地址存储于TLR。 (2) 读取上述地址命令, 存储于分支用命令缓冲器。 (3) C5设为1。		
汇编·助记	格式	影响标记	受影响的标记
setlr C5,D9	32bit	C5	-
备注			
D9为有符号的值, 下位1比特变为0。			

图246

setbb TAR

操作	从TAR所示的地址读取的命令填充于分支用命令缓冲器		
汇编·助记	格式	影响标记	受影响的标记
setbb TAR	16bit	-	-
备注			

图247

setbb LR

操作	从LR所示的地址读取的命令填充于分支用命令缓冲器		
汇编·助记	格式	影响标记	受影响的标记
setbb LR	16bit	—	—
备注			

图248

intd

操作	PSR.IE ← 0; 中断禁止状态。		
汇编·助记	格式	影响标记	受影响的标记
intd	32bit	PSR.IE	—
备注			
可通过特权标准0, 2执行。通过特权标准执行时, 发生特权命令执行例外。			

图249

inte

操作	PSR.IE <- 1; 中断许可状态。		
汇编·助记	格式	影响标记	受影响的标记
inte	32bit	PSR.IE	-
备注			
可通过特权标准0, 2执行。通过特权标准执行时, 发生特权命令执行例外。			

图250

vmpswd

操作	PSR.SWE <- 0; 中断禁止状态。		
汇编·助记	格式	影响标记	受影响的标记
vmpswd	32bit	PSR.SWE	-
备注			
可通过特权标准0, 2执行。通过特权标准执行时, 发生特权命令执行例外。			

图251

vmpswe

操作	PSR.SWE <- 1; 中断禁止状态。		
汇编·助记	格式	影响标记	受影响的标记
vmpswe	32bit	PSR.SWE	-
备注			
可通过特权标准0, 2执行。通过特权标准执行时, 发生特权命令执行例外。			

图252

vmpsleap

操作	PSR.SWE <- 1; LP sleep; VMP switch; 停止命令的发布, 从running移动ready切换LP。		
汇编·助记	格式	影响标记	受影响的标记
vmpsleap	32bit	PSR.SWE	-
备注			
直到可能切换VMP后停止(阻塞)命令发布。			

图 253
vmpwait

操作	PSR.SWE <- 1; LP wait; VMP switch; 停止命令的发布，从running移动waiting切换LP。		
汇编·助记	格式	影响标记	受影响的标记
vmpwait	32bit	PSR.SWE	-
备注			
直到可能切换VMP后停止（阻塞）命令发布。			

图 254
vmpsus

操作	PSR.SWE <- 1; LP suspend; VMP switch; 停止命令的发布，从running移动suspended waiting切换LP。		
汇编·助记	格式	影响标记	受影响的标记
vmpsus	32bit	PSR.SWE	-
备注			
直到可能切换VMP后停止（阻塞）命令发布。			

图255

rti

操作	<pre> if (PSR.EH == 0) { PC <- IPC; PSR <- IPSR; } else { PC <- EPC; PSR <- EPSR; } </pre> <p>从中断处理、例外处理的恢复时使用。 IPS或EPS向PSR转换、存储于IPC或EPC的地址 处分支。</p>		
汇编·助记	格式	影响标记	受影响的标记
rti	16bit	PSR	PSR.EH
备注			
<p>可通过特权标准0.2执行。通过特权标准执行时，发生 特权命令执行例外。</p>			

图 256

piNI(pi0I,pi1I,pi2I,pi3I,pi4I,pi5I,pi6I,pi7I)

操作	<pre> N(±0~7(pi0I,pi1I,...,pi7I) step1: ISR.PIEXE <- 1; if (IDMR1.piN == 1) { 冻结 插入移动次序终止 } IPSR <- PSR; IPC <- PC; if (LOIPLR1.piN == 0) { LOIVBR.IC <- 0x30+N; new_PC <- LOIVBR; new_PL <- 0; } else { IVBR.IC <- 0x30+N; new_PC <- IVBR; new_PL <- 2; } step2: PSR.IH <- 1; PSR.IE <- 0; PSR.LPIE[3:0] <- 0b0000; if (PSR.PL[1:0] >(u) new_PL[1:0]) { PSR.PL <- new_PL; } PC <- new_PC; 执行程序中断 </pre>		
	汇编·助记	格式	影响标记
piNI (pi0I,pi1I,pi2I,pi3I, pi4I,pi6I,pi6I,pi7I)	32bit	PSR.IH,PSR.IE, PSR.PL,PSR.LPIE	PSR
备注			
PSR.IH,PSR.IE,PSR.PL,PSR.LPIE以外的PSR的各位及CFR 无变化。			

图 257

piN(pi0,pi1,pi2,pi3,pi4,pi5,pi6,pi7)

操作	<pre> Ni≠0~7(pi0,pi1,...,pi7) step1: ISR.PIEXE <- 1; if (IDMR1.piN == 1) { 冻结 插入移动次序终止 } IPSR <- PSR; IPC <- PC; if (LOIPLR1.piN == 0) { LOIVBR.IC <- 0x30+N; new_PC <- LOIVBR; new_PL <- 0; } else { IVBR.IC <- 0x30+N; new_PC <- IVBR; new_PL <- 2; } step2: PSR.IH <- 1; PSR.IE <- 0; PSR.LPIE[3:0] <- 0b0000; if (PSR.PL[1:0] >(u) new_PL[1:0]) { PSR.PL <- new_PL; } PC <- new_PC; </pre> <p>程序中中断中使用 PSR和PC存储于IPSR和IPC，分支于中断程序， 变更特权标准</p>		
	汇编·助记	格式	影响标记
piN(pi0,pi1,pi2,pi3, pi4,pi5,pi6,pi7)	16bit	PSR.IH,PSR.IE, PSR.PL,PSR.LPIE	PSR
备注			
PSR.IH,PSR.IE,PSR.PL,PSR.LPIE以外的PSR的各位及CFR无变化。			

图258

scN(sc0,sc1,sc2,sc3,sc4,sc5,sc6,sc7)

操作	<pre> N:=0~7(sc0,sc1,...,sc7) step1: ISR.SYSCALL <- 1; if (IDMR1.scN == 1) { 冻结 插入移动次序终止 } IPSR <- PSR; IPC <- PC; if (LOIPLR1.scN == 0) { LOIVBR.IC <- 0x08+N; new_PC <- LOIVBR; new_PL <- 0; } else { IVBR.IC <- 0x08+N; new_PC <- IVBR; new_PL <- 2; } step2: PSR.IH <- 1; PSR.IE <- 0; if (PSR.PL[1:0] >(u) new_PL[1:0]) { PSR.PL <- new_PL; } PC <- new_PC; </pre> <p>程序中中断中使用 PSR和PC存储于IPSR和IPC，分支于中断程序， 变更特权标准</p>		
汇编·助记	格式	影响标记	受影响的标记
scN(sc0,sc1,sc2,sc3 (sc4,sc5,sc6,sc7)	16bit	PSR.IH,PSR.IE, PSR.PL	PSR
备注			
PSR.IH,PSR.IE,PSR.PL,PSR.IPIE以外的PSR的各位及CFR无变化。			

图259

add Rc,Ra,Rb

操作	$Rc \leftarrow Ra + Rb;$ Ra与Rb相加。		
汇编·助记	格式	影响标记	受影响的标记
$Rc \leftarrow Ra + Rb;$	32bit	—	—
备注			

图260

add Rc3,Ra3,Rb3

操作	$Rc3 \leftarrow Ra3 + Rb3;$ Ra3与Rb3相加，存储于Rc3。		
汇编·助记	格式	影响标记	受影响的标记
add Rc3,Ra3,Rb3	16bit	—	—
备注			
可能使用的判定是C7。（不能进行条件执行）			

图261

add Ra2,Rb2

操作	$Ra2 \leftarrow Rb2 + Ra2;$ Ra2与Rb2相加, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
add Ra2,Rb2	16bit	-	-
备注			
可能使用的判定是C7。(不能进行条件执行)			

图262

add Rb,Ra,I12

操作	$Rb \leftarrow Ra + sext(I12);$ Ra与当前值(I12)相加。		
汇编·助记	格式	影响标记	受影响的标记
add Rb,Ra,I12	32bit	-	-
备注			
I12为有符号的值。			

图263

add Ra2,I5

操作	$Ra2 \leftarrow Ra2 + sext(I5);$ Ra2与当前值 (I5) 相加, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
add Ra2,I5	16bit	-	-
备注			
I5为有符号的值。 可能使用的判定只是C7。(不能进行条件执行)			

图264

add SP,I19

操作	$SP \leftarrow SP + (sext(I19[18:3]) \ll 3);$ SP与当前值 (I19) 相加, 存储于SP。		
汇编·助记	格式	影响标记	受影响的标记
add SP,I19	32bit	-	-
备注			
I19为有符号的值, 下位3比特变为0。			

图265

add SP,I11

操作	$SP \leftarrow SP + (\text{sext}(I11[10:3]) \ll 3);$ SP与当前值 (I11) 相加, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
add SP,I11	16bit	—	—
备注			
I11为有符号的值, 下位3比特变为0。 可能使用的判定只是C7。(不能进行条件执行)			

图266

addu Rb,GP,I13

操作	$Rb \leftarrow GP + \text{uext}(I13);$ GP与当前值 (I13) 相加, 存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
addu Rb,GP,I13	32bit	—	—
备注			
I13为无符号的值。			

图267

addu Rb,SP,I13

操作	$Rb \leftarrow SP + \text{uext}(I13);$ SP与当前值 (I13) 相加, 存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
addu Rb,SP,I13	32bit	-	-
备注			
I13为有符号的值。			

图268

addu Ra3,SP,I6

操作	$Ra3 \leftarrow SP + (\text{uext}(I6[5:1]) \ll 1);$ SP与当前值 (I6) 相加, 存储于Ra3。		
汇编·助记	格式	影响标记	受影响的标记
addu Ra3,SP,I6	16bit	-	-
备注			
I6为无符号的值, 下位1比特变为0。 可能使用的判定只是C7。(不能进行条件执行)			

addvw Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(Ra[31:0]) + sext33(Rb[31:0]); Rc <- tmp[31:0]; Rc <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); Rc <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); </pre> <p>Ra和Rb相加。结果如不用有符号的32比特值表示，则饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
addvw Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图269

addvh Rc,Ra,Rb

操作	<pre>tmp[16:0] <- sext17(Ra[15:0]) + sext17(Rb[15:0]); Rc[31:0] <- Ra[31:0] + Rb[31:0]; Rc <- 0x0000_7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc <- 0xffff_8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]);</pre> <p>Ra和Rb相加。结果如不用有符号的16比特值表示，则饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
addvh Rc,Ra,Rb	32bit	-	-
备注			

图270

addc Rc,Ra,Rb

操作	<pre>tmp[32:0] <- Ra[31:0] + Rb[31:0] + C0; Rc <- tmp[31:0]; C0:C1 <- 0:1; C0:C1 <- 1:0, CFR.CAS <- 1, if (tmp[32] == 1);</pre> <p>Ra、Rb和Co相加。相加后的进位存储于C0、C1及CFR.CAS。</p>		
汇编·助记	格式	影响标记	受影响的标记
addc Rc,Ra,Rb	32bit	C0,C1,CFR.CAS	C0
备注			

图271

adds Rc,Ra,Rb

操作	$Rc \leftarrow (\text{sext33}(Ra) + \text{sext33}(Rb))[32:1];$ Ra与Rb相加，其结果右移位1比特算术。		
汇编·助记	格式	影响标记	受影响的标记
adds Rc,Ra,Rb	32bit	—	—
备注			

图272

addsr Rc,Ra,Rb

操作	$Rc \leftarrow (\text{sext33}(Ra) + \text{sext33}(Rb) + 0x0_0000_0001[32:0])[32:1];$ Ra和Rb相加，其结果右移位1比特算术。		
汇编·助记	格式	影响标记	受影响的标记
addsr Rc,Ra,Rb	32bit	—	—
备注			

图273

s1add Rc,Ra,Rb

操作	$Rc \leftarrow Ra + (Rb \ll 1);$ Rb 1比特算术左移位后的值与Ra相加。		
汇编·助记	格式	影响标记	受影响的标记
s1add Rc,Ra,Rb	32bit	-	-
备注			

图274

s1add Rc3,Ra3,Rb3

操作	$Rc3 \leftarrow Ra3 + (Rb3 \ll 1);$ Ra3和Rb3 1比特算术右移位相加，存储于Rc3。		
汇编·助记	格式	影响标记	受影响的标记
s1add Rc3,Ra3,Rb3	16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图275

s2add Rc,Ra,Rb

操作	$Rc \leftarrow Ra + (Rb \ll 2);$ Rb左移位2比特算术的值与Ra相加。		
汇编·助记	格式	影响标记	受影响的标记
s2add Rc,Ra,Rb	32bit	-	-
备注			

图276

s2add Rc3,Ra3,Rb3

操作	$Rc3 \leftarrow Ra3 + (Rb3 \ll 2);$ Ra3和Rb3右移位2比特算术相加，存储于Rc3。		
汇编·助记	格式	影响标记	受影响的标记
s2add Rc3,Ra3,Rb3	16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图277

addmsk Rc,Ra,Rb

操作	<pre> if (CFR.BPO <(u) 31) { Rc[31:CFR.BPO+1] <- Ra[31:CFR.BPO+1]; Rc[CFR.BPO:0] <- Ra[CFR.BPO:0] + Rb[CFR.BPO:0]; } else { Rc[31:0] <- Ra[31:0] + Rb[31:0]; } </pre> <p>在模数寻址中使用。 只在CFR.BPO指定范围内Ra与Rb相加。 没有指定的范围转送Ra的值。</p>		
汇编·助记	格式	影响标记	受影响的标记
addmsk Rc,Ra,Rb	32bit	—	CFR.BPO
备注			

图278

addarvw Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(Ra) + sext33(Rb) + uext33(~(Ra[31:31])); Rc <- tmp[31:0]; Rc <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); Rc <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); </pre> <p>进行图像处理的绝对值舍入。 Ra和Rb相加。Ra为正时，再加1。</p>		
汇编·助记	格式	影响标记	受影响的标记
addarvw Rc,Ra,Rb	32bit	CFR.OVS	—
备注			
<p>进行绝对值舍入时，在Rb中加入比舍入位下位的位在1中包含的值。 ex. Rb <- 0x0000_7fff (MSB 16bit舍入时)</p>			

图279

sub Rc,Rb,Ra

操作	<pre> Rc <- Rb - Ra; </pre> <p>Rb减Ra。</p>		
汇编·助记	格式	影响标记	受影响的标记
sub Rc,Rb,Ra	32bit	—	—
备注			
<p>请注意。操作数顺序。</p>			

图280

sub Rc3,Rb3,Ra3

操作	$Rc3 \leftarrow Rb3 - Ra3;$ Rb3减Ra3, 存储于Rc3。		
汇编·助记	格式	影响标记	受影响的标记
sub Rc3,Rb3,Ra3	16bit	—	—
备注			
<p>请注意操作数顺序。 可能使用的判定只是C7。（不可进行条件执行）</p>			

图281

sub Rb2,Ra2

操作	$Rb2 \leftarrow Rb2 - Ra2;$ Rb2减Ra2, 存储于Rc2。		
汇编·助记	格式	影响标记	受影响的标记
sub Rb2,Ra2	16bit	—	—
备注			
<p>请注意操作数顺序。 可能使用的判定只是C7。（不可进行条件执行）</p>			

图282

sub Rb,Ra,I12

操作	=> add Rb,Ra,-I12 Rb <- Ra - sext(I12); Ra减当前值(I12)。		
汇编·助记	格式	影响标记	受影响的标记
sub Rb,Ra,I12	相当于32bit	-	-
备注			
I12为有符号的值。			

图283

sub Ra2,I5

操作	=> add Ra2,-I5 Ra2 <- Ra2 + sext(I5); Rb2减当前值(I5)，存储于Rc2。		
汇编·助记	格式	影响标记	受影响的标记
sub Ra2,I5	相当于16bit	-	-
备注			
I5为有符号的值。 可能使用的判定只是C7。（不可进行条件执行）			

图284

sub SP,I19

操作	=> add SP,-I19 SP <- SP - (sext(I19[18:3]) << 3); SP减当前值(I19)存储于SP。		
汇编·助记	格式	影响标记	受影响的标记
sub SP,I19	相当于32bit	-	-
备注			
I19为有符号的值，下位3比特变为0。			

图285

sub SP,I11

操作	=> add SP,-I11 SP <- SP + (sext(I11[10:3]) << 3); SP减定值(I11)，存储于SP。		
汇编·助记	格式	影响标记	受影响的标记
sub SP,I11	相当于16bit	-	-
备注			
I11为有符号的值，下位3比特变为0。			
可能使用的判定只是C7。（不可进行条件执行）			

图286

subc Rc,Rb,Ra

操作	<pre>tmp[32:0] <- Rb[31:0] - Ra[31:0] - C0; Rc <- tmp[31:0]; C0:C1 <- 0:1; C0:C1 <- 1:0, CFR.CAS <- 1, if (tmp[32] == 1);</pre> <p>Rb减Ra, 再减C0。减后结果的借位存储于C0, C1, CFR.CAS。</p>		
汇编·助记	格式	影响标记	受影响的标记
subc Rc,Rb,Ra	32bit	C0,C1,CFR.CAS	C0
备注			
请注意操作数顺序。			

图287

subvw Rc,Rb,Ra

操作	<pre>tmp[32:0] <- sext33(Rb[31:0]) - sext33(Ra[31:0]); Rc <- tmp[31:0]; Rc <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); Rc <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]);</pre> <p>Rb减Ra。结果如不用有符号的32比特值表示, 则饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
subvw Rc,Rb,Ra	32bit	CFR.OVS	-
备注			
请注意操作数顺序。			

图288

subvh Rc,Rb,Ra

操作	$\text{tmp}[16:0] \leftarrow \text{sext17}(\text{Rb}[15:0]) - \text{sext17}(\text{Ra}[15:0]);$ $\text{Rc}[31:0] \leftarrow \text{Rb}[31:0] - \text{Ra}[31:0];$ $\text{Rc} \leftarrow 0x0000_7fff, \text{CFR.OVS} \leftarrow 1, \text{if } (\text{tmp}[16:0] > 0x0_7fff[16:0]);$ $\text{Rc} \leftarrow 0xffff_8000, \text{CFR.OVS} \leftarrow 1, \text{if } (\text{tmp}[16:0] < 0x1_8000[16:0]);$ Rb减Ra。结果如不用有符号的16比特值表示，则饱和。		
汇编·助记	格式	影响标记	受影响的标记
subvh Rc,Rb,Ra	32bit	CFR.OVS	—
备注			
请注意操作数顺序。			

图289

subs Rc,Rb,Ra

操作	$\text{Rc} \leftarrow (\text{sext33}(\text{Rb}) - \text{sext33}(\text{Ra}))\llbracket 32:1 \rrbracket;$ Rb减Ra，其结果1比特值算术右移位。		
汇编·助记	格式	影响标记	受影响的标记
subs Rc,Rb,Ra	32bit	—	—
备注			
请注意操作数顺序。			

图290

submsk Rc,Rb,Ra

操作	<pre> if (CFR.BPO <(u) 31) { Rc[31:CFR.BPO+1] <- Rb[31:CFR.BPO+1]; Rc[CFR.BPO:0] <- Rb[CFR.BPO:0] - Ra[CFR.BPO:0]; } else { Rc[31:0] <- Rb[31:0] - Ra[31:0]; } </pre> <p>在模数寻址中使用。 只在CFR.BPO指定的范围内Rb减Ra。没有指定的范围转换Rb的值。</p>			
	汇编·助记	格式	影响标记	受影响的标记
submsk Rc,Rb,Ra		32bit	-	CFR.BPO
备注				
请注意操作数顺序。				

图291

rsub Rb,Ra,I8

操作	<p>Rb <- sext(I8) - Ra;</p> <p>当前值(I12)减Ra。</p>			
	汇编·助记	格式	影响标记	受影响的标记
rsub Rb,Ra,I8		32bit	-	-
备注				
I8为有符号的值。 请注意操作数顺序。				

图292

rsub Ra2,I4

操作	$Ra2 \leftarrow sext(I4) - Ra2;$ 当前值(I4)减Ra2，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
rsub Ra2,I4	16bit	-	-
备注			
I4为有符号的值。 请注意操作数顺序。 可能使用的判定只是C7。（不可进行条件执行）			

图293

rsub Ra2,Rb2

操作	$Ra2 \leftarrow Rb2 - Ra2;$ Rb2减Ra2，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
rsub Ra2,Rb2	16bit	-	-
备注			
请注意操作数顺序。 可能使用的判定只是C7。（不可进行条件执行）			

图294

neg Rb,Ra

操作	\Rightarrow rsub Rb,Ra,0 Rb \leftarrow - Ra; 反转Ra的符号。		
汇编助记	格式	影响标记	受影响的标记
rsub Rb,Ra,0	相当于32bit	-	-
备注			

图295

neg Ra2

操作	\Rightarrow rsub Ra2,0 Ra2 \leftarrow sext(14) - Ra2; 反转Ra2的符号，存储于Ra2。		
汇编助记	格式	影响标记	受影响的标记
neg Ra2	相当于16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图296

negvh Rb,Ra

操作	<pre>tmp[16:0] <- -sxt17(Ra[16:0]); Rb[31:0] <- -Ra[31:0]; Rb <- 0x0000_7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rb <- 0xffff_8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]);</pre> <p>反转Ra符号，结果如不用有符号16比特表示，则饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
negvh Rb,Ra	32bit	CFR.OVS	—
备注			

图297

negvw Rb,Ra

操作	<pre>tmp[32:0] <- -sxt33(Ra[31:0]); Rb <- tmp[31:0], Rb <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); Rb <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]);</pre> <p>反转Ra符号，结果如不用有符号32比特表示，则饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
negvw Rb,Ra	32bit	CFR.OVS	—
备注			

图298

abs Rb,Ra

操作	Rb <- Ra, if (Ra >= 0); Rb <- -Ra, if (Ra < 0); 取Ra的绝对值。		
汇编·助记	格式	影响标记	受影响的标记
abs Rb,Ra	32bit	—	—
备注			

图 299

absvw Rb,Ra

操作	Rb <- Ra, if(Ra >= 0); Rb <- -Ra, if(Ra < 0); Rb <- 0x7fff_ffff, CFR.OVS <- 1, if (Ra == 0x8000_0000); 取Ra的绝对值，结果如不用有符号的 32比特值表示，则饱和。		
汇编·助记	格式	影响标记	受影响的标记
absvw Rb,Ra	32bit	CFR.OVS	—
备注			

图 300

absvh Rb,Ra

操作	Rb <- Ra, if (Ra[15:0] >= 0); Rb <- -Ra, if (Ra[15:0] < 0); Rb <- 0x0000_7fff, CFR.OVS <- 1, if (Ra[15:0] == 0x8000); 取Ra的绝对值，结果如不用有符号的 16比特值表示，则饱和。		
汇编·助记	格式	影响标记	受影响的标记
absvh Rb,Ra	32bit	CFR.OVS	-
备注			

图301

max Rc,Ra,Rb

操作	Rc <- Ra, C0:C1 <- 0:1, if (Ra >= Rb); Rc <- Rb, C0:C1 <- 1:0, if (Ra < Rb); Ra和Rb比较，大的一方存储于Rc。 另外，将比较的结果输入到C0和C1。		
汇编·助记	格式	影响标记	受影响的标记
max Rc,Ra,Rb	32bit	C0,C1	-
备注			

图302

图 303

min Rc,Ra,Rb

操作	Rc <- Ra, C0:C1 <- 0:1, if (Ra < Rb); Rc <- Ra, C0:C1 <- 1:0, if (Ra >= Rb); Ra与Rb比较，大的一方存储于Rc。另外，将比较的结果输入到C0和C1。		
汇编·助记	格式	影响标记	受影响的标记
min Rc,Ra,Rb	32bit	C0,C1	-
备注			

图 304

and Rc,Ra,Rb

操作	Rc <- Ra & Rb; 取Ra和Rb的逻辑积。		
汇编·助记	格式	影响标记	受影响的标记
and Rc,Ra,Rb	32bit	-	-
备注			

图 305

and Ra2,Rb2

操作	and Ra2,Rb2; 取Ra2和Rb2的逻辑积，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
and Ra2,Rb2	16bit	-	-
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图 306

and Rb,Ra,I8

操作	Rb <- Ra & uext(I8); 取Ra和当前值（I8）的逻辑积。		
汇编·助记	格式	影响标记	受影响的标记
and Rb,Ra,I8	32bit	-	-
备注			
I8为无符号的值。			

图307

andn Rc,Ra,Rb

操作	$Rc \leftarrow Ra \& \sim Rb;$ 取Ra和Rb的反转的逻辑积。		
汇编·助记	格式	影响标记	受影响的标记
andn Rc,Ra,Rb	32bit	—	—
备注			

图308

andn Ra2,Rb2

操作	$Ra2 \leftarrow Ra2 \& \sim Rb2;$ 取Ra和Rb2的反转的逻辑积，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
andn Ra2,Rb2	16bit	—	—
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图309

andn Rb,Ra,I8

操作	$Rb \leftarrow Ra \ \& \ \sim uext(I8);$ 取Ra和Rb的反转的逻辑积。		
汇编·助记	格式	影响标记	受影响的标记
andn Rb,Ra,I8	32bit	—	—
备注			

图310

or Rc,Ra,Rb

操作	$Rc \leftarrow Ra \ \ Rb;$ 取Ra和Rb的逻辑积。		
汇编·助记	格式	影响标记	受影响的标记
or Rc,Ra,Rb	32bit	—	—
备注			

图311

or Ra2,Rb2

操作	$Ra2 \leftarrow Ra2 Rb2;$ 取Ra和Rb2的反转的逻辑积，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
or Ra2,Rb2	16bit	—	—
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图312

or Rb,Ra,I8

操作	$Rb \leftarrow Ra uext(I8);$ 取Ra和当前值（I8）的逻辑积		
汇编·助记	格式	影响标记	受影响的标记
or Rb,Ra,I8	32bit	—	—
备注			
I8为符号的值。			

图313

xor Rc,Ra,Rb

操作	$Rc \leftarrow Ra \wedge Rb;$ 取Ra和Rb2的排他逻辑和。		
汇编·助记	格式	影响标记	受影响的标记
xor Rc,Ra,Rb	32bit	—	—
备注			

图314

xor Ra2,Rb2

操作	$Ra2 \leftarrow Ra2 \wedge Rb2;$ 取Ra和Rb2的排他逻辑和，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
xor Ra2,Rb2	16bit	—	—
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图315

xor Rb,Ra,l8

操作	$Rb \leftarrow Ra \wedge \text{uext}(l8);$ 取Ra和Rb2的排他逻辑和。		
汇编·助记	格式	影响标记	受影响的标记
xor Rb,Ra,l8	32bit	—	—
备注			
l8为无符号的值			

图316

not Rb,Ra

操作	$\Rightarrow \text{rsub } Rb,Ra,-1$ $Rb \leftarrow \text{sext}(-1) - Ra;$ 反转Ra的各位。		
汇编·助记	格式	影响标记	受影响的标记
not Rb,Ra	相当于32bit	—	—
备注			

图317

not Ra2

操作	\Rightarrow rsub Ra2,-1 Ra2 <- sext(14) - Ra2; 反转Ra2的各位，存储于Ra2		
汇编·助记	格式	影响标记	受影响的标记
not Ra2	相当于16bit	-	-
备注			

图318

cmpCC Cm,Ra,Rb

操作	\Rightarrow cmpCCn Cm,Ra,Rb,C7 Cm <- result ; Ra和Rb进行比较，比较结果存储于Cm。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC Cm,Ra,Rb	相当于32bit	Cm	-
备注			
在cc中下一个cc比较条件可记载。			

图319

cmpCC C6,Ra2,Rb2

操作	C6 <- result; Ra2和Rb2进行比较，比较结果存储于C6。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC C6,Ra2,Rb2	16bit	C6	—
备注			
在cc中下一个cc比较条件可记载。 可能使用的判定只是C7（不能进行条件执行）			

图320

cmpCC Cm,Ra,I5

操作	=> cmpCCn Cm,Ra,I5,C7 Cm <- result; Ra2和Rb2进行比较，比较结果存储于C6。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC Cm,Ra,I5		Cm	—
备注			
在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时，I5为有符号的值，符号扩展后进行比较。 cc为gtu/geu/leu/ltu时，I5为无符号的值。			

图321

cmpCC C6,Ra2,I4

操作	C6 <- result; Ra2和当前值 (I4) 进行比较, 比较结果存储于C6。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC C6,Ra2,I4	16bit	C6	-
备注			
在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu可能使用的判定只是C7 (不能进行条件执行)			

图322

cmpCC Cm:Cm+1,Ra,Rb

操作	=> cmpCCn Cm:Cm+1,Ra,Rb,C7 Cm <- result; Cm+1 <- ~result; Ra和Rb进行比较, 比较结果存储于Cm和Cm+1。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC Cm:Cm+1,Ra,Rb		Cm,Cm+1	-
备注			
在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu请使m为偶数。			

图 323

cmpCC Cm:Cm+1,Ra,I5

操作	=> cmpCCn Cm:Cm+1,Ra,I5,C7		
	Cm <- result; Cm+1 <- ~result; Ra和当前值 (I5) 进行比较, 比较结果存储于Cm和Cm+1。		
汇编·助记	格式	影响标记	受影响的标记
cmpCC Cm:Cm+1,Ra,I5	相当于32bit	Cm,Cm+1	—
备注			
请使m为偶数 在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时, I5为有符号的值, 符号扩展后进行比较。 cc为gtu/geu/leu/ltu时, I5为无符号的值。			

图 324

cmpCCn Cm,Ra,Rb,Cn

操作	Cm <- result & Cn; 取Ra与Rb的比较结果, 和与cm的定位条件, 其结果存储于Cm。		
	汇编·助记	格式	影响标记
cmpCCn Cm,Ra,Rb,Cn	相当于32bit	Cm	Cn
备注			
在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu 请使m为偶数			

图325

cmpCCn Cm,Ra,I5,Cn

操作	Cm <- result & Cn; Ra和当前值 (I5) 进行比较, 比较结果存储于Cm。		
汇编·助记	格式	影响标记	受影响的标记
cmpCCn Cm,Ra,I5,Cn	32bit	Cm	Cn
备注			
请使m为偶数 在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时, I5为有符号的值, 符号扩展后进行比较。 cc为gtu/geu/leu/ltu时, I5为无符号的值。			

图326

cmpCCn Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- result & Cn; Cm+1 <- ~result & Cn; Ra和Rb进行比较, 比较结果存储于Cm和Cm+1		
汇编·助记	格式	影响标记	受影响的标记
cmpCCn Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu 请使m为偶数			

图327

cmpCCn Cm:Cm+1,Ra,I5,Cn

操作	Cm <- result & Cn; Cm+1 <- ~result & Cn; 取Ra和当前值 (I5) 进行比较, 比较结果存储于Cm和Cm+1		
汇编·助记	格式	影响标记	受影响的标记
cmpCCn Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数 在cc中下一个cc比较条件可记载eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时, I5为有符号的值, 符号扩展后进行比较。 cc为gtu/geu/leu/ltu时, I5为无符号的值。			

图328

cmpCCa Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- result & Cn; Cm+1 <- ~(result & Cn); 取Ra和当前值Rbr 比较结果与Cn的“与”条件, 存储于Cm和Cm+1。		
汇编·助记	格式	影响标记	受影响的标记
cmpCCa Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
在cc中可记载下一个cc比较条件。eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu 请使m为偶数			

图 329

cmpCCa Cm:Cm+1,Ra,I5,Cn

操作	Cm <- result & Cn; Cm+1 <- ~(result & Cn); 取Ra和当前值 (I5) 比较结果与Cn的“与”条件 存储于Cm和Cm+1		
汇编·助记	格式	影响标记	受影响的标记
cmpCCa Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数 在cc中可记载下一个cc比较条件eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时, I5为有符号的值, 符号扩展后进行比较。 cc为gtu/geu/leu/ltu时, I5为无符号的值。			

图 330

cmpCCo Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- result Cn; Cm+1 <- ~(result Cn); 取Ra和Rb的比较结果和与cn的“或”条件, 其结果 存储于Cm和Cm+1		
汇编·助记	格式	影响标记	受影响的标记
cmpCCo Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
在cc中可记载下一个cc比较条件eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu 请使m为偶数			

图331

cmpCCo Cm:Cm+1,Ra,I5,Cn

操作	Cm <- result Cn; Cm+1 <- ~(result Cn); 取Ra和Rb的比较结果和与cn的“或”条件，其结果存储于Cm和Cm+1		
汇编·助记	格式	影响标记	受影响的标记
cmpCCo Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数 在ccl中可记载下一个cc比较条件eq/ne/gt/ge/gtu/geu/le/lt/leu/ltu cc为eq/ne/gt/ge/le/lt时，I5为有符号的值，符号扩展后进行比较。 cc为gtu/geu/leu/ltu时，I5为无符号的值。			

图332

tstz Cm,Ra,Rb

操作	=> tstzn Cm,Ra,Rb,C7 Cm <- tstz(Ra,Rb); Ra和Rb的位测试结果存储于Cm。当逻辑积为0时位测试输入1。		
汇编·助记	格式	影响标记	受影响的标记
tstz Cm,Ra,Rb	相当于32bit	Cm	-
备注			

图333

tstz C6,Ra2,Rb2

操作	C6 <- tstz(Ra2,Rb2); 进行Ra2和Rb2的位测试，结果存储于C6。当逻辑积为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstz C6,Ra2,Rb2	16bit	C6	—
备注			
可能使用的判定只是C7（不能进行条件执行）			

图334

tstz Cm,Ra,I5

操作	=> tstzn Cm,Ra,I5,C7 Cm <- tstz(Ra,uext(I5)); Ra和定值（I5）的位测试结果存储于Cm。当逻辑积为0时位测试输出1。		
编程·助记	幅度	影响标记	受影响的标记
tstz Cm,Ra,I5	相当于32bit	Cm	—
备注			
I5为无符号的值。			

图 335

tstz C6,Ra2,I4

操作	C6 <- tstz(Ra2,uxxt(I4)); 进行Ra和Rb2的位测试，结果存储于C6。当逻辑积为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstz C6,Ra2,I4	16bit	C6	-
备注			
I4为无符号的值。 可能使用的判定只是C7（不能进行条件执行）			

图 336

tstz Cm:Cm+1,Ra,Rb

操作	=> tstzn Cm:Cm+1,Ra,Rb,C7 Cm <- tstz(Ra,Rb); Cm+1 <- ~tstz(Ra,Rb); Ra和Rb的位测试结果存储于Cm和Cm+1。当逻辑积为0时位测试输出0。		
汇编·助记	格式	影响标记	受影响的标记
tstz Cm:Cm+1,Ra,Rb	相当于32bit	Cm,Cm+1	-
备注			
请使m为偶数。			

图 337

tstz Cm:Cm+1,Ra,I5

操作	⇒ tstzn Cm:Cm+1,Ra,I5,C7		
	Cm ← tstz(Ra,uext(I5)); Cm+1 ← ~tstz(Ra,uext(I5)); Ra和当前值 (I5) 的位测试结果存储于Cm和Cm+1。 当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstz Cm:Cm+1,Ra,I5	相当于32bit	Cm,Cm+1	—
备注			
请使m为偶数 I5为无符号的值。			

图 338

tstzn Cm,Ra,Rb,Cn

操作	Cm ← tstz(Ra,Rb) & Cn;		
	取Ra和Rb的位测试结果和与cn的嵌套条件，结果 存储于Cm。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzn Cm,Ra,Rb,Cn	32bit	Cm	Cn
备注			

图339

tstzn Cm,Ra,I5,Cn

操作	Cm <- tstz(Ra,uext(I5)) & Cn; 取Ra和Rb的位测试结果和与cn的嵌套条件，结果存储于Cm。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzn Cm,Ra,I5,Cn	32bit	Cm	Cn
备注			
I5为无符号的值。			

图340

tstzn Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- tstz(Ra,Rb) & Cn; Cm+1 <- ~tstz(Ra,Rb) & Cn; 取Ra和Rb的位测试结果和与cn的嵌套条件，结果存储于Cm。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzn Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数			

图341

tstzn Cm:Cm+1,Ra,I5,Cn

操作:	Cm <- tstz(Ra,uext(I5)) & Cn; Cm+1 <- ~tstz(Ra,uext(I5)) & Cn; 取Ra和Rb的位测试结果和与cn的嵌套条件，结果存储于Cm。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzn Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数 I5为无符号的值。			

图342

tstza Cm:Cm+1,Ra,Rb,Cn

操作:	Cm <- tstz(Ra,Rb) & Cn; Cm+1 <- ~(tstz(Ra,Rb) & Cn); 取Ra和Rb的位测试结果和与cn的嵌套条件，结果存储于Cm。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstza Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数			

tstza Cm:Cm+1,Ra,I5,Cn

操作	$Cm \leftarrow tstz(Ra, uext(I5)) \& Cn;$ $Cm+1 \leftarrow \sim(tstz(Ra, uext(I5)) \& Cn);$		
	取Ra和当前值(I5)的位测试结果和与Cn的“与”条件， 结果存储于Cm和Cm+1。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstza Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。 I5为无符号的值。			

图 343

tstzo Cm:Cm+1,Ra,Rb,Cn

操作	$Cm \leftarrow tstz(Ra, Rb) Cn;$ $Cm+1 \leftarrow \sim(tstz(Ra, Rb) Cn);$		
	取Ra和Rb的位测试结果和与Cn的“或”条件， 结果有存储于Cm和Cm+1。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzo Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。			

图 344

tstzo Cm:Cm+1,Ra,I5,Cn

操作	Cm <- tstz(Ra,uext(I5)) Cn; Cm+1 <- ~(tstz(Ra,uext(I5)) Cn);		
	取Ra和当前值 (I5) 的位测试结果和与Cn的“或”条件， 结果有存储于Cm和Cm+1。当逻辑积为0时位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstzo Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。 I5为无符号的值。			

图345

tstn Cm,Ra,Rb

操作	=> tstnn Cm,Ra,Rb,C7 Cm <- tstn(Ra,Rb); Ra和Rb的位测试结果存储于Cm。 当逻辑积不为0时，位测试输出1。		
	汇编·助记	格式	影响标记
tstn Cm,Ra,Rb	相当于32bit	Cm	-
备注			

图346

tstn C6,Ra2,Rb2

操作	C6 <- tstn(Ra2,Rb2); 进行Ra2和Rb2的位测试，结果存储于C6。 当逻辑积的结果不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstn C6,Ra2,Rb2	16bit	C6	—
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图347

tstn Cm,Ra,I5

操作	=> tstnn Cm,Ra,I5,C7 Cm <- tstn(Ra,uext(I5)); Ra和定值（I5）的位测试结果存储于Cm。 当逻辑积不为0时，位测试输入1。		
编程·助记	幅度	影响标记	受影响的标记
tstn Cm,Ra,I5	32bitシノニム	Cm	—
备注			
I5为无符号的值。			

图348

tstn C6,Ra2,I4

操作	C6 <- tstn(Ra2,uext(I4)); 进行Ra2和Rb2的位测试，结果存储于C6。 当逻辑积的结果不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstn C6,Ra2,I4	16bit	C6	—
备注			
I4为无符号的值。 可能使用的判定只是C7。（不可进行条件执行）			

图349

tstn Cm:Cm+1,Ra,Rb

操作	=> tstn Cm:Cm+1,Ra,Rb,C7 Cm <- tstn(Ra,Rb); Cm+1 <- ~tstn(Ra,Rb); Ra和Rb的位测试结果存储于Cm和Cm+1。 当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstn Cm:Cm+1,Ra,Rb	相当于32bit	Cm,Cm+1	—
备注			
请使m为偶数。			

图350

tstn Cm:Cm+1,Ra,I5

操作	\Rightarrow tstnn Cm:Cm+1,Ra,I5,C7 Cm <- tstn(Ra,uext(I5)); Cm+1 <- ~tstn(Ra,uext(I5)); Ra和定值 (I5) 的位测试结果存储于Cm和Cm+1。 当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstn Cm:Cm+1,Ra,I5	相当于32bit	Cm,Cm+1	-
备注			
请使m为偶数。 I5为无符号的值。			

图351

tstnn Cm,Ra,Rb,Cn

操作	Cm <- tstn(Ra,Rb) & Cn; 取Ra和Rb的位测试结果和与Cn的嵌套条件，其结果存储于Cm。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstnn Cm,Ra,Rb,Cn	32bit	Cm	Cn
备注			

图352

tstnn Cm,Ra,I5,Cn

操作	Cm <- tstn(Ra,uext(I5)) & Cn; 取Ra和当前值(I5)的位测试结果和与Cn的嵌套条件，其结果存储于Cm。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstnn Cm,Ra,I5,Cn	32bit	Cm	Cn
备注			
I5为无符号的值。			

图353

tstnn Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- tstn(Ra,Rb) & Cn; Cm+1 <- ~tstn(Ra,Rb) & Cn; 取Ra和Rb的位测试结果和与Cn的嵌套条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstnn Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。			

图354

tstnn Cm:Cm+1,Ra,I5,Cn

操作	Cm <- tstn(Ra,uext(I5)) & Cn; Cm+1 <- ~tstn(Ra,uext(I5)) & Cn;		
	取Ra和当前值(I5)的位测试结果和与Cn的嵌套条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstnn Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。 I5为无符号的值。			

图 355

tstna Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- tstn(Ra,Rb) & Cn; Cm+1 <- ~(tstn(Ra,Rb) & Cn);		
	取Ra和Rb的位测试结果和与Cn的“与”条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstna Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。			

图 356

tstna Cm:Cm+1,Ra,I5,Cn

操作	Cm <- tstn(Ra,uext(I5)) & Cn; Cm+1 <- ~(tstn(Ra,uext(I5)) & Cn);		
	取Ra和定值(I5)的位测试结果和与Cn的“或”条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
编程·助记	幅度	影响标记	受影响的标记
tstna Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。 I5为无符号的值。			

图357

tstno Cm:Cm+1,Ra,Rb,Cn

操作	Cm <- tstn(Ra,Rb) Cn; Cm+1 <- ~(tstn(Ra,Rb) Cn);		
	取Ra和Rb的位测试结果和与Cn的“或”条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstno Cm:Cm+1,Ra,Rb,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。			

图358

tstno Cm:Cm+1,Ra,I5,Cn

操作	$Cm \leftarrow \text{tstn}(Ra, \text{uext}(I5)) Cn;$ $Cm+1 \leftarrow \sim(\text{tstn}(Ra, \text{uext}(I5)) Cn);$ 取Ra和当前值(I5)的位测试结果和与Cn的“或”条件，其结果存储于Cm和Cm+1。当逻辑积不为0时，位测试输出1。		
汇编·助记	格式	影响标记	受影响的标记
tstno Cm:Cm+1,Ra,I5,Cn	32bit	Cm,Cm+1	Cn
备注			
请使m为偶数。 I5为无符号的值。			

图359

mov Rb,Ra

操作	$\Rightarrow \text{or } Rb, Ra, 0$ $Rb \leftarrow Ra;$ 将Ra传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,Ra	相当于bit	-	-
备注			

图360

mov Ra2,Rb

操作	Ra2 <- Rb; 将Rb的值传输至Ra2。		
汇编·助记	格式	影响标记	受影响的标记
mov Ra2,Rb	16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图361

mov Ra,I16

操作	Ra <- sext(I16); 在Ra处将值(I16)符号扩展并存储。		
汇编·助记	格式	影响标记	受影响的标记
mov Ra,I16	32bit	-	-
备注			
I16为有符号的值。			

图362

mov Ra2,I8

操作	Ra2 <- sext(I8); 当前值(I18)符号扩展, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
mov Ra2,I8	16bit	—	—
备注			
I8有符号的值。 可能使用的判定只是C7。(不可进行条件执行)			

图363

mov Rb,TAR

操作	Rb <- TAR; 将TAR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,TAR	32bit	—	—
备注			

图364

mov Rb2,TAR

操作	Rb2 <- TAR; 将TAR的值传输至Rb2。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb2,TAR	16bit	—	—
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图365

mov Rb,LR

操作	Rb <- LR; 将在LR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,LR	32bit	—	—
备注			

图366

mov Rb2,LR

操作	Rb2 <- LR; 将LR的值传输至Rb2。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb2,LR	16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图367

mov Rb,SVR

操作	Rb[31:16] <- 0x0000; Rb[15: 0] <- SVR; 将SVR的值传输至Rb的下位16比特。 在上位16比特处输入0。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,SVR	32bit	-	SVR
备注			

图368

mov Rb,PSR

操作	Rb <- PSR; 将PSR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PSR	32bit	—	PSR
备注			

图369

mov Rb,CFR

操作	Rb <- CFR; 将CFR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,CFR	32bit	—	CFR
备注			

图370

mov Rb,MH0

操作	Rb <- MH0; 将MH0的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,MH0	32bit	—	—
备注			

图371

mov Rb2,MH0

操作	Rb2 <- MH0; 将MH0的值传输至Rb2。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb2,MH0	16bit	—	—
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图372

mov Rb,MH1

操作	Rb <- MH1; 将MH1的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,MH1	32bit	-	-
备注			

图373

mov Rb2,MH1

操作	Rb2 <- MH1; 将MH0的值传输至Rb2。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb2,MH1	16bit	-	-
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图374

7

mov Rb,ML0

操作	Rb <- ML0; 将ML0的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,ML0	32bit	-	-
备注			

图375

mov Rb,ML1

操作	Rb <- ML1; 将ML1的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,ML1	32bit	-	-
备注			

图376

mov Rb,IPC

操作	Rb <- IPC; 将IPC的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,IPC	32bit	—	—
备注			

图377

mov Rb,IPSR

操作	Rb <- IPSR; 将EPSR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,IPSR	32bit	—	—
备注			
可能使用的判定只是C7。（不可进行条件执行）			

图378

mov Rb,PC

操作	Rb <- next_PC; 将IPC的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PC	32bit	-	-
备注			

图379

mov Rb,EPC

操作	Rb <- EPC; 将EPC的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,EPC	32bit	-	-
备注			

图380

mov Rb,EPSR

操作	Rb <- EPSR; 将EPSR的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,EPSR	32bit	-	-
备注			

图381

mov Rb,PSR0

操作	Rb[7:0] <- PSR[31:24]; Rb[31:8] <- 0; 将PSR0的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PSR0	32bit	-	PSR0
备注			

图382

mov Rb,PSR1

操作	$Rb[7:0] \leftarrow PSR[23:16]; Rb[31:8] \leftarrow 0;$ 将PSR1的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PSR1	32bit	—	PSR1
备注			

图383

mov Rb,PSR2

操作	$Rb[7:0] \leftarrow PSR[15: 8]; Rb[31:8] \leftarrow 0;$ 将PSR2的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PSR2	32bit	—	PSR2
备注			

图384

mov Rb,PSR3

操作	$Rb[7:0] \leftarrow PSR[7:0]; Rb[31:8] \leftarrow 0;$ 将PSR3的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,PSR3	32bit	—	PSR3
备注			

图 385

mov Rb,CFR0

操作	$Rb[7:0] \leftarrow CFR[31:24]; Rb[31:8] \leftarrow 0;$ 将CFR0的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,CFR0	32bit	—	CFR0
备注			

图 386

mov Rb,CFR1

操作	Rb[7:0] <- CFR[23:16]; Rb[31:8] <- 0; 将CFR1的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,CFR1	32bit	—	CFR1
备注			

图387

mov Rb,CFR2

操作	Rb[7:0] <- CFR[15: 8]; Rb[31:8] <- 0; 将CFR3的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,CFR2	32bit	—	CFR2
备注			

图388

mov Rb,CFR3

操作	Rb[7:0] <- CFR[7: 0]; Rb[31:8] <- 0; 将CFR3的值传输至Rb。		
汇编·助记	格式	影响标记	受影响的标记
mov Rb,CFR3	32bit	—	CFR3
备注			

图 389

mov LR,Rb

操作	LR <- Rb; 将Rb的值传输至LR。		
汇编·助记	格式	影响标记	受影响的标记
mov LR,Rb	32bit	—	—
备注			
不进行向分支用命令缓冲器的命令填充。			

图 390

mov LR,Rb2

操作	LR <- Rb2; 将Rb2的值传输至LR。		
汇编·助记	格式	影响标记	受影响的标记
mov LR,Rb2	16bit	—	—
备注			
可能使用的判定只是C7。（不可进行条件执行） 不进行向分支用命令缓冲器的命令填充。			

图391

mov TAR,Rb

操作	TAR <- Rb; 将Rb的值传输至TAR。		
汇编·助记	格式	影响标记	受影响的标记
mov TAR,Rb	32bit	—	—
备注			
不进行向分支用命令缓冲器的命令填充。			

图392

mov TAR,Rb2

操作	TAR <- Rb2; 将Rb2的值传输至TAR。		
汇编·助记	格式	影响标记	受影响的标记
mov TAR,Rb2	16bit	—	—
备注			
可能使用的判定只是C7。（不可进行条件执行） 不进行向分支用命令缓冲器的命令填充。			

图393

mov SVR,Rb

操作	SVR <- Rb[16:0]; 将Rb的下位16比特的值传输至SCR。		
汇编·助记	格式	影响标记	受影响的标记
mov SVR,Rb	32bit	SVR	—
备注			

图394

mov PSR,Rb

操作	PSR <- Rb; 将Rb的值传输至PSR。		
汇编·助记	格式	影响标记	受影响的标记
mov PSR,Rb	32bit	PSR	—
备注			
通过特权标准0.2可能执行。 通过特权标准3时发生特权命令执行例外。			

图395

mov CFR,Rb

操作	CFR <- Rb; 将Rb的值传输至CFR。		
汇编·助记	格式	影响标记	受影响的标记
mov CFR,Rb	32bit	CFR	—
备注			

图396

mov MH0,Rb

操作	MH0 <- Rb; 将Rb的值传输至MH0。		
汇编助记	格式	影响标记	受影响的标记
mov MH0,Rb	32bit	-	-
备注			
在积和命令中以便MHm/MLm为目的，在MHm/MLm处存储0时请使用mul,Mm,r31,Ra,0。积命令以外的命令中检测乘法/积和后的MH使用的危险性，产生执行时间。			

图397

mov MH0,Rb2

操作	MH0 <- Rb2; 将Rb2的值传输至MH0。		
汇编助记	格式	影响标记	受影响的标记
mov MH0,Rb2	16bit	-	-
备注			
使用可能的判定只是C7。（不可进行条件执行）			

图398

mov MH1,Rb

操作	MH1 <- Rb; 将Rb的值传输至MH1。		
汇编·助记	格式	影响标记	受影响的标记
mov MH1,Rb	32bit	—	—
备注			
在积和命令中以使MHm/MLm为目的，在MHm/MLm处存储0时 请使用mul,Mm,r31,Ra,0。积命令以外的命令中检测乘法/积 和后的MH使用的危险性，产生执行时间。			

图399

mov MH1,Rb2

操作	MH1 <- Rb2; 将Rb2的值传输至MH1。		
汇编·助记	格式	影响标记	受影响的标记
mov MH1,Rb2	16bit	—	—
备注			
使用可能的判定只是C7。（不可进行条件执行）			

图400

mov ML0,Rb

操作	ML0 ← Rb; 将Rb的值传输至ML0。		
汇编·助记	格式	影响标记	受影响的标记
mov ML0,Rb	32bit	—	—
备注			
在积和命令中以使MHm/MLm为目的，在MHm/MLm处存储0时 请使用mul,Mm,r31,Ra,0。积命令以外的命令中检测乘法/积和 后的MH使用的危险性，产生执行时间。			

图401

mov ML1,Rb

操作	ML1 ← Rb; 将Rb的值传输至ML1。		
汇编·助记	格式	影响标记	受影响的标记
mov ML1,Rb	32bit	—	—
备注			
在积和命令中以使MHm/MLm为目的，在MHm/MLm处存储0时 请使用mul,Mm,r31,Ra,0。积命令以外的命令中检测乘法/积和 后的MH使用的危险性，产生执行时间。			

图402

mov IPC,Rb

操作	IPC ← Rb; 将Rb的值传输至IPC。		
汇编·助记	格式	影响标记	受影响的标记
mov IPC,Rb	32bit	—	—
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图403

mov IPSR,Rb

操作	IPSR ← Rb; 将Rb的值传输至PSR。		
汇编·助记	格式	影响标记	受影响的标记
mov IPSR,Rb	32bit	—	—
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图404

mov EPC,Rb

操作	EPC <- Rb; 将Rb的值传输至EPC。		
汇编·助记	格式	影响标记	受影响的标记
mov EPC,Rb	32bit	—	—
备注			
通过特权标准0.2可能执行。 通过特权标准3时发生特权命令执行例外。			

图405

mov EPSR,Rb

操作	EPSR <- Rb; 将Rb的值传输至EPSR。		
汇编·助记	格式	影响标记	受影响的标记
mov EPSR,Rb	32bit	—	—
备注			
通过特权标准0.2可能执行。 通过特权标准3时发生特权命令执行例外。			

图406

mov PSR0,Rb

操作	PSR[31:24] <- Rb[7:0]; 将Rb的值传输至PSR0。		
汇编·助记	格式	影响标记	受影响的标记
mov PSR0,Rb	32bit	PSR0	-
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图407

mov PSR1,Rb

操作	PSR[23:16] <- Rb[7:0]; 将Rb的值传输至PSR1。		
汇编·助记	格式	影响标记	受影响的标记
mov PSR1,Rb	32bit	PSR1	-
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图408

mov PSR2,Rb

操作	PSR[23:16] <- Rb[7:0]; 将Rb的值传输至PSR2。		
汇编·助记	格式	影响标记	受影响的标记
mov PSR2,Rb	32bit	PSR2	—
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图409

mov PSR3,Rb

操作	PSR[23:16] <- Rb[7:0]; 将Rb的值传输至PSR3。		
汇编·助记	格式	影响标记	受影响的标记
mov PSR3,Rb	32bit	PSR3	—
备注			
通过特权标准0, 2可能执行。 通过特权标准3时发生特权命令执行例外。			

图410

mov CFR0,Rb

操作	CFR[31:24] ← Rb[7:0]; 将Rb的值传输至CFR0。		
汇编·助记	格式	影响标记	受影响的标记
mov CFR0,Rb	32bit	CFR	—
备注			

图411

mov CFR1,Rb

操作	CFR[23:16] ← Rb[7:0]; 将Rb的值传输至CFR1。		
汇编·助记	格式	影响标记	受影响的标记
mov CFR1,Rb	32bit	CFR1	—
备注			

图412

mov CFR2,Rb

操作	CFR[15: 8] ← Rb[7:0]; 将Rb的值传输至CFR2。		
汇编·助记	格式	影响标记	受影响的标记
mov CFR2,Rb	32bit	CFR2	—
备注			

图413

mov CFR3,Rb

操作	CFR[7: 0] ← Rb[7:0]; 将Rb的值传输至CFR31。		
汇编·助记	格式	影响标记	受影响的标记
mov CFR3,Rb	32bit	CFR3	—
备注			

图414

movclovs Cm:Cm+1

操作	$Cm:Cm+1 \leftarrow CFR.OVS:~CFR.OVS; CFR.OVS \leftarrow 0;$ 清除OFR.OVS, 传输至Cm和Cm+1。		
汇编·助记	格式	影响标记	受影响的标记
movclovs Cm:Cm+1	16bit	Cm,Cm+1,CFR.OVS	CFR.OVS
备注			
请使Cm为偶数。			

图415

movcf Ci,Cj,Cm,Cn

操作	$Ci \leftarrow Cj; Cm \leftarrow Cn;$ 进行标志的移位。将Cj传输至Ci, Cn传输至Cm。		
汇编·助记	格式	影响标记	受影响的标记
movcf Ci,Cj,Cm,Cn	32bit	Ci,Cm	Cj,Cn
备注			
Ci和Cm中不能指定同一的标志。 如指定时的操作是不定的(安装相关)。			

图416

mvclcas Cm:Cm+1

操作	$\text{Cm:Cm+1} \leftarrow \text{CFR.CAS}; \sim\text{CFR.CAS}; \text{CFR.CAS} \leftarrow 0;$ <p>清除OFR.OVS, 传输至Cm和Cm+1。</p>		
汇编·助记	格式	影响标记	受影响的标记
mvclcas Cm:Cm+1	16bit	Cm,Cm+1,CFR.CAS	CFR.CAS
备注			
请使Cm为偶数。			

图417

sethi Ra,I16

操作	$\text{Ra} \leftarrow (\text{uext}(I16) \ll 16) + \text{uext}(\text{Ra}[15:0]);$ <p>Ra的上位16比特中存储当前值(I16)。 不影响Ra的下位16比特。</p>		
汇编·助记	格式	影响标记	受影响的标记
sethi Ra,I16	32bit	-	-
备注			
I16为无符号的值。			

图418

图419

setlo Ra,I16

操作	=> mov Ra,I16 Ra <- sext(I16); Ra中当前值 (I16) 符号扩展后存储。		
汇编·助记	格式	影响标记	受影响的标记
setlo Ra,I16	相当于32bit	-	-
备注			
I16写做无符号的值，但在Ra中写入将I16的最上位位看做符号位并进行符号扩展后的值。I16写为负值时就是汇编错误和sethi·命令，将32位值写入寄存器时使用。			

图420

vcchk

操作	if (VC3:VC2:VC1:VC0==0b0000) { C4:C5 <- 1:0; } else { C4:C5 <- 0:1; } C3:C2:C1:C0 <- VC3:VC2:VC1:VC0; 判断VC0~VC3是否全为0，输入C4和C5。 将VC0~VC3转送至C0~C3。		
汇编·助记	格式	影响标记	受影响的标记
vcchk	16bit	C0,C1,C2,C3,C4,C5	VC0,VC1,VC2,VC3
备注			

图421

nop

操作	no operation 什么都不执行。		
汇编·助记	格式	影响标记	受影响的标记
nop	16bit	-	-
备注			
不产生危险。			

图422

asl Rc,Ra,Rb

操作	$\text{tmp}[5:0] \leftarrow \text{Rb}[5:0];$ $\text{tmp}[5:0] \leftarrow 31, \text{ if } (\text{Rb} > 0\text{x}0000_001\text{f});$ $\text{tmp}[5:0] \leftarrow -31, \text{ if } (\text{Rb} < 0\text{xffff_ffe1});$ $\text{Rc} \leftarrow \text{Ra} \ll \text{abs}(\text{tmp}[5:0]);$ $\text{Rc} \leftarrow \text{Ra} \gg \text{abs}(\text{tmp}[5:0]), \text{ if } (\text{tmp}[5:0] < 0);$ Ra算术左移位Rb所示的位数。Rb在±31内饱和，如是负值时则算术右移位。		
汇编·助记	格式	影响标记	受影响的标记
asl Rc,Ra,Rb	32bit	-	-
备注			

图423

asl Rb,Ra,I5

操作	$Rb \leftarrow Ra \ll I5[4:0];$ Ra算术左移位当前定值 (I5) 位数。		
汇编·助记	格式	影响标记	受影响的标记
asl Rb,Ra,I5	32bit	—	—
备注			
I5为无符号的值。			

图424

asl Ra2,I4

操作	$Ra2 \leftarrow Ra2 \ll \text{uext}(I4);$ $Ra2 \leftarrow Ra2 \ll 16, \text{if } (I4 == 0);$ Ra2算术左移位当前值 (I4) 位。当前值 (I4) 为0时, 16位算术左移位。		
汇编·助记	格式	影响标记	受影响的标记
asl Ra2,I4	16bit	—	—
备注			
I4为有符号的值。 可能使用的判定只是C7。(不能进行条件执行)			

图425

aslvw Rc,Ra,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[62:0] <- sext63(Ra[31:0]) << abs(tmp[5:0]); tmp2[62:0] <- sext63(Ra[31:0]) >> abs(tmp[5:0]), if (tmp[5:0] < 0); Rc <- tmp2[31:0]; Rc <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); Rc <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); </pre> <p>Ra算术左移位Rb所示的位数。结果不用有符号的32位表示时饱和。Rb在±31内饱和，如果是负值时，则算术右移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslvw Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图426

aslww Rb,Ra,I5

操作	<pre> tmp[62:0] <- sext63(Ra[31:0]) << I5[4:0]; Rb <- tmp[31:0]; Rb <- 0x7fff_fff, CFR.OVS <- 1, if (tmp[62:0] > 0x0000_0000_7fff_fff[62:0]); Rb <- 0x8000_0000, CFR.OVS <- 1, if (tmp[62:0] < 0x7fff_fff_8000_0000[62:0]); </pre> <p>Ra算术左移位Rb当前值 (I5) 的位数。结果不用有符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslww Rb,Ra,I5	32bit	CFR.OVS	-
备注			

图427

aslvh Rc,Ra,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[46:0] <- sext47(Ra[15:0]) << abs(tmp[5:0]); tmp2[46:0] <- sext47(Ra[15:0]) >> abs(tmp[5:0]), if (tmp[5:0] < 0); Rc <- sext(tmp2[15:0]); Rc <- 0x0000_7fff, CFR.OVS <- 1, if (tmp2[46:0] > 0x0000_0000_7fff[46:0]); Rc <- 0xffff_8000, CFR.OVS <- 1, if (tmp2[46:0] < 0x7fff_ffff_8000[46:0]); </pre> <p>Ra算术左移位Rb所示的位数。结果不用有符号的16位表示时饱和。Rb在±31内饱和，如果是负值时，则算术右移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslvh Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图428

aslvh Rb,Ra,I5

操作	<pre>tmp[46:0] <- sext47(Ra[[15:0]) << I5[4:0]; Rb <- tmp[31:0]; Rb <- 0x0000_7fff, CFR.OVS <- 1, if (tmp[46:0] > 0x0000_0000_7fff[46:0]); Rb <- 0xffff_8000, CFR.OVS <- 1, if (tmp[46:0] < 0x7fff_fff_8000[46:0]);</pre> <p>Ra算术左移位Rb当前值 (I5) 的位数。结果不用有符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslvh Rb,Ra,I5	32bit	CFR.OVS	—
备注			

图429

asr Rc,Ra,Rb

操作	<pre>tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); Rc <- Ra >> abs(tmp[5:0]); Rc <- Ra << abs(tmp[5:0]), if (tmp[5:0] < 0);</pre> <p>Ra算术右移位Rb所示的位数。Rb在±31内饱和，如是负值时则算术左移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
asr Rc,Ra,Rb	32bit	—	—
备注			

图430

asr Rb,Ra,I5

操作	Rb <- Ra >> I5[4:0]; Ra算术右移位当前值 (I5) 位数。		
汇编·助记	格式	影响标记	受影响的标记
asr Rb,Ra,I5	32bit	-	-
备注			
I5为无符号的值。			

图431

asr Ra2,I4

操作	Ra2 <- Ra2 >> uext(I4); Ra2 <- Ra2 >> 16, if (I4 == 0); Ra2进行算术左移位当前值 (I4) 位。当前值 (I4) 为0时，16位算术左移位。		
汇程·助记	格式	影响标记	受影响的标记
asr Ra2,I4	16bit	-	-
备注			
I4为无符号的值。 可能使用的判定只是C7。(不能进行条件执行)			

图432

asrww Rc,Ra,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[62:0] <- sext63(Ra[31:0]) >> abs(tmp[5:0]); tmp2[62:0] <- sext63(Ra[31:0]) << abs(tmp[5:0]), if (tmp[5:0] < 0); Rc <- tmp2[31:0]; Rc <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); Rc <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); Ra算术右移位Rb所示的位数。结果不用 有符号的32位表示时饱和。Rb在±31内饱和， 如果是负值时，则算术左移位。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
asrww Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图433

asrvh Rc,Ra,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[46:0] <- sext47(Ra[15:0]) >> abs(tmp[5:0]); tmp2[46:0] <- sext47(Ra[15:0]) << abs(tmp[5:0]), if (tmp[5:0] < 0); Rc <- sext(tmp2[15:0]); Rc <- 0x0000_7fff, CFR.OVS <- 1, if (tmp2[46:0] > 0x0000_0000_7fff[46:0]); Rc <- 0xffff_8000, CFR.OVS <- 1, if (tmp2[46:0] < 0x7fff_fff_8000[46:0]); Ra算术右移位Rb所示的位数。结果不用 有符号的16位表示时饱和。Rb在±31内饱和， 如果是负值时，则算术左移位。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
asrvh Rc,Ra,Rb	32bit	CFR.OVS	-
备注			

图434

lsl Rc,Ra,Rb

操作	=> asl Rc,Ra,Rb		
	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); Rc <- Ra << abs(tmp[5:0]); Rc <- Ra >> abs(tmp[5:0]),if (tmp[5:0] < 0); </pre> <p>Ra逻辑左移位Rb所示的位数。Rb在±31内饱和，如是负值时则算术右移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
lsl Rc,Ra,Rb	相当于32bit	-	-
备注			
※ 移动量为负时，进行算术右移位。			

图435

lsl Rc,Ra,I5

操作	=> asl Rc,Ra,I5		
	<pre> Rb <- Ra << I5[4:0]; </pre> <p>Ra逻辑左移位当前值（I5）位数。</p>		
编程·助记	幅度	影响标记	受影响的标记
lsl Rc,Ra,I5	相当于32bit	-	-
备注			
I5为无符号的值。			

图436

lsl Ra2,I4

操作	\Rightarrow asl Ra2,I4 Ra2 <- Ra2 << uext(I4); Ra2 <- Ra2 << 16, if (I4 == 0); Ra2逻辑左移位当前值 (I4) 位数, 存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
lsl Ra2,I4	相当于32bit	-	-
备注			
I4为无符号的值。I4为0时，16位进行逻辑左移动。 可能使用的判定只是C7。（不可进行条件执行）			

图437

lsl Rc,Ra,Rb

操作	tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); Rc <- Ra >>> abs(tmp[5:0]); Rc <- Ra << abs(tmp[5:0]), if (tmp[5:0] < 0); Ra逻辑右移位Rb所示的位数。Rb在 ±31内饱和，如是负值时逻辑左移位。		
汇编·助记	格式	影响标记	受影响的标记
lsl Rc,Ra,Rb	32bit	-	-
备注			

图438

lsl Rb,Ra,I5

操作	Rb <- Ra >>> I5[4:0]; Ra逻辑右移位当前值（I5）的位数。		
汇编·助记	格式	影响标记	受影响的标记
lsl Rb,Ra,I5	32bit	—	—
备注			
I5是无符号的值			

图439

rol Rc,Ra,Rb

操作	tmp[31:0] <- Ra << Rb[4:0]; tmp2[31:0] <- Ra >>> (32 - Rb[4:0]); Rc <- tmp[31:0] tmp2[31:0]; Ra左旋转Rb所示的位数。		
汇编·助记	格式	影响标记	受影响的标记
rol Rc,Ra,Rb	32bit	—	—
备注			

图440

rol Rb,Ra,I5

操作	<pre>tmp [31:0] <- Ra << I5[4:0]; tmp2[31:0] <- Ra >>> (32 - I5[4:0]); Rb <- tmp[31:0] tmp2[31:0];</pre> <p>Ra左旋转当前值（I5）的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
rol Rb,Ra,I5	32bit	-	-
备注			
I5是无符号的值			

图441

ror Rb,Ra,I5

操作	<pre>=> rol Rb,Ra,(32-I5)&0x1f</pre> <pre>tmp [31:0] <- Ra << I5[4:0]; tmp2[31:0] <- Ra >>> (32 - I5[4:0]); Rb <- tmp[31:0] tmp2[31:0];</pre> <p>Ra右旋转当前值（I5）的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
ror Rb,Ra,I5	相当于32bit	-	-
备注			
I5是无符号的值			

图442

aslp Mm,Ra,Mn,Rb

操作	<pre>tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:MLn << abs(tmp[6:0]); MHm:MLm <- MHn:MLn >> abs(tmp[6:0]), if (tmp[6:0] < 0); Ra <- MLm;</pre> <p>Mn算术左移位Rb所示的位数。Rb在±63内饱和，如是负值时，则算术右移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslp Mm,Ra,Mn,Rb	32bit	-	-
备注			

图443

aslp Mm,Rb,Mn,I6

操作	<pre>MHm:MLm <- MHn:MLn << I6[5:0]; Rb <- MLm;</pre> <p>Mn算术左移位当前值（I6）的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
aslp Mm,Rb,Mn,I6	32bit	-	-
备注			
I6是无符号的值			

图444

aslp Mm,Rc,MHn,Ra,Rb

操作	tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:Ra << abs(tmp[6:0]); MHm:MLm <- MHn:Ra >> abs(tmp[6:0]),if (tmp[6:0] < 0); Rc <- MLm; MHn:Ra Mn算术左移位Rb所示的位数。Rb在±63 内饱和，如是负值时，则算术右移位。		
汇编·助记	格式	影响标记	受影响的标记
aslp Mm,Rc,MHn,Ra,Rb	32bit	-	-
备注			

图445

aslp Mm,Rb,MHn,Ra,I6

操作	MHm:MLm <- MHn:Ra << I6[6:0]; Rb <- MLm; MHn:Ra Mn算术右移位当前定值（I6）的位数。		
汇编·助记	格式	影响标记	受影响的标记
aslp Mm,Rb,MHn,Ra,I6	32bit	-	-
备注			
I6是无符号的值			

图446

aslpvw Mm,Ra,Mn,Rb

操作	<pre> tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); tmp2[126:0] <- sext127(MHn:MLn) << abs(tmp[6:0]); tmp2[126:0] <- sext127(MHn:MLn) >> abs(tmp[6:0]), if (tmp[6:0] < 0); MHm:MLm <- tmp2[63:0]; MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[126:0] > sext127(0x0_7fff_ffff[126:0])); MHm:MLm <- 0x0000_0000_8000_0000, CFR.OVS <- 1, if (tmp2[126:0] < sext127(0x1_8000_0000[126:0])); Ra <- MLm; MHn:Ra Mn算术左移位Rb所示的位数。 结果不能用有符号32位数表示时则饱和。 Rb在±63内饱和，如是负值时，则算术右移位。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
aslpvw Mm,Ra,Mn,Rb	32bit	CFR.OVS	—
备注			

图 447

aslpvw Mm,Rb,Mn,I6

操作	<pre> tmp[126:0] <- sext127(MHn:MLn) << I6[5:0]; MHm:MLm <- tmp[63:0]; MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp[126:0] > sext127(0x0_7fff_ffff); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp[126:0] < sext127(0x1_8000_0000); Rb <- MLm; Mn算术左移位当前值 (I6) 的位数。 结果不能用有符号32位数表示时则饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
aslpvw Mm,Rb,Mn,I6	32bit	CFR.OVS	—
备注			
I6是无符号的值			

图 448

asrp Mm,Ra,Mn,Rb

操作	<pre> tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:MLn >> abs(tmp[6:0]); MHm:MLm <- MHn:MLn << abs(tmp[6:0]), if (tmp[6:0] < 0); Ra <- MLm; Mn算术右移位Rb所示的位数。Rb在±63 内饱和，如是负值时，则算术左移位。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
asrp Mm,Ra,Mn,Rb	32bit	—	—
备注			

图449

asrp Mm,Rb,Mn,I6

操作	MHm:MLm <- MHn:MLn >> I6[5:0]; Rb <- MLm; Ra算术右移位当前值 (I6) 的位数。		
汇编·助记	格式	影响标记	受影响的标记
asrp Mm,Rb,Mn,I6	32bit	-	-
备注			
I6是无符号的值			

图450

asrp Mm,Rc,MHn,Ra,Rb

操作	tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:Ra >> abs(tmp[6:0]); MHm:MLm <- MHn:Ra << abs(tmp[6:0]), if (tmp[6:0] < 0); Rc <- MLm; MH: Ra算术右移位Rb所示的位数。Rb在±63内饱和，如是负值时，则算术左移位。		
汇编·助记	格式	影响标记	受影响的标记
asrp Mm,Rc,MHn,Ra,Rb	32bit	-	-
备注			

图451

asrp Mm,Rb,MHn,Ra,I6

操作	MHm:MLm <- MHn:Ra >> I6[5:0]; Rb <- MLm; MHn: Ra算术右移位当前值（I6）的位数。		
汇编·助记	格式	影响标记	受影响的标记
asrp Mm,Rb,MHn,Ra,I6	32bit	—	—
备注			
I6是无符号的值			

图452

asrpvw Mm,Ra,Mn,Rb

操作	<pre> tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); tmp2[126:0] <- sext127(MHn:MLn) >> abs(tmp[6:0]); tmp2[126:0] <- sext127(MHn:MLn) << abs(tmp[6:0]), if (tmp[6:0] < 0); MHm:MLm <- tmp2[63:0]; MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[126:0] > sext127(0x0_7fff_ffff[126:0]); MHm:MLm <- 0x0000_0000_8000_0000, CFR.OVS <- 1, if (tmp2[126:0] < sext127(0x1_8000_0000[126:0]); Ra <- MLm; Mn算术右移位Rb所示的位数。 结果不能用有符号32位数表示时则饱和。 Rb在±63内饱和，如是负值时，则算术左移位。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
asrpvw Mm,Ra,Mn,Rb	32bit	CFR.OVS	-
备注			

图453

lslp Mm,Ra,Mn,Rb

操作	=> aslp Mm,Ra,Mn,Rb MHm:MLm <- MHn:MLn << I6[5:0]; Rb <- MLm; Mn逻辑左移位Rb所示的位数。Rb在±63内饱和，如是负值时，则算术右移位。		
汇编·助记	格式	影响标记	受影响的标记
lslp Mm,Ra,Mn,Rb	相当于32bit	—	—
备注			
* 移动量为负值时，则算术右移位			

图454

lslp Mm,Rb,Mn,I6

操作	=> aslp Mm,Rb,Mn,I6 MHm:MLm <- MHn:MLn << I6[5:0]; Rb <- MLm; Mn逻辑左移位当前值（I6）的位数。		
汇编·助记	格式	影响标记	受影响的标记
lslp Mm,Rb,Mn,I6	相当于32bit	—	—
备注			
I6是无符号的值			

图455

lslp Mm,Rc,MHn,Ra,Rb

操作	=> aslp Mm,Rc,MHn,Ra,Rb		
	tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:Ra << abs(tmp[6:0]); MHm:MLm <- MHn:Ra >> abs(tmp[6:0]),if (tmp[6:0] < 0); Rc <- MLm; MHn:Ra逻辑左移位Rb所示的位数。Rb在±63内饱和，如是负值时，则算术右移位。		
汇编·助记	格式	影响标记	受影响的标记
lslp Mm,Rc,MHn,Ra,Rb	相当于32bit	—	—
备注			
* 移动量为负值时，则算术右移位			

图456

lslp Mm,Rb,MHn,Ra,I6

操作	=> aslp Mm,Rb,MHn,Ra,I6		
	MHm:MLm <- MHn:Ra << I6[5:0]; Rb <- MLm; MHn:Ra逻辑左移位当前值（I6）的位数。		
汇编·助记	格式	影响标记	受影响的标记
lslp Mm,Rb,MHn,Ra,I6	相当于32bit	—	—
备注			
I6是无符号的值			

图457

lsrp Mm,Ra,Mn,Rb

操作	<pre>tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:MLn >>> abs(tmp[6:0]); MHm:MLm <- MHn:MLn << abs(tmp[6:0]), if (tmp[6:0] < 0); Ra <- MLm;</pre> <p>Mn逻辑右移位Rb所示的位数。Rb在±63内饱和，如是负值时，则逻辑左移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
lsrp Mm,Ra,Mn,Rb	32bit	—	—
备注			

图458

lsrp Mm,Rb,Mn,I6

操作	<pre>MHm:MLm <- MHn:MLn >>> I6[5:0]; Rb <- MLm;</pre> <p>Mn逻辑右移位当前值（I6）的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
lsrp Mm,Rb,Mn,I6	32bit	—	—
备注			
I6是无符号的值			

图459

lsrp Mm,Rc,MHn,Ra,Rb

操作	<pre>tmp[6:0] <- Rb[6:0]; tmp[6:0] <- 63, if (Rb > 0x0000_003f); tmp[6:0] <- -63, if (Rb < 0xffff_ffc1); MHm:MLm <- MHn:Ra >>> abs(tmp[6:0]); MHm:MLm <- MHn:Ra << abs(tmp[6:0]), if (tmp[6:0] < 0); Rc <- MLm;</pre> <p>MHn:Ra逻辑右移位Rb所示的位数。Rb在±63内饱和，如是负值时，则逻辑左移位。</p>		
汇编·助记	格式	影响标记	受影响的标记
lsrp Mm,Rc,MHn,Ra,Rb	32bit	—	—
备注			

图460

lsrp Mm,Rb,MHn,Ra,I6

操作	<pre>MHm:MLm <- MHn:Ra >>> I6[5:0]; Rb <- MLm;</pre> <p>MHn:Ra逻辑右移位当前值（I6）的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
lsrp Mm,Rb,MHn,Ra,I6	32bit	—	—
备注			
I6是无符号的值			

图461

extr Rc,Ra,Rb

操作	<pre> if (Rb[12:8] >=(u) Rb[4:0]) { Rc <- sext(Ra[Rb[12:8]:Rb[4:0]],Rb[12:8]-Rb[4:0]+1); } else { Rc <- UD; } </pre> <p>用Rb指定位位置，取出Ra内容的一部分进行符号扩展，存储于Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
extr Rc,Ra,Rb	32bit	—	—
备注			
Rb[12:8],Rb[4:0]作为无符号的值处理。			

图462

extr Rb,Ra,Ib5,Ia5

操作	<pre> if (Ib5 >=(u) Ia5) { Rb <- sext(Ra[Ib5:Ia5],Ib5-Ia5+1); } else { Rb <- UD; } </pre> <p>用Ia5和Ib5指定位位置，取出Ra内容的一部分进行符号扩展，存储于Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
extr Rb,Ra,Ib5,Ia5	32bit	—	—
备注			
Ia5和Ib5是无符号的值。			

图463

ext Rb,Ra,I5

操作	<pre> => extr Rb,Ra,I5-1,0 If ((I5-1) >=(u) 0) { Rb <- sext(Ra[I5-1:0]); } else { Rb <- UD; } </pre> <p>符号扩展从Ra的0位到当前值（I5）位，存储于Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
ext Rb,Ra,I5		-	-
备注			
Ia5作为无符号的值处理。			

图464

exth Ra2

操作	<pre> Ra2 <- sext(Ra2[15:0]); </pre> <p>将Ra2的下位16位符号扩展，存储于Ra2。</p>		
汇编·助记	格式	影响标记	受影响的标记
exth Ra2	16bit	-	-
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图465

extb Ra2

操作	Ra2 <- sext(Ra2[7:0]); 将Ra2的下位8位符号扩展，存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
extb Ra2	16bit	-	-
备注			
可能使用的判定只是C7。（不能进行条件执行）			

图466

extru Rc,Ra,Rb

操作	<pre> if (Rb[12:8] >=(u) Rb[4:0]) { Rc <- uext(Ra[Rb[12:8]:Rb[4:0]],Rb[12:8]-Rb[4:0]+1); } else { Rc <- UD; } </pre> 用Rb指定位位置，取出Ra内容的一部分，不进行符号扩展，存储于Rc。		
汇编·助记	格式	影响标记	受影响的标记
extru Rc,Ra,Rb	32bit	-	-
备注			
Rb[12:8],Rb[4:0]作为无符号的值处理。			

图467

extru Rb,Ra,Ib5,Ia5

操作	<pre> if (Ib5 >=(u) Ia5) { Rb <- uext(Ra[Ib5:Ia5],Ib5-Ia5+1); } else { Rb <- UD; } </pre> <p>用Ia5和Ib5指定位置，取出Ra内容的一部分，不进行符号扩展，存储于Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
extru Rb,Ra,Ib5,Ia5	32bit	-	-
备注			
Ia5和Ib5为无符号的值。			

图468

extu Rb,Ra,I5

操作	<pre> => extru Rb,Ra,I5-1,0 if ((I5 -1) >=(u) 0) { Rb <- uext(Ra[I5-1:0]); } else { Rb <- UD; } </pre> <p>从Ra的0位到当前值（I5）位，不进行符号扩展存储于Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
extu Rb,Ra,I5		-	-
备注			
I5为无符号的值。			

图469

exthi Ra2

操作	Ra2 <- uext(Ra2[15:0]); Ra2的下位16位不进行符号扩展存储于Ra2。		
汇编·助记	格式	影响标记	受影响的标记
exthi Ra2	16bit	-	-
备注			
可能使用的判定，只是C7。（不能进行条件执行）			

图470

extbu Ra2

操作	Ra2 <- uext(Ra2[7:0]); 从Ra的0到定值（15）比特符号扩展存储于Ra。		
汇编·助记	格式	影响标记	受影响的标记
extbu Ra2	16bit	-	-
备注			
可能使用的判定，只是CT（不能进行条件执行）			

图471

mskgen Rc,Rb

操作	<pre> if (Rb[12:8] >=(u) Rb[4:0]) { Rc<-一位位置Rb[4:0]~Rb[12:8] 为1, 其他 :0 } else { Re<-一位位置0~Rb[12:8],Rb[4:0]~31 为1 其他 :0 } </pre> <p>用Rb指定位位置生成有连续1的值的屏蔽数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
mskgen Rc,Rb	32bit	—	—
备注			
Rb[12:8],Rb[4:0] 为无符号的值。			

图472

mskgen Rb,Ib5,Ia5

操作	<pre> if (Ib5 >=(u) Ia5) { Rc<-一位位置Ia5~Ib5 为1, 其他 :0 } else { Re<-一位位置:0~Ib5,Ia5~31 为1 其他 :0 } </pre> <p>用Ia5和Ib5指定位位置生成有连续1的值的屏蔽数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
mskgen Rb,Ib5,Ia5	32bit	—	—
备注			
Ia5和Ib5为无符号的值。			

图473

msk Rc,Ra,Rb

动作	<pre> if (Rb[12:8] >=(u) Rb[4:0]) { Rc<一位位置:Rb[4:0]~Rb[12:8] 为Ra的内容 :0 } else { Re<一位位置0~Rb[4:0],Rb[12:8]~31 为Ra之内 其他:0 } </pre> <p>用Rb指定位位置，屏蔽Ra的内容。</p>		
汇编·助记	格式	影响标记	受影响的标记
msk Rc,Ra,Rb	32bit	—	—
备注			
Rb[12:8],Rb[4:0] 作为无符号的值处理。			

图474

msk Rb,Ra,Ib5,Ia5

操作	<pre> if (Ib5 >=(u) Ia5) { Rc<一位位置: 到Ia5~Ib5的Ra的内容。其他: 0 } else { Re<一位位置: 0~Ib5,Ia5~31为Ra的内容。其他: 0 } </pre> <p>用Ia5和Ib5指定位位置，屏蔽Ra的内容。</p>		
汇编·助记	格式	影响标记	受影响的标记
msk Rb,Ra,Ib5,Ia5	32bit	—	—
备注			
Ia和Ib为无符号的值。			

图475

satw Mm,Rb,Mn

操作	MHm:MLm <- MHn:MLn; MHm:MLm <- 0x7fffffff[31: 0], if (MHn:MLn > 0x00000000_7fffffff[63:0]); MHm:MLm <- 0x80000000[31: 0], if (MHn:MLn < 0xffffffff_80000000[63:0]); Rb <- MLm; Mn用有符号的32位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
satw Mm,Rb,Mn	32bit	—	—
备注			

图476

sath Rb,Ra

操作	Rb <- Ra; Rb <- 0x00007fff[31: 0], if (Ra[31:0] > 0x00007fff[31:0]); Rb <- 0xffff8000[31: 0], if (Ra[31:0] < 0xffff8000[31:0]); Ra用有符号的16位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
sath Rb,Ra	32bit	—	—
备注			

图477

sat12 Rb,Ra

操作	Rb <- Ra; Rb <- 0x000007ff[31: 0], if (Ra[31:0] > 0x000007ff[31:0]); Rb <- 0xffff800[31: 0], if (Ra[31:0] < 0xffff800[31:0]); Ra用有符号的12位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
sat12 Rb,Ra	32bit	—	—
备注			

图478

sat9 Rb,Ra

操作	Rb <- Ra; Rb <- 0x000000ff[31: 0], if (Ra[31:0] > 0x000000ff[31:0]); Rb <- 0xffff00[31: 0], if (Ra[31:0] < 0xffff00[31:0]); Ra用有符号的9位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
sat9 Rb,Ra	32bit	—	—
备注			

图479

satb Rb,Ra

操作	Rb <- Ra; Rb <- 0x0000007f[31: 0], if (Ra[31:0] > 0x0000007f[31:0]); Rb <- 0xfffff80[31: 0], if (Ra[31:0] < 0xfffff80[31:0]); Ra用有符号的8位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
satb Rb,Ra	32bit	-	-
备注			

图480

satbu Rb,Ra

操作	Rb <- Ra; Rb <- 0x000000ff[31: 0], if (Ra[31:0] > 0x000000ff[31:0]); Rb <- 0x00000000[31: 0], if (Ra[31:0] < 0x00000000[31:0]); Ra用无符号的8位值饱和，存储于Rb。		
汇编·助记	格式	影响标记	受影响的标记
satbu Rb,Ra	32bit	-	-
备注			

图481

extw Mm,Rb,Ra

操作	MHm:MLm <- 0x0000_0000:Ra, if(Ra[31] == 0); MHm:MLm <- 0xffff_ffff:Ra, if(Ra[31] == 1); Rb <- MLm; 将Ra符号扩展存储于Mm。		
汇编·助记	格式	影响标记	受影响的标记
extw Mm,Rb,Ra	32bit	—	—
备注			

图482

vintlh Rc,Ra,Rb

操作	Rc[31:16] <- Ra[15: 0]; Rc[15: 0] <- Rb[15: 0]; 各寄存器以半字矢量的形式处理。 Ra的下位16位和Rb的下位16位压缩		
汇编·助记	格式	影响标记	受影响的标记
vintlh Rc,Ra,Rb	32bit	—	—
备注			

图483

vintlhh Rc,Ra,Rb

操作	Rc[31:16] <- Ra[31:16]; Rc[15: 0] <- Rb[31:16]; 各寄存器以半字矢量的形式处理。 Ra的上位16位和Rb的上位16位压缩		
汇编·助记	格式	影响标记	受影响的标记
vintlhh Rc,Ra,Rb	32bit	—	—
备注			

图484

vintlhb Rc,Ra,Rb

操作	Rc[31:24] <- Ra[15: 8]; Rc[23:16] <- Rb[15: 8]; Rc[15: 8] <- Ra[7: 0]; Rc[7: 0] <- Rb[7: 0]; 各寄存器以字节矢量的形式处理。 Ra的下位16位和Rb的下位16位压缩		
汇编·助记	格式	影响标记	受影响的标记
vintlhb Rc,Ra,Rb	32bit	—	—
备注			

图485

vintlhb Rc,Ra,Rb

操作	Rc[31:24] <- Ra[31:24]; Rc[23:16] <- Rb[31:24]; Rc[15: 8] <- Ra[23:16]; Rc[7: 0] <- Rb[23:16]; 各寄存器以字节矢量的形式处理。 Ra的上位16位和Rb的上位16位压缩		
汇编·助记	格式	影响标记	受影响的标记
vintlhb Rc,Ra,Rb	32bit	-	-
备注			

图486

valn Rc,Ra,Rb

操作	tmp[87:0] <- Ra:Rb << (CFR.ALN[1:0] << 3); Rc <- tmp[63:32]; 根据CFR.ALN进行字节调整。		
汇编·助记	格式	影响标记	受影响的标记
valn Rc,Ra,Rb	32bit	-	CFR.ALN
备注			

图487

valn1 Rc,Ra,Rb

操作	$Rc[31:8] \leftarrow Ra[23:0];$ $Rc[7:0] \leftarrow Rb[31:24];$ 字节调整		
汇编·助记	格式	影响标记	受影响的标记
valn1 Rc,Ra,Rb	32bit	-	-
备注			

图488

valn2 Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[15:0];$ $Rc[15:0] \leftarrow Rb[31:16];$ 字节调整		
汇编·助记	格式	影响标记	受影响的标记
valn2 Rc,Ra,Rb	32bit	-	-
备注			

图 489

valn3 Rc,Ra,Rb

操作	Rc[31:24] <- Ra[7: 0]; Rc[23: 0] <- Rb[31: 8]; 字节调整		
汇编·助记	格式	影响标记	受影响的标记
valn3 Rc,Ra,Rb	32bit	—	—
备注			

图 490

valnc1 Rc,Ra,Rb

操作	tmp [87: 0] <- Ra:Rb << (CFR.ALN[1:0] << 3); tmp2[31: 0] <- tmp[63:32]; Rc <- tmp2[31:24]:tmp2[31:24]:tmp2[23:16]:tmp2[23:16], if (VC0 == 0); Rc <- tmp2[31:24]:tmp2[23:16]:tmp2[23:16]:tmp2[15: 8], if (VC0 == 1); 在图像处理的动态补偿中使用。 根据CFR.ALN进行字节调整，根据VC0的值变更取出的字节数据。		
汇编·助记	格式	影响标记	受影响的标记
valnc1 Rc,Ra,Rb	32bit	—	CFR.ALN,VC0
备注			

图491

valnvc2 Rc,Ra,Rb

操作	<pre>tmp [87: 0] <- Ra:Rb << (CFR.ALN[1:0] << 3); tmp2[31: 0] <- tmp[47:16]; Rc <- tmp2[31:24]:tmp2[31:24]:tmp2[23:16]:tmp2[23:16], if (VC0 == 0); Rc <- tmp2[31:24]:tmp2[23:16]:tmp2[23:16]:tmp2[15: 8], if (VC0 == 1);</pre> <p>在图像处理的动态补偿中使用。</p> <p>根据CFR.ALN进行字节调整，根据VC0的值变更取出的字节数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
valnvc2 Rc,Ra,Rb	32bit	—	CFR.ALN,VC0
备注			

图492

valnvc3 Rc,Ra,Rb

操作	<pre>tmp [87: 0] <- Ra:Rb << (CFR.ALN[1:0] << 3); tmp2[31: 0] <- tmp[63:32]; Rc <- tmp2[31:24]:tmp2[31:24]:tmp2[23:16]:tmp2[23:16], if (VC0 == 0); Rc <- tmp2[31:24]:tmp2[15: 8]:tmp2[23:16]:tmp2[7: 0], if (VC0 == 1);</pre> <p>在图像处理的动态补偿中使用。</p> <p>根据CFR.ALN进行字节调整，根据VC0的值变更取出的字节数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
valnvc3 Rc,Ra,Rb	32bit	—	CFR.ALN,VC0
备注			

图493

valnc4 Rc,Ra,Rb

操作	<pre>tmp [87: 0] <- Ra:Rb << (CFR.ALN[1:0] << 3); tmp2[31: 0] <- tmp[47:16]; Rc <- tmp2[31:24]:tmp2[31:24]:tmp2[23:16]:tmp2[23:16], if (VC0 == 0); Rc <- tmp2[31:24]:tmp2[15: 8]:tmp2[23:16]:tmp2[7: 0], if (VC0 == 1);</pre> <p>在图像处理的动态补偿中使用。 根据CFR.ALN的值变更取出的字节数据。</p>		
汇编·助记	格式	影响标记	受影响的标记
valnc4 Rc,Ra,Rb	32bit	-	CFR.ALN,VC0
备注			
CFR.ALN[1:0]=3 时 tmp2[7: 0]为0x00			

图494

vxchngh Rb,Ra

操作	<pre>=> valn2 Rb,Ra,Ra Rb[31:16] <- Ra[15: 0]; Rb[15: 0] <- Ra[31:16];</pre> <p>各寄存器以半字向量的形式处理。 替换Ra的上位16比特和下位16比特，存储于Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxchngh Rb,Ra	相当于32bit	-	-
备注			

图495

byterev Rb,Ra

操作	$Rb[31:24] \leftarrow Ra[7:0];$ $Rb[23:16] \leftarrow Ra[15:8];$ $Rb[15:8] \leftarrow Ra[23:16];$ $Rb[7:0] \leftarrow Ra[31:24];$ 各寄存器以字节矢量的形式处理 反转字节指令		
汇程·助记	格式	影响标记	受影响的标记
byterev Rb,Ra	32bit	-	-
备注			

图496

vstovb Rb,Ra

操作	$Rb[31:24] \leftarrow Ra[7:0];$ $Rb[23:16] \leftarrow Ra[7:0];$ $Rb[15:8] \leftarrow Ra[7:0];$ $Rb[7:0] \leftarrow Ra[7:0];$ 各寄存器以字节矢量的形式处理 将Ra的下位8比特转送至Rb。		
编程·助记	幅度	影响标记	受影响的标记
vstovb Rb,Ra	32bit	-	-
备注			

图497

vstovh Rb,Ra

操作	Rb[31:16] <- Ra[15:0]; Rb[15: 0] <- Ra[15:0]; 各寄存器以半字矢量形式处理 将Ra下位的16比特转送至Rb		
汇编·助记	格式	影响标记	受影响的标记
vstovh Rb,Ra	32bit	-	-
备注			

图498

vlunpkh Rb:Rb+1,Ra

操作	Rb <- sext(Ra[31:16]); Rb+1 <- sext(Ra[15: 0]); Ra以半字矢量形式处理 将Ra在Rb和Rb+1的下位扩展符号解除压缩		
汇编·助记	格式	影响标记	受影响的标记
vlunpkh Rb:Rb+1,Ra	32bit	-	-
备注			
请将Rb设为偶数编号			

图499

vlunpkhu Rb:Rb+1,Ra

操作	Rb <- uext(Ra[31:16]); Rb+1 <- uext(Ra[15:0]); Ra以半字矢量形式处理。 将Ra在Rb和Rb+1的下位不扩展符号，解除压缩		
汇编·助记	格式	影响标记	受影响的标记
vlunpkhu Rb:Rb+1,Ra	32bit	-	-
备注			
请将Rb设为偶数编号			

图500

vlunpkb Rb:Rb+1,Ra

操作	Rb[31:16] <- sext16(Ra[31:24]); Rb[15:0] <- sext16(Ra[23:16]); Rb+1[31:16] <- sext16(Ra[15:8]); Rb+1[15:0] <- sext16(Ra[7:0]); Ra以字节矢量形式，Rb和Rb+1以半字矢量形式处理 将Ra在Rb和Rb+1的下位扩展符号，解除压缩		
汇编·助记	格式	影响标记	受影响的标记
vlunpkb Rb:Rb+1,Ra	32bit	-	-
备注			
请将Rb设为偶数编号			

图501

vlunpkbu Rb:Rb+1,Ra

操作	Rb[31:16] <- uext16(Ra[31:24]); Rb[15:0] <- uext16(Ra[23:16]); Rb+1[31:16] <- uext16(Ra[15:8]); Rb+1[15:0] <- uext16(Ra[7:0]); Ra以字节矢量形式，Rb和Rb+1以半字矢量形式处理 将Ra在Rb和Rb+1的下位不扩展符号，解除压缩		
汇编·助记	格式	影响标记	受影响的标记
vlunpkbu Rb:Rb+1,Ra	32bit	—	—
备注			
请将Rb设为偶数编号			

图502

vhunpkh Rb:Rb+1,Ra

操作	Rb <- Ra[31:16]:16'b0; Rb+1 <- Ra[15:0]:16'b0; Ra以半字矢量形式处理。 将Ra在Rb和Rb+1的上位解除压缩		
汇编·助记	格式	影响标记	受影响的标记
vhunpkh Rb:Rb+1,Ra	32bit	—	—
备注			
请将Rb设为偶数编号			

图503

vhunpkb Rb:Rb+1,Ra

操作	$Rb[31:16] \leftarrow Ra[31:24]:8'b0;$ $Rb[15:0] \leftarrow Ra[23:16]:8'b0;$ $Rb+1[31:16] \leftarrow Ra[15:8]:8'b0;$ $Rb+1[15:0] \leftarrow Ra[7:0]:8'b0;$		
	Ra以字节矢量形式，Rb和Rb+1以半字矢量形式处理 将Ra在Rb和Rb+1的上位解除压缩。		
汇编·助记	格式	影响标记	受影响的标记
vhunpkb Rb:Rb+1,Ra	32bit	—	—
备注			

图504

vunpk1 Rb,Mn

操作	$Rb \leftarrow MHn[31:16]:MLn[31:16]$		
	Mn以字矢量形式处理 MHn、MLn寄存器的上位16位，连结并写入Rb。		
汇编·助记	格式	影响标记	受影响的标记
vunpk1 Rb,Mn	32bit	—	—
备注			

图 505

vunpk2 Rb,Mn

操作	$Rb \leftarrow MHn[15:0]:MLn[15:0]$ Mn以字矢量形式处理 MHn、MLn寄存器的上位16位，连结并写入Rb。		
汇编·助记	格式	影响标记	受影响的标记
vunpk2 Rb,Mn	32bit	—	—
备注			

图 506

vipkh Rc,Rb,Ra

操作	$Rc[31:16] \leftarrow Rb[15:0];$ $Rc[31:16] \leftarrow 0x7fff[15:0], \text{ if } (Rb[31:0] > 0x00007fff[31:0]);$ $Rc[31:16] \leftarrow 0x8000[15:0], \text{ if } (Rb[31:0] < 0xffff8000[31:0]);$ $Rc[15:0] \leftarrow Ra[15:0];$ $Rc[15:0] \leftarrow 0x7fff[15:0], \text{ if } (Ra[31:0] > 0x00007fff[31:0]);$ $Rc[15:0] \leftarrow 0x8000[15:0], \text{ if } (Ra[31:0] < 0xffff8000[31:0]);$ Rc以半字矢量形式处理 将Ra和Rb的下位压缩并存储到Rc。Ra和Rb不能带符号16位表示时饱和的压缩。		
汇编·助记	格式	影响标记	受影响的标记
vipkh Rc,Rb,Ra	32bit	—	—
备注			

图 507

vlpkhu Rc,Rb,Ra

操作	<p> $Rc[31:16] \leftarrow Rb[15:0];$ $Rc[31:16] \leftarrow 0xffff[15:0], \text{ if } (Rb[31:0] > 0x0000ffff[31:0]);$ $Rc[31:16] \leftarrow 0x0000[15:0], \text{ if } (Rb[31:0] < 0x00000000[31:0]);$ $Rc[15:0] \leftarrow Ra[15:0];$ $Rc[15:0] \leftarrow 0xffff[15:0], \text{ if } (Ra[31:0] > 0x0000ffff[31:0]);$ $Rc[15:0] \leftarrow 0x0000[15:0], \text{ if } (Ra[31:0] < 0x00000000[31:0]);$ </p> <p>Rc以半字矢量形式处理</p> <p>将Ra和Rb的下位压缩并存储到Rc。Ra和Rb不能用带符号16位表示时饱和后压缩。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlpkhu Rc,Rb,Ra	32bit	-	-
备注			

图508

vipkb Rc,Rb,Ra

操作	<pre> Rc[31:24] <- Rb[23:16]; Rc[31:24] <- 0x7f[7: 0], if (Rb[31:16] > 0x007f[15:0]); Rc[31:24] <- 0x80[7: 0], if (Rb[31:16] < 0xff80[15:0]); Rc[23:16] <- Rb[7: 0]; Rc[23:16] <- 0x7f[7: 0], if (Rb[15: 0] > 0x007f[15:0]); Rc[23:16] <- 0x80[7: 0], if (Rb[15: 0] < 0xff80[15:0]); Rc[15: 8] <- Ra[23:16]; Rc[15: 8] <- 0x7f[7: 0], if (Ra[31:16] > 0x007f[15:0]); Rc[15: 8] <- 0x80[7: 0], if (Ra[31:16] < 0xff80[15:0]); Rc[7: 0] <- Ra[7: 0]; Rc[7: 0] <- 0x7f[7: 0], if (Ra[15: 0] > 0x007f[15:0]); Rc[7: 0] <- 0x80[7: 0], if (Ra[15: 0] < 0xff80[15:0]); Rc和Rb以字节矢量形式处理，Rc以半字矢量形式处理。 将Ra和Rb的下位压缩并存储到Rc。Ra和Rb不能用带 符号8位表示时饱和后压缩。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vipkb Rc,Rb,Ra	32bit	—	—
备注			

图509

vlpkbu Rc,Rb,Ra

操作	<pre> Rc[31:24] <- Rb[23:16]; Rc[31:24] <- 0xff[7: 0], if (Rb[31:16] > 0x00ff[15:0]); Rc[31:24] <- 0x00[7: 0], if (Rb[31:16] < 0x0000[15:0]); Rc[23:16] <- Rb[7: 0]; Rc[23:16] <- 0xff[7: 0], if (Rb[15: 0] > 0x00ff[15:0]); Rc[23:16] <- 0x00[7: 0], if (Rb[15: 0] < 0x0000[15:0]); Rc[15: 8] <- Ra[23:16]; Rc[15: 8] <- 0xff[7: 0], if (Ra[31:16] > 0x00ff[15:0]); Rc[15: 8] <- 0x00[7: 0], if (Ra[31:16] < 0x0000[15:0]); Rc[7: 0] <- Ra[7: 0]; Rc[7: 0] <- 0xff[7: 0], if (Ra[15: 0] > 0x00ff[15:0]); Rc[7: 0] <- 0x00[7: 0], if (Ra[15: 0] < 0x0000[15:0]); </pre> <p>Ra和Rb以字节矢量形式处理，Rc以半字矢量形式处理 将Ra和Rb的下位压缩并存储到Rc。Ra和Rb不能用带 符号8位表示时，饱和后压缩。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlpkbu Rc,Rb,Ra	32bit	—	—
备注			

图510

vhpkh Rc,Ra,Rb

操作	=> vintlhh Rc,Ra,Rb		
	Rc[31:16] <- Ra[31:16]; Rc[15: 0] <- Rb[31:16]; 各寄存器以半字形式处理。 将Ra的下位16位和Rb的上位16位打包。		
汇编·助记	格式	影响标记	受影响的标记
vhpkh Rc,Ra,Rb	相当于32bit	—	—
备注			

图511

vhpkb Rc,Ra,Rb

操作	Rc[31:24] <- Ra[31:24]; Rc[23:16] <- Ra[15: 8]; Rc[15: 8] <- Rb[31:24]; Rc[7: 0] <- Rb[15: 8]; Ra和Rb以字节形式，Rc以半字矢量形式处理 将Ra和Rb的上位，压缩并存储到Rc。		
	汇编·助记	格式	影响标记
vhpkb Rc,Ra,Rb	32bit	—	—
备注			

图512

vexth Mm,Rb,Ra

操作	MHm[31:16]:MLm[31:16] <- 0xffff:Ra[31:16], if (Ra[31] == 1); MHm[31:16]:MLm[31:16] <- 0x0000:Ra[31:16], if (Ra[31] == 0); MHm[15:0]:MLm[15:0] <- 0xffff:Ra[15:0], if (Ra[15] == 1); MHm[15:0]:MLm[15:0] <- 0x0000:Ra[15:0], if (Ra[15] == 0); Rb <- MLm; Ra、Rb以半字矢量形式，Mm以字矢量形式处理。 将Ra扩展符号存储到Mm。		
汇编·助记	格式	影响标记	受影响的标记
vexth Mm,Rb,Ra	32bit	-	-
备注			

图513

bseq0 Rb,Ra

操作	Rb <- bseq(Ra,0); 计算从Ra的MSB开始连续的0的个数，存储到Rb。		
汇编·助记	格式	影响标记	受影响的标记
bseq0 Rb,Ra	32bit	-	-
备注			

图514

bseq1 Rb,Ra

操作	Rb <- bseq(Ra,1); 计算从Ra的MSB开始的连续的1的个数，存储到Rb		
汇编·助记	格式	影响标记	受影响的标记
bseq1 Rb,Ra	32bit	-	-
备注			

图515

bseq Rb,Ra

操作	Rb <- bseq(Ra[30:0]:~Ra[31],Ra[31]); Rb <- 0, if (Ra == 0); 计算从Ra的MSB的1位以下开始连续的符号位数，并存储到Rb，Ra为0时，输出“0”。		
汇编·助记	格式	影响标记	受影响的标记
bseq Rb,Ra	32bit	-	-
备注			
请注意统计是排除了MSB的。			

图516

bcnt1 Rb,Ra

操作	Rb <- bcnt(Ra,1); 计算Ra的1的个数、存储到Rb		
汇编·助记	格式	影响标记	受影响的标记
bcnt1 Rb,Ra	32bit	-	-
备注			

图517

rndvh Rb,Ra

操作	tmp[32:0] <- sext33(Ra) + 0x0_0000_8000[32:0]; Rb <- sext(tmp[31:16]); Rb <- 0x0000_7fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff); 进行32位值到16位值的四舍五入计算，结果 不能用带符号的16位表示时饱和		
汇编·助记	格式	影响标记	受影响的标记
rndvh Rb,Ra	32bit	CFR.OVS	-
备注			
不会发生负的溢出。			

图518

mskbrvb Rc,Ra,Rb

操作	<pre> for (i = 0; i < 8; i++) { tmp[7-i] <- Rb[i]; } tmp[31:8] <- 0; if (CFR.BPO <(u) 31[4:0]) { Rc[31:CFR.BPO+1] <- Ra[31:CFR.BPO+1]; Rc[CFR.BPO:0] <- tmp[CFR.BPO:0]; } else { Rc[31:0] <- tmp[31:0]; } </pre> <p>位逆向定址使用。 将Rb的下位8位逆向，在CFR、BPO显示位的位置 连结Ra和Rb</p>		
汇编·助记	格式	影响标记	受影响的标记
mskbrvb Rc,Ra,Rb	32bit	—	CFR.BPO
备注			

图519

mskbrvh Rc,Ra,Rb

操作	<pre> for (i = 0; i < 16; i++) { tmp[15-i] <- Rb[i]; } tmp[31:16] <- 0; if (CFR.BPO <(u) 31[4:0]) { Rc[31:CFR.BPO+1] <- Ra[31:CFR.BPO+1]; Rc[CFR.BPO:0] <- tmp[CFR.BPO:0]; } else { Rc[31:0] <- tmp[31:0]; } </pre> <p>位逆向定址使用。 将Rb的下位16位逆向，在CFR、BPO显示位的位置 连结Ra和Rb</p>		
汇编·助记	格式	影响标记	受影响的标记
mskbrvh Rc,Ra,Rb	32bit	—	CFR.BPO
备注			

图520

movp Rc:Rc+1,Ra,Rb

操作	Rc <- Ra; Rc+1 <- Rb; 将Ra转送到Rc, Rb转送到Rc+1		
汇编·助记	格式	影响标记	受影响的标记
movp Rc:Rc+1,Ra,Rb	32bit	—	—
备注			
请将Rc设为偶数编号。			

图521

hmul Mm,Rc,Ra,Rb

操作	MHm:MLm <- smul(Ra,sext(Rb[31:16])); Rc <- MLm[31:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Rb以半字矢量形式处理。 Ra和Rb的上位16位以整数相乘, 结果存储到Mm和Rc		
汇编·助记	格式	影响标记	受影响的标记
hmul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图522

lmul Mm,Rc,Ra,Rb

操作	MHm:MLm <- smul(Ra, sext(Rb[15:0])); Rc <- MLm[31:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Rb以半字矢量形式处理。 将Ra和Rb的下位16位以整数相乘，将结果存储到Mm和Rc		
汇编·助记	格式	影响标记	受影响的标记
lmul Mm,Rc,Ra,Rb	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc			

图523

fmulhh Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm:MLm <- sext64(tmp[31:0]); MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra、Rb、Rc以16位值处理，Mm以32位值处理。 Ra和Rb以固定小数点相乘，结果存储到Mm 和Rc。结果不能用带符号的32位表示时为 饱和 </pre>		
汇编·助记	格式	影响标记	受影响的标记
fmulhh Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc			

图524

fmulhr Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm:MLm <- sext64(tmp2[31:16]:16'b0); MHm:MLm <- 0x0000_0000_7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm:MLm <- UD, if (PSR.FXP == 1); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以16位值处理，Mm以32位值处理。 Ra和Rb以固定小数点相乘，相乘后，从32位向16位四舍五入，存储到Mm和Rc。结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmulhr Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1时，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc			

图525

fmulhw Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm:MLm <- sext64(tmp[31:0]); MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000); Rc <- MLm[31:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rc以32位值处理，Mm,Rc以32位值处理。 Ra和Rb以固定小数点相乘，结果存储到Mm和Rc 结果不能用带符号32位显示时为饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
fmulhw Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图526

fmulhww Mm,Rc,Ra,Rb

操作	<pre> tmp[48:0] <- sext49(smul(Ra[31:0],Rb[15:0]))<<FXP_SHIFT; tmp[48:0] <- 0x0_7fff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[48:0] <- 0x0_ffff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm:MLm <- sext64(tmp[47:0]); MHm:MLm <- 0x0000_7fff_ffff_ffff, CFR.OVS <- 1, if (tmp[48:0] > 0x0_7fff_ffff_ffff[48:0]; MHm:MLm <- 0xffff_8000_0000_0000, CFR.OVS <- 1, if (tmp[48:0] < 0x1_8000_0000_0000[48:0]; Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rc以32位值处理，Rb以16位值处理， Mm以48位值处理。 Ra和Rb以固定小数点相乘，结果存储到Mm和Rc 结果不能用带符号的48位显示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
fmulhww Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图527

mul Mm,Rc,Ra,Rb

操作	MHm:MLm <- smul(Ra,Rb); Rc <- MLm[31:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rb整数相乘，结果存储到Mm和Rc		
汇编·助记	格式	影响标记	受影响的标记
mul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图528

mul Mm,Rb,Ra,I8

操作	MHm:MLm <- smul(Ra,sext(I8)); Rb <- MLm, if (reg#(Rb) != 30 && reg#(Rb) != 31); Ra和当前值 (I8) 带符号相乘。		
汇编·助记	格式	影响标记	受影响的标记
mul Mm,Rb,Ra,I8	32bit	—	—
备注			
I8为带符号的值。 Rb指定为r30或r31时，无法写入Rb。			

图 529

mulu Mm,Rc,Ra,Rb

操作	MHm:MLm <- umul(Ra,Rb); Rc <- MLm, if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rb不带符号整数相乘，结果存储到Mm和Rc		
汇编·助记	格式	影响标记	受影响的标记
mulu Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 530

mulu Mm,Rb,Ra,I8

操作	MHm:MLm <- umul(Ra,uxxt(I8)); Rb <- MLm, if (reg#(Rb) != 30 && reg#(Rb) != 31); Ra和当前值（I8）不带符号相乘。		
汇编·助记	格式	影响标记	受影响的标记
mulu Mm,Rb,Ra,I8	32bit	—	—
备注			
I8为不带符号的值。			
Rb指定为r30或r31时，不能写入Rb。			

图531

fmulww Mm,Rc,Ra,Rb

操作	<pre> tmp[64:0] <- sext65(smul(Ra,Rb)) << FXP_SHIFT; tmp[64:0] <- 0x0_7fff_ffff_ffff, CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 0)); tmp[64:0] <- 0x0_ffff_ffff_ffff, CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 1)); MHm:MLm <- tmp[63:0]; MHm:MLm <- 0x7fff_ffff_ffff_ffff, CFR.OVS <- 1, if (tmp[64:0] > 0x0_7fff_ffff_ffff[64:0]); MHm:MLm <- 0x8000_0000_0000_0000, CFR.OVS <- 1, if (tmp[64:0] < 0x1_8000_0000_0000[64:0]); Rc <- MHm[31:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb,Rc以32位值处理, Mm以64位值处理。 Ra和Rb以固定小数点相乘, 结果存储到Mm和Rc, 结果不能表示为带符号的64位时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmulww Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图532

hmac Mm,Rc,Ra,Rb,Mn

操作	$\text{MHm:MLm} \leftarrow \text{smul}(\text{Ra}, \text{sext}(\text{Rb}[31:16])) + \text{MHn:MLn};$ $\text{Rc} \leftarrow \text{MLm}, \text{ if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$ <p>Rb以半字矢量形式处理 Ra和Rb的上位16位整数相乘，与Mm相加，结果存储到Mm和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
hmac Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图533

hmac M0,Rc,Ra,Rb,Rx

操作	$\text{MH0:ML0} \leftarrow \text{sext64}(\text{Rx}) + \text{smul}(\text{Ra}, \text{sext}(\text{Rb}[31:16]));$ $\text{Rc} \leftarrow \text{ML0}, \text{ if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$ <p>Rb以半字矢量形式处理 Ra和Rb的上位16位整数相乘，与Rx相加，结果存储到M0和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
hmac M0,Rc,Ra,Rb,Rx	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rb。			

图 534

lmac Mm,Rc,Ra,Rb,Mn

操作	$\text{MHm:MLm} \leftarrow \text{smul}(\text{Ra}, \text{sext}(\text{Rb}[15:0])) + \text{MHn:MLn};$ $\text{Rc} \leftarrow \text{MLm}, \text{ if } (\text{reg\#}(\text{Rc}) \neq 30 \ \&\& \ \text{reg\#}(\text{Rc}) \neq 31);$ <p>Rb以半字矢量形式处理 Ra和Rb的下位16位整数相乘，与Mm相加，结果 存储到Mm和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
lmac Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 535

lmac M0,Rc,Ra,Rb,Rx

操作	$\text{MH0:ML0} \leftarrow \text{sext64}(\text{Rx}) + \text{smul}(\text{Ra}, \text{sext}(\text{Rb}[15:0]));$ $\text{Rc} \leftarrow \text{ML0}, \text{ if } (\text{reg\#}(\text{Rc}) \neq 30 \ \&\& \ \text{reg\#}(\text{Rc}) \neq 31);$ <p>Rb以半字矢量形式处理 Ra和Rb的下位16位整数相乘，与Rx相加，结果 存储到M0和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
lmac M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rb。			

图 536

fmachh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && PSR.FXP == 1)); tmp2[33:0] <- MHn[1:0]:MLn + sext34(tmp[31:0]); MHm:MLm <- sext164(tmp2[31:0]); MHm:MLm <- 0x0000_0000_7fff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_fff[33:0]); MHm:MLm <- 0xffff_fff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra, Rb,Rc以16位值处理, Mm,Mn以32位值处理。 Ra和Rb以固定小数点相乘, 与Mn相加, 结果存储 到Mm和Rc, 结果不能用带符号的16位表示时饱和 </pre>		
汇编·助记	格式	影响标记	受影响的标记
fmachh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rb。			

图537

fmachh M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(umul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) + sext34(tmp[31:0]) MH0:ML0 <- sext64(tmp2[31:0]); MH0:ML0 <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0:ML0 <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MH0[15:0]:ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra, Rb,Rc以16位值处理, M0,Rx以32位值处理。 Ra和Rb以固定小数点相乘, 与Rx相加, 结果存储 到M0和Rc, 结果不能用带符号的32位表示时饱和 </pre>		
汇编助记	格式	影响标记	受影响的标记
fmachh M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rb。			

图 538

fmachr Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(umul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && PSR.FXP == 1)); tmp2[34:0] <- MHm[2:0]:MLn + sext35(tmp[31:0]) + sext35(0x0_8000); MHm:MLn = sext64(tmp2[31:16]:16'b0); MHm:MLn <- 0x0000_0000_7fff_0000, CFR.OVS <- 1, If (tmp2[34:16] > 0x0_7fff[18:0]); MHm:MLn <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, If (tmp2[34:16] < 0x7_8000[18:0]); MHm:MLn = UD, If (PSR.FXP == 1); Rc <- MHm[15:0]:MLm[31:16], If (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以16位值处理, Mm, Mn以32位值处理。 Ra和Rb以固定小数点相乘、与Mn相加。相加后从32位到16位四舍五入, 存储到Mm和Rc, 结果不能用带符号的16位表示时为饱和</p>		
编程·助记	幅度	影响标记	受影响的标记
fmachr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话, 未组装Mm, Rc变为UD。 Mm的下位16位, 经常变为0。 Rc指定为r30或r31时, 不能写入Rc			

图539

fmachhr M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<< FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- (sext35(Rx[15:0])<<16) + sext35(tmp[32:0]) + sext35(0x0_8000); MH0:ML0 <- sext64(tmp2[31:16]:16'b0); MH0:ML0 <- 0x0000_0000_7fff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7fff[18:0]); MH0:ML0 <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MH0:ML0 <- UD, if (PSR.FXP == 1); Rc <- MH0[15:0]:ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以16位值处理, M0, Rx以32位值处理。 Ra和Rb以固定小数点相乘、与Rx相加。相加后从32位到16位四舍五入, 存储到M0和Rc, 结果不能用带符号的16位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmachhr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话, 未组装Mm, Rc变为UD。 Mm的下位16位, 经常变为0。 Rc指定为r30或r31时, 不能写入Rc			

图 540

fmachw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- MHn[1:0]:MLn + sext34(tmp[32:0]); MHm:MLm <- sext64(tmp2[31:0]); MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MLm, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rb以16位值处理，Mm，Mn，Rc以32位值处理。 Ra和Rb以固定小数点相乘、与Mn相加。结果存储到 Mm与Rc，结果不能用带符号的32位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmachw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图541

fmachw M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(Rx) + sext34(tmp[32:0]); MH0:ML0 <- sext64(tmp[31:0]); MH0:ML0 <- 0x0000_0000_7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0:ML0 <- 0xffff_fff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- ML0, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rb以16位值处理，M0，Rx，Rc以32位值处理。 Ra和Rb以固定小数点相乘、与Rx相加。结果存储到M0与Rc，结果不能用带符号的32位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmachw M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图542

fmachww Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[48:0] <- sext49(smul(Ra[31:0],Rb[15:0]))<<FXP_SHIFT tmp[48:0] <- 0x0_7fff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[48:0] <- 0x0_ffff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[49:0] <- MHn[17:0]:MLn + sext50(tmp[48:0]); MHm:MLm <- sext64(tmp2[47:0]); MHm:MLm <- 0x0000_7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[49:0] > 0x0_7fff_ffff_ffff[49:0]); MHm:MLm <- 0xffff_8000_0000_0000, CFR.OVS <- 1, if (tmp2[49:0] < 0x3_8000_0000_0000[49:0]); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rc以32位值处理, Rb以16位值处理,Mm,Mn以48位值处理。</p> <p>Ra和Rb以固定小数点相乘、与Mm相加。结果存储到Mm与Rc, 结果i不能用带符号的48位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmachww Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图 543

fmachww M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[48:0] <- sext49(amt4(Ra[31:0],Rb[15:0]))<<FXP_SHIFT tmp[48:0] <- 0x0_7fff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[48:0] <- 0x0_ffff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[49:0] <- (sext50(Rx)<<16) + sext50(tmp[48:0]); MH0:ML0 <- sext64(tmp2[47:0]); MH0:ML0 <- 0x0000_7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[49:0] > 0x0_7fff_ffff_ffff[49:0]); MH0:ML0 <- 0xffff_8000_0000_0000, CFR.OVS <- 1, if (tmp2[49:0] < 0x3_8000_0000_0000[49:0]); Rc <- MH0[15:0]:ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rc,Rx以32位值处理, Rb以16位值处理,M0以48位值处理。</p> <p>Ra和Rb以固定小数点相乘、与Rx相加。结果存储到M0和Rc, 结果能用48位带符号的表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmachww M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图 544

mac Mm,Rc,Ra,Rb,Mn

操作	MHm:MLm <- smul(Ra,Rb) + MHn:MLn; Rc <- MLm, if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rb整数相乘、与Mn相加。结果存储到Mm与Rc。		
汇编·助记	格式	影响标记	受影响的标记
mac Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 545

mac M0,Rc,Ra,Rb,Rx

操作	MH0:ML0 <- sext64(Rx) + smul(Ra,Rb); Rc <- ML0, if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra和Rb整数相乘、与Rx相加。结果存储到M0与Rc。		
汇编·助记	格式	影响标记	受影响的标记
mac M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 546

fmacww Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[64:0] <- sext65(smul(Ra,Rb)) << FXP_SHIFT; tmp[64:0] <- 0x0_7fff_ffff_ffff[64:0], CFR.OVS <- 1 if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 0)); tmp[64:0] <- 0x0_ffff_ffff_ffff[64:0], CFR.OVS <- 1 if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 1)); tmp2[65:0] <- sext66(MHn:MLn) + sext66(tmp[64:0]); MHm:MLm <- tmp2[63:0]; MHm:MLm <- 0x7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[65:0] > 0x0_7fff_ffff_ffff[65:0]); MHm:MLm <- 0x8000_0000_0000_0000, CFR.OVS <- 1, if (tmp2[65:0] < 0x3_8000_0000_0000_0000[65:0]); Rc <- MHm, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rc以32位值处理，Rb以16位值处理，Mm,Mn以48位值处理。</p> <p>Ra和Rb以固定小数点相乘、与Mn相加。结果存储到Mm与Rc，结果不能用带符号的48位表示时饱和</p>		
编程助记	幅度	影响标记	受影响的标记
fmacww Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 547

fmacww M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[64:0] <- sext65(smul(Ra,Rb))<<FXP_SHIFT; tmp[64:0] <- 0x0_7fff_ffff_ffff[64:0], CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 0)); tmp[64:0] <- 0x0_ffff_ffff_ffff[64:0], CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 1)); tmp2[65:0] <- (sext66(Rx)<<32) + sext66(tmp[64:0]); MH0:ML0 <- tmp2[63:0]; MH0:ML0 <- 0x7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[65:0] > 0x0_7fff_ffff_ffff[65:0]); MH0:ML0 <- 0x8000_0000_0000_0000, CFR.OVS <- 1, if (tmp2[65:0] < 0x3_8000_0000_0000[65:0]); Rc <- MH0, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb,Rc,Rx以32位值处理, M0以64位值处理。 Ra和Rb以固定小数点相乘、与Rx相加。结果存储到M0与Rc, 结果不能用带符号的64位表示时饱和</p>		
汇程·助记	格式	影响标记	受影响的标记
fmacww M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图 548

hmsu Mm,Rc,Ra,Rb,Mn

操作	$MHm:MLm \leftarrow MHn:MLn - smul(Ra, sext(Rb[31:16]));$ $Rc \leftarrow MLm, \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ <p>Rb以半字矢量形式处理。 Ra和Rb的上位16位整数相乘、从Mn减去，结果 存储到Mm和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
hmsu Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 549

hmsu M0,Rc,Ra,Rb,Rx

操作	$MH0:ML0 \leftarrow sext64(Rx) - smul(Ra, sext(Rb[31:16]));$ $Rc \leftarrow ML0, \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ <p>Rb以半字矢量形式处理。 Ra和Rb的上位16位整数相乘、从Rx减去，结果 存储到M0和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
hmsu M0,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 550

lmsu Mm,Rc,Ra,Rb,Mn

操作	MHm:MLm <- MHn:MLn - smul(Ra,sext(Rb[15:0])); Rc <- MLm, if (reg#(Rc) != 30 && reg#(Rc) != 31);		
	Rb以半字矢量形式处理。 Ra和Rb的上位16位整数相乘、从Mn减去，结果 存储到Mm和Rc		
汇编·助记	格式	影响标记	受影响的标记
lmsu Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 551

lmsu M0,Rc,Ra,Rb,Rx

操作	MH0:ML0 <- sext64(Rx) - smul(Ra,sext(Rb[15:0])); Rc <- ML0, if (reg#(Rc) != 30 && reg#(Rc) != 31);		
	Rb以半字矢量形式处理。 Ra和Rb的上位16位整数相乘、从Rx减去，结果 存储到M0和Rc		
汇编·助记	格式	影响标记	受影响的标记
lmsu M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 552

fmsuhh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- MHn[1:0]:MLn - sext34(tmp[32:0]); MHm:MLm <- sext64(tmp2[33:0]); MHm:MLm <- 0x0000_0000_7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm:MLm <- 0xffff_fff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb,Rc以16位值处理, Mm,Mn以32位值处理。 Ra和Rb以固定小数点相乘后, 从M减去, 结果存储到Mm和Rc, 结果不能用带符号的32 位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rb。			

图 553

fmsuhh M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP ==0)); tmp[32:0] <- 0x0_ffff_ffff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP ==1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) - sext34(tmp[32:0]) MH0:ML0 <- sext64(tmp2[31:0]); MH0:ML0 <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0:ML0 <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MH0[15:0]:ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb,Rc以16位值处理, M0,Rx以32位值处理。 Ra和Rb以固定小数点相乘后, 从Rx减去, 结果存储到Mm和Rc, 结果不能用带符号的32 位表示时饱和</p>		
汇编助记	格式	影响标记	受影响的标记
fmsuhh M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rb。			

图 554

fmsuhhr Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- MHm[2:0]:MLn - sext35(tmp[32:0]) + sext35(0x0_8000); MHm:MLn <- sext64(tmp2[31:16]:16'b0); MHm:MLn <- 0x0000_0000_7fff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7fff[18:0]); MHm:MLn <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MHm:MLn <- UD, if (PSR.FXP == 1); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以16位值处理, Mm, Mn以32位值处理。 Ra和Rb以固定小数点相乘、从Mn减去, 减去后, 从32位到16位四舍五入, 存储到Mm和Rc。 结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhhr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话, 未组装Mm, Rc变为UD。 Mm的下位16位, 经常变为0。 Rc指定为r30或r31时, 不能写入Rc			

图555

fmsuhhr M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0xffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- (sext35(Rx[15:0])<<16) - sext35(tmp[32:0]) + sext35(0x0_8000); MH0:ML0 <- sext64(tmp2[31:16]:16'b0); MH0:ML0 <- 0x0000_0000_7fff_0000, CFR.OVS <- 1, if (tmp3[34:16] > 0x0_7fff[18:0]); MH0:ML0 <- 0xffff_fff_8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MH0:ML0 <- UD, if (PSR.FXP == 1); Rc <- MH0[15:0]:ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以16位值处理, M0, Rx以32位值处理。 Ra和Rb以固定小数点相乘、从Rx减去, 减去后, 从32位到16位四舍五入, 存储到M0和Rc。结果 不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhhr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话, 未组装Mm, Rc变为uD。 Mm的下位16位, 经常变为0。 Rc指定为r30或r31时, 不能写入Rc			

图 556

fmsuhw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- MHn[1:0]:MLn - sext34(tmp[31:0]); MHm:MLm <- sext64(tmp2[33:0]); MHm:MLm <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm:MLm <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- MLm, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rb以16位值处理，Mm，Mn，Rc以32位值处理。 Ra和Rb以固定小数点相乘、从Mn减去，结果存储到Mm和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 557

fmsuhw M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(Rx) - sext34(tmp[31:0]); MH0:ML0 <- sext64(tmp[31:0]); MH0:ML0 <- 0x0000_0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0:ML0 <- 0xffff_ffff_8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc <- ML0, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rb以16位值处理，M0，Rc,Rx以32位值处理。 Ra和Rb以固定小数点相乘、从Rx减去，结果存储到M0和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhw M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 558

fmsuhww Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[48:0] <- sext49(smuli(Ra[31:0],Rb[15:0]))<<FXP_SHIFT; tmp[48:0] <- 0x0_7fff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[48:0] <- 0x0_ffff_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[49:0] <- sext50(MHn[17:0]:MLn) - sext50(tmp[48:0]); MHm:MLm <- sext64(tmp2[47:0]); MHm:MLm <- 0x0000_7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[49:0] > 0x0_7fff_ffff_ffff[49:0]); MHm:MLm <- 0xffff_8000_0000_0000, CFR.OVS <- 1, if (tmp2[49:0] < 0x3_8000_0000_0000[49:0]); Rc <- MHm[15:0]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra和Rc以32位值处理，Rb以16位值处理。 Mm、Mn以48位值处理。 Ra和Rb以固定小数点相乘，从Mn减去，结果存储到Mm和Rc。结果不能用带符号的48位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhww Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 559

fmsubww M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[48:0] <- sext49(smul(Ra[31:0],Rb[15:0]))<<FXP_SHIFT; tmp[48:0] <- 0x0_7fff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[48:0] <- 0x0_ffff_ffff[48:0], CFR.OVS <- 1, if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[49:0] <- (sext50(Rx)<<16) - sext50(tmp[48:0]); MH0:MLO <- sext64(tmp2[47:0]); MH0:MLO <- 0x0000_7fff_ffff, CFR.OVS <- 1, if (tmp2[49:0] > 0x0_7fff_ffff[49:0]); MH0:MLO <- 0xffff_8000_0000_0000, CFR.OVS <- 1, if (tmp2[49:0] < 0x3_8000_0000_0000[49:0]); Rc <- MH0[15:0]:MLO[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rc,Rx以32位值处理，Rb以16位值处理，M0以48位值处理。 Ra和Rb以固定小数点相乘，从Rx减去，结果存储到M0和Rc。结果不能用带符号的48位表示时饱和</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuhww M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图560

msu Mm,Rc,Ra,Rb,Mn

操作	$\text{MHm:MLm} \leftarrow \text{MHn:MLn} - \text{smul}(\text{Ra}, \text{Rb});$ $\text{Rc} \leftarrow \text{MLm}, \text{ if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$ <p>Ra和Rb整数相乘，从Mm减去。结果存储到Mm和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
msu Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图561

msu M0,Rc,Ra,Rb,Rx

操作	$\text{MH0:ML0} \leftarrow \text{sext64}(\text{Rx}) - \text{smul}(\text{Ra}, \text{Rb});$ $\text{Rc} \leftarrow \text{ML0}, \text{ if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$ <p>Ra和Rb整数相乘，从Rx减去。结果存储到M0和Rc</p>		
汇编·助记	格式	影响标记	受影响的标记
msu M0,Rc,Ra,Rb,Rx	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 562

fmsuww Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[64:0] <- sext65(smul(Ra,Rb))<<FXP_SHIFT tmp[64:0] <- 0x0_7fff_ffff_ffff[64:0], CFR.OVS <- 1 if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 0)); tmp[64:0] <- 0x0_ffff_ffff_ffff[64:0], CFR.OVS <- 1 if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 1)); tmp2[65:0] <- sext66(MHn:MLn) - sext66(tmp[64:0]); MHm:MLm <- tmp2[63:0]; MHm:MLm <- 0x7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[65:0] > 0x0_7fff_ffff_ffff[65:0]); MHm:MLm <- 0x8000_0000_0000_0000, CFR.OVS <- 1, if (tmp2[65:0] < 0x3_8000_0000_0000[65:0]); Rc <- MHm, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以32位值处理, Mm, Mn以64位值处理。 Ra和Rb以固定小数点相乘、从Mn减去, 结果存储到Mm和Rc。结果不能用带符号的64位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuww Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图 563

fmsuww M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[64:0] <- sext66(smul(Ra,Rb))<<FXP_SHIFT; tmp[64:0] <- 0x0_7fff_ffff_ffff[64:0], CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 0)); tmp[64:0] <- 0x0_ffff_ffff_ffff[64:0], CFR.OVS <- 1, if ((Ra == 0x8000_0000 && Rb == 0x8000_0000) && (PSR.FXP == 1)); tmp2[65:0] <- (sext66(Rx)<<32) - sext66(tmp[64:0]); MH0:ML0 <- tmp2[63:0]; MH0:ML0 <- 0x7fff_ffff_ffff, CFR.OVS <- 1, if (tmp2[65:0] > 0x0_7fff_ffff_ffff[65:0]) MH0:ML0 <- 0x8000_0000_0000_0000, CFR.OVS <- 1, if (tmp2[65:0] < 0x3_8000_0000_0000[65:0]) Rc <- MH0, if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以32位值处理，M0以64位值处理。 Ra和Rb以固定小数点相乘、从Rx减去，结果存储到M0和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
fmsuww M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图564

div MHm,Rc,MHn,Ra,Rb

操作	<pre> tmp[63:0] <- sdiv(MHn:Ra,Rb); MHm <- smod(MHn:Ra,Rb); Rc <- tmp[31:0]; Rc <- UD; MHm <- UD; CFR.OVS <- 1, if (Rb == 0 tmp[63:0] > 0x00000000_7fffffff[63:0] tmp[63:0] < 0xffffffff_80000000[63:0]); MHn: Ra作为带符号的值被Rb除。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
div MHm,Rc,MHn,Ra,Rb	32bit	CFR.OVS	—
备注			
34循环命令，识别到动态值为16位以下，就变为17循环命令。			

图565

divu MHm,Rc,MHn,Ra,Rb

操作	<pre> tmp[63:0] <- udiv(MHn:Ra,Rb); MHm <- umod(MHn:Ra,Rb); Rc <- tmp[31:0]; Rc <- UD; MHm <- UD; CFR.OVS <- 1, if (Rb == 0 tmp[63:0] >(u) 0x00000000_ffffffff[63:0]); MHn: Ra作为不带符号的值被Rb除。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
divu MHm,Rc,MHn,Ra,Rb	32bit	CFR.OVS	—
备注			
34循环命令，识别到动态值为16位以下，就变为17循环命令。			

图566

dbgm0

操作	调试用命令。		
汇编·助记	格式	影响标记	受影响的标记
dbgm0	32bit	—	—
备注			
只限为发送一次，发送2次以上的操作尚未定义。 115的使用方法被系统预约。			

图567

dbgm1

操作	调试用命令。		
汇编·助记	格式	影响标记	受影响的标记
dbgm1	32bit	—	—
备注			
只限为发送一次，发送2次以上的操作尚未定义。 115的使用方法被系统预约。			

图568

dbgm2 I15

操作	调试用命令。		
汇编·助记	格式	影响标记	受影响的标记
dbgm2 I15	32bit	—	—
备注			
<p>只限为发送一次，发送2次以上的操作尚未定义。 I15的使用方法被系统预约。</p>			

图569

dbgm3 I15

操作	调试用命令。		
汇编·助记	格式	影响标记	受影响的标记
dbgm3 I15	32bit	—	—
备注			
<p>只限为发送一次，发送2次以上的操作尚未定义。 I15的使用方法被系统预约。</p>			

图 570

vaddh Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Rb[31:16];$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[15:0];$ 各寄存器以半字矢量形式处理。 Ra和Rb相加(SIMD straight)。		
汇编·助记	格式	影响标记	受影响的标记
vaddh Rc,Ra,Rb	32bit	—	—
备注			

图 571

vxaddh Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Rb[15:0];$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[31:16];$ 各寄存器以半字矢量形式处理。 Ra和Rb相加(SIMD cross)。		
汇编·助记	格式	影响标记	受影响的标记
vxaddh Rc,Ra,Rb	32bit	—	—
备注			

图572

vhaddh Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Rb[31:16];$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[31:16];$ 各寄存器以半字矢量形式处理。 Ra和Rb相加(SIMD high)。		
汇编·助记	格式	影响标记	受影响的标记
vhaddh Rc,Ra,Rb	32bit	—	—
备注			

图573

vladdh Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Rb[15:0];$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[15:0];$ 各寄存器以半字矢量形式处理。 Ra和Rb相加(SIMD low)。		
汇编·助记	格式	影响标记	受影响的标记
vladdh Rc,Ra,Rb	32bit	—	—
备注			

图574

vaddhvh Rc,Ra,Rb

操作	<pre> tmp[16:0] <- sext17(Ra[31:16]) + sext17(Rb[31:16]); Rc[31:16] <- tmp[15: 0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Ra[15: 0]) + sext17(Rb[15: 0]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。</p> <p>Ra和Rb相加(SIMD straight)。结果不能用16位值表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaddhvh Rc,Ra,Rb	32bit	—	—
备注			

图575

vxaddhvh Rc,Ra,Rb

操作	<pre> tmp[16:0] <- sext17(Ra[31:16]) + sext17(Rb[15:0]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Ra[15: 0]) + sext17(Rb[31:16]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); 各寄存器以半字矢量形式处理。 Ra和Rb相加(SIMD cross)。结果不能用16位值表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vxaddhvh Rc,Ra,Rb	32bit	CFR.OVS	-
备注			

图576

vhaddhvh Rc,Ra,Rb

操作	<pre> tmp[16:0] <- sext17(Ra[31:16]) + sext17(Rb[31:16]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Ra[15: 0]) + sext17(Rb[31:16]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。</p> <p>Ra和Rb相加(SIMD high)。结果不能用16位值表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhaddhvh Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图577

vladdhvh Rc,Ra,Rb

操作	<pre> tmp[16:0] <- sext17(Ra[31:16]) + sext17(Rb[15: 0]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Ra[15: 0]) + sext17(Rb[15: 0]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); 各寄存器以半字矢量形式处理。 Ra和Rb相加 (SIMD low)。结果不能用16位值表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vladdhvh Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图578

vsaddh Rb,Ra,I8

操作	Raの2つの要素にI8を加算 Rb[31:16] ← Ra[31:16] + sext16(I8); Rb[15: 0] ← Ra[15: 0] + sext16(I8); 各寄存器以半字矢量形式处理。 Ra和当前值(I8)相加。		
汇编·助记	格式	影响标记	受影响的标记
vsaddh Rb,Ra,I8	32bit	—	—
备注			
I8为带符号的值。			

图579

vaddsh Rc,Ra,Rb

操作	Rc[31:16] ← (sext17(Ra[31:16]) + sext17(Rb[31:16]))[16: 1]; Rc[15: 0] ← (sext17(Ra[15: 0]) + sext17(Rb[15: 0]))[16: 1]; 各寄存器以半字矢量形式处理。 Ra和Rb相加、结果向右算术移位1位。		
汇编·助记	格式	影响标记	受影响的标记
vaddsh Rc,Ra,Rb	32bit	—	—
备注			

图580

vaddsrh Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow (sxt17(Ra[31:16]) + sxt17(Rb[31:16]) + 0x0_0001[16:0])[16:1];$ $Rc[15:0] \leftarrow (sxt17(Ra[15:0]) + sxt17(Rb[15:0]) + 0x0_0001[16:0])[16:1];$ <p>各寄存器以半字矢量形式处理。 Ra和Rb和1相加、结果向右算术移位1位。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaddsrh Rc,Ra,Rb	32bit	—	—
备注			

图581

vaddhvc Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Ra[31:16], \text{ if } (VC2 == 0);$ $Rc[31:16] \leftarrow Ra[31:16] + Rb[31:16], \text{ if } (VC2 == 1);$ $Rc[15:0] \leftarrow Ra[15:0] + Ra[15:0], \text{ if } (VC2 == 0);$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[15:0], \text{ if } (VC2 == 1);$ <p>用于图像处理的动态补偿。 各寄存器以半字矢量形式处理。 Ra和Ra或Rb相加。由VC2决定和Ra还是Rb相加。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaddhvc Rc,Ra,Rb	32bit	—	VC2
备注			

图582

vaddrhvc Rc,Ra,Rb

操作	$Rc[31:16] \leftarrow Ra[31:16] + Ra[31:16] + sext16(1), \text{ if } (VC2 == 0);$ $Rc[31:16] \leftarrow Ra[31:16] + Rb[31:16] + sext16(1), \text{ if } (VC2 == 1);$ $Rc[15:0] \leftarrow Ra[15:0] + Ra[15:0] + sext16(1), \text{ if } (VC2 == 0);$ $Rc[15:0] \leftarrow Ra[15:0] + Rb[15:0] + sext16(1), \text{ if } (VC2 == 1);$ 各寄存器以半字矢量形式处理。 Ra和Ra或Rb相加，然后为了四舍五入与1相加， 由VC2决定与Ra还是Rb相加。		
汇编·助记	格式	影响标记	受影响的标记
vaddrhvc Rc,Ra,Rb	32bit	-	VC2
备注			

图583

vaddb Rc,Ra,Rb

操作	$Rc[31:24] \leftarrow Ra[31:24] + Rb[31:24];$ $Rc[23:16] \leftarrow Ra[23:16] + Rb[23:16];$ $Rc[15:8] \leftarrow Ra[15:8] + Rb[15:8];$ $Rc[7:0] \leftarrow Ra[7:0] + Rb[7:0];$ 各寄存器以字节矢量形式处理。 Ra和Rb相加。		
汇编·助记	格式	影响标记	受影响的标记
vaddb Rc,Ra,Rb	32bit	-	-
备注			

图584

vsaddb Rb,Ra,I8

操作	$Rb[31:24] \leftarrow Ra[31:24] + I8;$ $Rb[23:16] \leftarrow Ra[23:16] + I8;$ $Rb[15: 8] \leftarrow Ra[15: 8] + I8;$ $Rb[7: 0] \leftarrow Ra[7: 0] + I8;$ 各寄存器以字节矢量形式处理。 Ra和当前值(I8)相加。		
汇编·助记	格式	影响标记	受影响的标记
vsaddb Rb,Ra,I8	32bit	—	—
备注			
18为带符号的值。			

图585

vaddsb Rc,Ra,Rb

操作	$Rc[31:24] \leftarrow (sext9(Ra[31:24]) + sext9(Rb[31:24]))[8: 1];$ $Rc[23:16] \leftarrow (sext9(Ra[23:16]) + sext9(Rb[23:16]))[8: 1];$ $Rc[15: 8] \leftarrow (sext9(Ra[15: 8]) + sext9(Rb[15: 8]))[8: 1];$ $Rc[7: 0] \leftarrow (sext9(Ra[7: 0]) + sext9(Rb[7: 0]))[8: 1];$ 各寄存器以字节矢量形式处理。 Ra和Rb相加、结果向右算术移位1位。		
汇编·助记	格式	影响标记	受影响的标记
vaddsb Rc,Ra,Rb	32bit	—	—
备注			

vaddsrB Rc,Ra,Rb

操作	$\text{Rc}[31:24] \leftarrow (\text{sext9}(\text{Ra}[31:24]) + \text{sext9}(\text{Rb}[31:24]) + 0x001[8:0])[8:1];$ $\text{Rc}[23:16] \leftarrow (\text{sext9}(\text{Ra}[23:16]) + \text{sext9}(\text{Rb}[23:16]) + 0x001[8:0])[8:1];$ $\text{Rc}[15:8] \leftarrow (\text{sext9}(\text{Ra}[15:8]) + \text{sext9}(\text{Rb}[15:8]) + 0x001[8:0])[8:1];$ $\text{Rc}[7:0] \leftarrow (\text{sext9}(\text{Ra}[7:0]) + \text{sext9}(\text{Rb}[7:0]) + 0x001[8:0])[8:1];$		
	<p>各寄存器以半字矢量形式处理。 Ra和Rb和1相加，结果向右算术移位1位。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaddsrB Rc,Ra,Rb	32bit	—	—
备注			

图586

vsubh Rc,Rb,Ra

操作	$\text{Rc}[31:16] \leftarrow \text{Rb}[31:16] - \text{Ra}[31:16];$ $\text{Rc}[15:0] \leftarrow \text{Rb}[15:0] - \text{Ra}[15:0];$		
	<p>各寄存器以半字矢量形式处理。 从Rb减去Ra（SIMD straight）。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsubh Rc,Rb,Ra	32bit	—	—
备注			
注意操作数顺序。			

图587

vxsubh Rc,Rb,Ra

操作	Rc[31:16] <- Rb[31:16] - Ra[15: 0]; Rc[15: 0] <- Rb[15: 0] - Ra[31:16]; 各寄存器以半字矢量形式处理。 从Rb减去Ra（SIMD cross）。		
汇编·助记	格式	影响标记	受影响的标记
vxsubh Rc,Rb,Ra	32bit	-	-
备注			
注意操作数顺序。			

图 588

vhsbh Rc,Rb,Ra

操作	Rc[31:16] <- Rb[31:16] - Ra[31:16]; Rc[15: 0] <- Rb[15: 0] - Ra[31:16]; 各寄存器以半字矢量形式处理。 从Rb减去Ra（SIMD high）。		
汇编·助记	格式	影响标记	受影响的标记
vhsbh Rc,Rb,Ra	32bit	-	-
备注			
注意操作数顺序。			

图 589

vsubh Rc,Rb,Ra

操作	$Rc[31:16] \leftarrow Rb[31:16] - Ra[15:0];$ $Rc[15:0] \leftarrow Rb[15:0] - Ra[15:0];$ 各寄存器以半字矢量形式处理。 从Rb减去Ra (SIMD low)。		
汇编·助记	格式	影响标记	受影响的标记
vsubh Rc,Rb,Ra	32bit	—	—
备注			
注意操作数顺序。			

图590

vsubhvh Rc,Rb,Ra

操作	<pre> tmp[16:0] <- sext17(Rb[31:16]) - sext17(Ra[31:16]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Rb[15: 0]) - sext17(Ra[15: 0]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。</p> <p>从Rb减去Ra (SIMD straight), 结果不能用带符号的16位值表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsubhvh Rc,Rb,Ra	32bit	CFR.OVS	—
备注			
注意操作数顺序。			

图 591

vsubhvh Rc,Rb,Ra

操作	<pre> tmp[16:0] <- sext17(Rb[31:16]) - sext17(Ra[15:0]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, If (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, If (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Rb[15: 0]) - sext17(Ra[31:16]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, If (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, If (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。</p> <p>从Rb减去Ra (SIMD cross), 结果不能用带符号的16位值表示饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsubhvh Rc,Rb,Ra	32bit	CFR.OVS	-
备注			
注意操作数顺序。			

图592

vsubhvh Rc,Rb,Ra

操作	<pre> tmp[16:0] <- sext17(Rb[31:16]) - sext17(Ra[31:16]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Rb[15: 0]) - sext17(Ra[31:16]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。 从Rb减去Ra (SIMD high), 结果不能用带符号的16位值表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsubhvh Rc,Rb,Ra	32bit	CFR.OVS	—
备注			
注意操作数顺序。			

图593

vlsubhvh Rc,Rb,Ra

操作	<pre> tmp[16:0] <- sext17(Rb[31:16]) - sext17(Ra[15: 0]); Rc[31:16] <- tmp[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Rb[15: 0]) - sext17(Ra[15: 0]); Rc[15: 0] <- tmp[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。 从Rb减去Ra（SIMD low），结果不能用带符号的16位值表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlsubhvh Rc,Rb,Ra	32bit	CFR.OVS	-
备注			
注意操作数顺序。			

图594

vssubh Rb,Ra,I8

操作	=> vsaddh Rb,Ra,-I8		
	Rb[31:16] <- Ra[31:16] - sext16(I8); Rb[15: 0] <- Ra[15: 0] - sext16(I8); 各寄存器以半字矢量形式处理。 从Rb减去当前值 (I8) 。		
汇编·助记	格式	影响标记	受影响的标记
vssubh Rb,Ra,I8	相当于32bit	—	—
备注			
注意操作数顺序。			

图595

vsubb Rc,Rb,Ra

操作	Rc[31:24] <- Rb[31:24] - Ra[31:24]; Rc[23:16] <- Rb[23:16] - Ra[23:16]; Rc[15: 8] <- Rb[15: 8] - Ra[15: 8]; Rc[7: 0] <- Rb[7: 0] - Ra[7: 0]; 各寄存器以字节矢量形式处理。 从Rb减去Ra。		
	汇编·助记	格式	影响标记
vsubb Rc,Rb,Ra	32bit	—	—
备注			
注意操作数顺序。			

图596

vssubb Rb,Ra,I8

操作	=> vsaddb Rb,Ra,-I8		
	Rb[31:24] <- Ra[31:24] - I8; Rb[23:16] <- Ra[23:16] - I8; Rb[15: 8] <- Ra[15: 8] - I8; Rb[7: 0] <- Ra[7: 0] - I8;		
各寄存器以字节矢量形式处理。 从Rb减去当前值（I8）。			
汇编·助记	格式	影响标记	受影响的标记
vssubb Rb,Ra,I8	相当于32bit	-	-
备注			
I8为带符号的值。			

图597

vsubsh Rc,Rb,Ra

操作	Rc[31:16] <- (sext17(Rb[31:16]) - sext17(Ra[31:16]))[16:1]; Rc[15: 0] <- (sext17(Rb[15: 0]) - sext17(Ra[15: 0]))[16:1];		
	各寄存器以半字矢量形式处理。 从Rb减去Ra，结果向右算术移位1位。		
汇编·助记	格式	影响标记	受影响的标记
vsubsh Rc,Rb,Ra	32bit	-	-
备注			
注意操作数顺序。			

图598

vsrsubh Rb,Ra,I8

操作	Rb[31:16] <- sext16(I8) - Ra[31:16]; Rb[15: 0] <- sext16(I8) - Ra[15: 0]; 各寄存器以半字矢量形式处理。 从当前值 (I8) 减去Ra。		
汇编·助记	格式	影响标记	受影响的标记
vsrsubh Rb,Ra,I8	32bit	—	—
备注			
I8为带符号的值。 注意操作数顺序。			

图599

vsrsubb Rb,Ra,I8

操作	Rb[31:24] <- I8 - Ra[31:24]; Rb[23:16] <- I8 - Ra[23:16]; Rb[15: 8] <- I8 - Ra[15: 8]; Rb[7: 0] <- I8 - Ra[7: 0]; 各寄存器以半字矢量形式处理。 从当前值 (I8) 减去Ra。		
汇编·助记	格式	影响标记	受影响的标记
vsrsubb Rb,Ra,I8	32bit	—	—
备注			
I8为带符号的值。 注意操作数顺序。			

图600

vsumh Rb,Ra

操作	$\text{Rb}[31:0] \leftarrow \text{sext}(\text{Ra}[31:16]) + \text{sext}(\text{Ra}[15:0]);$ <p>各寄存器以半字矢量形式处理。 Ra的上位16位值和Ra的下位16位值相加，存储到Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsumh Rb,Ra	32bit	-	-
备注			

图601

vsumh2 Rb,Ra

操作	$\text{Rb}[31:16] \leftarrow \text{sext16}(\text{Ra}[31:24]) + \text{sext16}(\text{Ra}[23:16]);$ $\text{Rb}[15:0] \leftarrow \text{sext16}(\text{Ra}[15:8]) + \text{sext16}(\text{Ra}[7:0]);$ <p>用于图像处理的动态补偿。 Ra以字节矢量形式处理，Rb以半字矢量形式处理。 Ra的上位16位值的2个8位值，Ra的下位16位值的2个8位值相加，结果以半字矢量形式存储到Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsumh2 Rb,Ra	32bit	-	-
备注			

图602

vsumrh2 Rb,Ra

操作	$\text{Rb}[31:16] \leftarrow \text{sext16}(\text{Ra}[31:24]) + \text{sext16}(\text{Ra}[23:16]) + \text{sext16}(1);$ $\text{Rb}[15:0] \leftarrow \text{sext16}(\text{Ra}[15:8]) + \text{sext16}(\text{Ra}[7:0]) + \text{sext16}(1);$		
	<p>用于图像处理的动态补偿。</p> <p>Ra以字节矢量形式处理，Rb以半字矢量形式处理。</p> <p>Ra的上位16位值的2个8位值，Ra的下位16位的2个8位值相加，然后为了四舍五入与1相加，结果以半字矢量形式存储到Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsumrh2 Rb,Ra	32bit	-	-
备注			

图603

vnegh Rb,Ra

操作	$\Rightarrow \text{vrsubh Rb,Ra,0}$		
	$\text{Rb}[31:16] \leftarrow -\text{Ra}[31:16];$ $\text{Rb}[15:0] \leftarrow -\text{Ra}[15:0];$ <p>各寄存器以半字矢量形式处理。</p> <p>反转Ra的符号。</p>		
汇编·助记	格式	影响标记	受影响的标记
vnegh Rb,Ra	相当于32bit	-	-
备注			

图604

vneghvh Rb,Ra

操作	<pre> tmp[16:0] <- sext17(Ra[31:16]); Rb[31:16] <- tmp[15:0]; Rb[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rb[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); tmp[16:0] <- sext17(Ra[15: 0]); Rb[15: 0] <- tmp[15:0]; Rb[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[16:0] > 0x0_7fff[16:0]); Rb[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[16:0] < 0x1_8000[16:0]); </pre> <p>各寄存器以半字矢量形式处理。 反转Ra符号，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vneghvh Rb,Ra	32bit	CFR.OVS	—
备注			

图605

vnegb Rb,Ra

操作	=> vsrsubb Rb,Ra,0		
	Rb[31:24] <- - Ra[31:24]; Rb[23:16] <- - Ra[23:16]; Rb[15: 8] <- - Ra[15: 8]; Rb[7: 0] <- - Ra[7: 0]; 各寄存器以半字矢量形式处理。 反转Ra的符号。		
汇编·助记	格式	影响标记	受影响的标记
vnegb Rb,Ra		-	-
备注			

图606

vabshvh Rb,Ra

操作	Rb[31:16] <- Ra[31:16], if (Ra[31:16] >= 0); Rb[31:16] <- -Ra[31:16], if ((Ra[31:16] < 0)); Rb[31:16] <- 7fff, CFR.OVS <- 1, if (Ra[31:16] == 0x8000); Rb[15: 0] <- Ra[15: 0], if (Ra[15: 0] >= 0); Rb[15: 0] <- -Ra[15: 0], if ((Ra[15: 0] < 0)); Rb[15: 0] <- 7fff, CFR.OVS <- 1, if (Ra[15: 0] == 0x8000); 各寄存器以半字矢量形式处理。 取Ra的绝对值，结果不能用带符号的16位表示时饱和。		
	编程·助记	幅度	影响标记
vabshvh Rb,Ra	32bit	CFR.OVS	-
备注			

图607

vasubb Rc,Rb,Ra

操作	VC3:Rc[31:24] <- sext9(Rb[31:24]) - sext9(Ra[31:24]); VC2:Rc[23:16] <- sext9(Rb[23:16]) - sext9(Ra[23:16]); VC1:Rc[15: 8] <- sext9(Rb[15: 8]) - sext9(Ra[15: 8]); VC0:Rc[7: 0] <- sext9(Rb[7: 0]) - sext9(Ra[7: 0]);		
	用于图像处理动作预测的差分绝对值和 与vabssumb配合使用。 各寄存器以字节矢量形式处理。 从Rb减去Ra，各8位值的符号位分别存储到VC0—VC3。		
汇编·助记	格式	影响标记	受影响的标记
vasubb Rc,Rb,Ra	32bit	VC0,VC1,VC2,VC3	—
备注			
注意操作数顺序。			

图 608

vsgnh Rb,Ra

操作	Rb[31:16] <- 0x0001, if (Ra[31:16] > 0); Rb[31:16] <- 0x0000, if (Ra[31:16] == 0); Rb[31:16] <- 0xffff, if (Ra[31:16] < 0); Rb[15:0] <- 0x0001, if (Ra[15:0] > 0); Rb[15:0] <- 0x0000, if (Ra[15:0] == 0); Rb[15:0] <- 0xffff, if (Ra[15:0] < 0); 用于图像处理的逆量子化。 各寄存器以半字矢量形式处理。 Ra为正时输出1, 为负时输出-1, 为0时输出0。		
汇编·助记	格式	影响标记	受影响的标记
vsgnh Rb,Ra	32bit	-	-
备注			

图 609

vmaxh Rc,Ra,Rb

操作	Rc[31:16] <- max(Ra[31:16],Rb[31:16]); Rc[15:0] <- max(Ra[15:0],Rb[15:0]); 各寄存器以半字矢量形式处理。 Ra与Rb相比, 较大的一方存储到Rc。		
汇编·助记	格式	影响标记	受影响的标记
vmaxh Rc,Ra,Rb	32bit	-	-
备注			

图 610

vmaxb Rc,Ra,Rb

操作	Rc[31:24] <- max(Ra[31:24],Rb[31:24]); Rc[23:16] <- max(Ra[23:16],Rb[23:16]); Rc[15: 8] <- max(Ra[15: 8],Rb[15: 8]); Rc[7: 0] <- max(Ra[7: 0],Rb[7: 0]); 各寄存器以字节矢量形式处理。 Ra与Rb相比,较大的一方存储到Rc。		
汇编·助记	格式	影响标记	受影响的标记
vmaxb Rc,Ra,Rb	32bit	-	-
备注			

图611

vminh Rc,Ra,Rb

操作	Rc[31:16] <- min(Ra[31:16],Rb[31:16]); Rc[15: 0] <- min(Ra[15: 0],Rb[15: 0]); 各寄存器以半字矢量形式处理。 Ra与Rb相比,较小一方存储到Rc。		
汇编·助记	格式	影响标记	受影响的标记
vminh Rc,Ra,Rb	32bit	-	-
备注			

图612

vminb Rc,Ra,Rb

操作	Rc[31:24] <- min(Ra[31:24],Rb[31:24]); Rc[23:16] <- min(Ra[23:16],Rb[23:16]); Rc[15: 8] <- min(Ra[15: 8],Rb[15: 8]); Rc[7: 0] <- min(Ra[7: 0],Rb[7: 0]); 各寄存器以半字矢量形式处理。 Ra与Rb相比,较大的一方存储到Rc。		
汇编·助记	格式	影响标记	受影响的标记
vminb Rc,Ra,Rb	32bit	-	-
备注			

图613

vsel Rc,Ra,Rb

操作	Rc[31:24] <- (VC3 == 1) ? Ra[31:24] : Rb[31:24]; Rc[23:16] <- (VC2 == 1) ? Ra[23:16] : Rb[23:16]; Rc[15: 8] <- (VC1 == 1) ? Ra[15: 8] : Rb[15: 8]; Rc[7: 0] <- (VC0 == 1) ? Ra[7: 0] : Rb[7: 0]; 各寄存器以半字矢量形式处理。 根据VC0—VC3的值,选择Ra或Rb,存储到Rc。		
汇编·助记	格式	影响标记	受影响的标记
vsel Rc,Ra,Rb	32bit	-	VC0,VC1,VC2,VC3
备注			

图614

图615

vmovt Rb,Ra

操作	=> vsel Rb,Ra,Rb		
	Rb[31:24] <- (VC3 == 1) ? Ra[31:24] : Rb[31:24]; Rb[23:16] <- (VC2 == 1) ? Ra[23:16] : Rb[23:16]; Rb[15: 8] <- (VC1 == 1) ? Ra[15: 8] : Rb[15: 8]; Rb[7: 0] <- (VC0 == 1) ? Ra[7: 0] : Rb[7: 0];		
各寄存器以字节矢量形式处理。			
根据VC0~VC3的值，选择Ra或Rb存储到Rb。			
汇编·助记	格式	影响标记	受影响的标记
vmovt Rb,Ra		—	VC0,VC1,VC2,VC3
备注			

图616

vscmpeq b Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] == sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] == sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] == sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] == sext8(I5));		
	Ra以字节矢量形式处理。 根据VC0~VC3的值，选择Ra或Rb存储到Rc。 与当前值(25)比较，结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpeq b Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图617

vscmpneb Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] != sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] != sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] != sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] != sext8(I5)); Ra以字节矢量形式处理。 Ra与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpneb Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图618

vscmpgtb Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] > sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] > sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] > sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] > sext8(I5)); Ra以字节矢量形式处理。 Ra与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpgtb Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图 619

vscmpleb Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] <= sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] <= sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] <= sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] <= sext8(I5)); Ra以字节矢量形式处理。 Ra与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpleb Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图 620

vscmpgeb Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] >= sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] >= sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] >= sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] >= sext8(I5)); Ra以字节矢量形式处理。 Ra与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpgeb Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图621

vscmpltb Ra,I5

操作	VC3 <- 0; VC3 <- 1, if(Ra[31:24] < sext8(I5)); VC2 <- 0; VC2 <- 1, if(Ra[23:16] < sext8(I5)); VC1 <- 0; VC1 <- 1, if(Ra[15: 8] < sext8(I5)); VC0 <- 0; VC0 <- 1, if(Ra[7: 0] < sext8(I5)); Ra以字节矢量形式处理。 Ra与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpltb Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图622

vscmpeqh Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] == sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] == sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpeqh Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图623

vscmpneh Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] != sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] != sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpneh Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图624

vscmpgth Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] > sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] > sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpgth Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图625

vscmpleh Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] <= sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] <= sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpleh Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图626

vscmpgeh Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] >= sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] >= sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmpgeh Ra,I5	32bit	VC0,VC1, VC2,VC3	—
备注			
15为带符号的值。			

图627

vscmplth Ra,I5

操作	VC3:VC2 <- 0:0; VC3:VC2 <- 1:1, if(Ra[31:16] < sext16(I5)); VC1:VC0 <- 0:0; VC1:VC0 <- 1:1, if(Ra[15: 0] < sext16(I5)); Ra以半字矢量形式处理。 与当前值(15)比较, 结果存储到VC0~VC3。		
汇编·助记	格式	影响标记	受影响的标记
vscmplth Ra,I5	32bit	—	—
备注			
15为带符号的值。			

图628

vcmpeqh Ra,Rb

操作	<pre> if (Ra[31:16]==Rb[31:16]) { VC3:VC2<-1:1; } else { VC3:VC2<-0:0; } if (Ra[15:0]==Rb[15:0]) { VC1:VC0<-1:1; } else { VC1:VC0<-0:0; } </pre> <p>各寄存器以半字矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpeqh Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图629

vcmpneh Ra,Rb

操作	<pre> if (Ra[31:16] != Rb[31:16]) { VC3:VC2 <- 1:1; } else { VC3:VC2 <- 0:0; } if (Ra[15:0] != Rb[15:0]) { VC1:VC0 <- 1:1; } else { VC1:VC0 <- 0:0; } </pre> <p>各寄存器以半字矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpneh Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图 630

vcmpgth Ra,Rb

操作	<pre> if (Ra[31:16]> Rb[31:16]) { VC3:VC2<-1:1; } else { VC3:VC2<-0:0; } if (Ra[15: 0]> Rb[15: 0]) { VC1:VC0<-1:1; } else { VC1:VC0<-0:0; } </pre> <p>各寄存器以半字矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpgth Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图631

vcmpleh Ra,Rb

操作	<pre> if (Ra[31:16]<=Rb[31:16]) { VC3:VC2<-1:1; } else { VC3:VC2<-0:0; } if (Ra[15:0]<=Rb[15:0]) { VC1:VC0<-1:1; } else { VC1:VC0<-0:0; } 各寄存器以半字矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vcmpleh Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图 632

vcmpgeh Ra,Rb

操作	<pre> if (Ra[31:16]>=Rb[31:16]) { VC3:VC2<-1:1; } else { VC3:VC2<-0:0; } if (Ra[15:0]>=Rb[15:0]) { VC1:VC0<-1:1; } else { VC1:VC0<-0:0; } </pre> <p>各寄存器以半字矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpgeh Ra,Rb	32bit	VC0,VC1,VC2,VC3	-
备注			

图 633

vcmplth Ra,Rb

操作	<pre> if (Ra[31:16]< Rb[31:16]) { VC3:VC2<-1:1; } else { VC3:VC2<-0:0; } if (Ra[15: 0]< Rb[15: 0]) { VC1:VC0<-1:1; } else { VC1:VC0<-0:0; } </pre> <p>各寄存器以半字矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmplth Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图 634

vcmpeq b Ra,Rb

操作	<pre> if (Ra[31:24]==Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16]==Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8]==Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0]==Rb[7: 0]) { VC0<-1; } else { VC0<-0; } </pre> <p>各寄存器以字节矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpeq b Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图635

vcmpneb Ra,Rb

操作	<pre>if (Ra[31:24]!=Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16]!=Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8]!=Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0]!=Rb[7: 0]) { VC0<-1; } else { VC0<-0; }</pre> <p>各寄存器以字节矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpneb Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图636

vcmpgtb Ra,Rb

操作	<pre>if (Ra[31:24]> Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16]> Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8]> Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0]> Rb[7: 0]) { VC0<-1; } else { VC0<-0; }</pre> <p>各寄存器以字节矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpgtb Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图 637

vcmpleb Ra,Rb

操作	<pre>if (Ra[31:24] <= Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16] <= Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8] <= Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0] <= Rb[7: 0]) { VC0<-1; } else { VC0<-0; }</pre> <p>各寄存器以字节矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpleb Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图 638

vcmpgeb Ra,Rb

操作	<pre>if (Ra[31:24] >= Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16] >= Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8] >= Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0] >= Rb[7: 0]) { VC0<-1; } else { VC0<-0; }</pre> <p>各寄存器以字节矢量形式处理。 Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpgeb Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图639

vcmpltb Ra,Rb

操作	<pre>if (Ra[31:24]< Rb[31:24]) { VC3<-1; } else { VC3<-0; } if (Ra[23:16]< Rb[23:16]) { VC2<-1; } else { VC2<-0; } if (Ra[15: 8]< Rb[15: 8]) { VC1<-1; } else { VC1<-0; } if (Ra[7: 0]< Rb[7: 0]) { VC0<-1; } else { VC0<-0; }</pre> <p>各寄存器以字节矢量形式处理。</p> <p>Ra与Rb比较，结果存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vcmpltb Ra,Rb	32bit	VC0,VC1,VC2,VC3	—
备注			

图640

vaslh Rc,Ra,Rb

操作	<pre>tmp[4:0] <- Rb[4:0]; tmp[4:0] <- 15, if (Rb > 0x0000_000f); tmp[4:0] <- -15, if (Rb < 0xffff_fff1); Rc[31:16] <- Ra[31:16] << abs(tmp[4:0]); Rc[31:16] <- Ra[31:16] >> abs(tmp[4:0]), if (tmp[4:0] < 0); Rc[15: 0] <- Ra[15: 0] << abs(tmp[4:0]); Rc[15: 0] <- Ra[15: 0] >> abs(tmp[4:0]), if (tmp[4:0] < 0);</pre> <p>Ra与Rc以半字矢量形式处理。</p> <p>将Ra算术左移、Rb所示的位数，Rb在±15内饱和，为负时进行算术右移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaslh Rc,Ra,Rb	32bit	—	—
备注			

图641

vash Rb,Ra,I4

操作	Rb[31:16] <- Ra[31:16] << I4[3:0]; Rb[15:0] <- Ra[15:0] << I4[3:0]; 各寄存器以半字矢量形式处理。 Ra算术左位移当前值(14)位数。		
汇编·助记	格式	影响标记	受影响的标记
vash Rb,Ra,I4	32bit	—	—
备注			

图642

vaslvh Rc,Ra,Rb

操作	<pre> tmp[4:0] <- Rb[4:0]; tmp[4:0] <- 15, if (Rb > 0x0000_000f); tmp[4:0] <- -15, if (Rb < 0xffff_fff1); tmp2[30:0] <- sext31(Ra[31:16]) << abs(tmp[4:0]); tmp2[30:0] <- sext31(Ra[31:16]) >> abs(tmp[4:0]),if (tmp[4:0] < 0); Rc[31:16] <- tmp2[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp2[30:0] > 0x0000_7fff[30:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp2[30:0] < 0x7fff_8000[30:0]); tmp2[30:0] <- sext31(Ra[15: 0]) << abs(tmp[4:0]); tmp2[30:0] <- sext31(Ra[15: 0]) >> abs(tmp[4:0]),if (tmp[4:0] < 0); Rc[15: 0] <- tmp2[15:0]; Rc[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp2[30:0] > 0x0000_7fff[30:0]); Rc[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp2[30:0] < 0x7fff_8000[30:0]); </pre> <p>Ra与Rc以半字矢量形式处理。</p> <p>Ra算术左位移Rb所示位数，结果不能用带符号的16位表示时，Rb在±15的饱和，为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaslvh Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图643

vaslvh Rb,Ra,I4

操作	<pre> tmp[30:0] <- sext31(Ra[31:16]) << I4[3:0]; Rb[31:16] <- Ra[31:16] << I4[3:0]; Rb[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[30:0] > 0x0000_7fff[30:0]); Rb[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp[30:0] < 0x7fff_8000[30:0]); tmp[30:0] <- sext31(Ra[15: 0]) << I4[3:0]; Rb[15: 0] <- Ra[15: 0] << I4[3:0]; Rb[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[30:0] > 0x0000_7fff[30:0]); Rb[15: 0] <- 0x8000, CFR.OVS <- 1, if (tmp[30:0] < 0x7fff_8000[30:0]); </pre> <p>各寄存器以半字矢量形式处理。</p> <p>Ra作为2个16位值处理，算术左位移当前值(14)的位数。 结果不能用带符号的16位表示时为饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaslvh Rb,Ra,I4	32bit	CFR.OVS	—\
备注			
I4为不带符号的值。			

图644

vasrh Rb,Ra,I4

操作	Rb[31:16] <- Ra[31:16] >> I4[3:0]; Rb[15: 0] <- Ra[15: 0] >> I4[3:0]; 各寄存器以半字矢量形式处理。 Ra算术左移当前值(I4)位数。		
汇编·助记	格式	影响标记	受影响的标记
vasrh Rb,Ra,I4	32bit	—	—
备注			
I4为不带符号的值。			

图 645

vasrvh Rc,Ra,Rb

操作	<pre> tmp[4:0] <- Rb[4:0]; tmp[4:0] <- 15, if (Rb > 0x0000_000f); tmp[4:0] <- -15, if (Rb < 0xffff_fff1); tmp2[30:0] <- sext31(Ra[31:16]) >> abs(tmp[4:0]); tmp2[30:0] <- sext31(Ra[31:16]) << abs(tmp[4:0]), if (tmp[4:0] < 0); Rc[31:16] <- tmp2[15:0]; Rc[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp2[30:0] > 0x0000_7fff[30:0]); Rc[31:16] <- 0x8000, CFR.OVS <- 1, if (tmp2[30:0] < 0x7fff_8000[30:0]); tmp2[30:0] <- sext31(Ra[15:0]) >> abs(tmp[4:0]); tmp2[30:0] <- sext31(Ra[15:0]) << abs(tmp[4:0]), if (tmp[4:0] < 0); Rc[15:0] <- tmp2[15:0]; Rc[15:0] <- 0x7fff, CFR.OVS <- 1, if (tmp2[30:0] > 0x0000_7fff[30:0]); Rc[15:0] <- 0x8000, CFR.OVS <- 1, if (tmp2[30:0] < 0x7fff_8000[30:0]); Ra与Rc以半字矢量形式处理。 Ra算术左移Rb所示位数，结果不能用带符号的16位表示时， Rb在±15的饱和，为负时进行算术左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vasrvh Rc,Ra,Rb	32bit	CFR.OVS	—
备注			

图646

vslh Rc,Ra,Rb

操作	=> vash Rc,Ra,Rb		
	<pre> tmp[4:0] <- Rb[4:0]; tmp[4:0] <- 15, if (Rb > 0x0000_000f); tmp[4:0] <- -15, if (Rb < 0xffff_fff1); Rc[31:16] <- Ra[31:16] << abs(tmp[4:0]); Rc[31:16] <- Ra[31:16] >> abs(tmp[4:0]), if (tmp[4:0] < 0); Rc[15:0] <- Ra[15:0] << abs(tmp[4:0]); Rc[15:0] <- Ra[15:0] >> abs(tmp[4:0]), if (tmp[4:0] < 0); </pre> <p>Ra与Rc以半字矢量形式处理。</p> <p>Ra向Rb所示位数逻辑左位移，Rb在±15内饱和， 为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vslh Rc,Ra,Rb		—	—
备注			
※ 位移量为负时，变为算术右位移。			

图647

vslh Rb,Ra,I4

操作	\Rightarrow vash Rb,Ra,I4 Rb[31:16] <- Ra[31:16] << I4[3:0]; Rb[15: 0] <- Ra[15: 0] << I4[3:0]; 各寄存器以半字矢量形式处理。 Ra逻辑左位移当前值(14)位数。		
汇编·助记	格式	影响标记	受影响的标记
vslh Rb,Ra,I4	相当于32bit	—	—
备注			
I4为不带符号的值。			

图648

vlsrh Rc,Ra,Rb

操作	$tmp[4:0] <- Rb[4:0];$ $tmp[4:0] <- 15, \text{ if } (Rb > 0x0000_000f);$ $tmp[4:0] <- -15, \text{ if } (Rb < 0xffff_fff1);$ $Rc[31:16] <- Ra[31:16] >>> \text{abs}(tmp[4:0]);$ $Rc[31:16] <- Ra[31:16] << \text{abs}(tmp[4:0]), \text{ if } (tmp[4:0] < 0);$ $Rc[15: 0] <- Ra[15: 0] >>> \text{abs}(tmp[4:0]);$ $Rc[15: 0] <- Ra[15: 0] << \text{abs}(tmp[4:0]), \text{ if } (tmp[4:0] < 0);$ Ra与Rc以半字矢量形式处理。 Ra逻辑右位移、Rb所示的位数，Rb在±15内饱和， 为负时进行逻辑左右位移。		
汇编·助记	格式	影响标记	受影响的标记
vlsrh Rc,Ra,Rb	32bit	—	—
备注			

图649

vlsrh Rb,Ra,I4

操作	Rb[31:16] <- Ra[31:16] >>> I4[3:0]; Rb[15: 0] <- Ra[15: 0] >>> I4[3:0]; 各寄存器以半字矢量形式处理。 Ra逻辑右位移当前值(14)位数。		
汇编·助记	格式	影响标记	受影响的标记
vlsrh Rb,Ra,I4	32bit	—	—
备注			
14为不带符号的值。			

图650

vrolh Rc,Ra,Rb

操作	tmp[31:16] <- Ra[31:16] << Rb[3:0]; tmp2[31:16] <- Ra[31:16] >>> (16 - Rb[3:0]); Rc[31:16] <- tmp[31:16] tmp2[31:16]; tmp[15: 0] <- Ra[15: 0] << Rb[3:0]; tmp2[15: 0] <- Ra[15: 0] >>> (16 - Rb[3:0]); Rc[15: 0] <- tmp[15: 0] tmp2[15: 0]; Ra和Rc以半字矢量形式处理。 Ra向左旋转Rb所示的位数。		
汇编·助记	格式	影响标记	受影响的标记
vrolh Rc,Ra,Rb	32bit	—	—
备注			

图651

vrolh Rb,Ra,I4

操作	<pre>tmp1[31:16] <- Ra[31:16] << I4[3:0]; tmp2[31:16] <- Ra[31:16] >>> (16-I4[3:0]); tmp1[15: 0] <- Ra[15: 0] << I4[3:0]; tmp2[15: 0] <- Ra[15: 0] >>> (16-I4[3:0]); Rb <- tmp1[31:0] tmp2[31:0];</pre> <p>各寄存器以半字矢量形式处理。 Ra向左旋转当前值(14)的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
vrolh Rb,Ra,I4	32bit	—	—
备注			
14为无符号的值。			

图652

vrroh Rb,Ra,I4

操作	<pre>=> vrolh Rb,Ra,(16-I4)&0xf</pre> <pre>tmp1[31:16] <- Ra[31:16] << I4[3:0]; tmp2[31:16] <- Ra[31:16] >>> (16-I4[3:0]); tmp1[15: 0] <- Ra[15: 0] << I4[3:0]; tmp2[15: 0] <- Ra[15: 0] >>> (16-I4[3:0]); Rb <- tmp1[31:0] tmp2[31:0];</pre> <p>各寄存器以半字矢量形式处理。 Ra向右旋转当前值(14)的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
vrroh Rb,Ra,I4	相当于32bit	—	—
备注			
14为无符号的值。			

vasrh Rc,Ra,Rb

操作	<pre> tmp[4:0] <- Rb[4:0]; tmp[4:0] <- 15, if (Rb > 0x0000_000f); tmp[4:0] <- -15, if (Rb < 0xffff_fff1); Rc[31:16] <- Ra[31:16] >> abs(tmp[4:0]); Rc[31:16] <- Ra[31:16] << abs(tmp[4:0]), if (tmp[4:0] < 0); Rc[15: 0] <- Ra[15: 0] >> abs(tmp[4:0]); Rc[15: 0] <- Ra[15: 0] << abs(tmp[4:0]), if (tmp[4:0] < 0); </pre> <p>Ra与Rc以半字矢量形式处理。</p> <p>将Ra算术右位移Rb所示的位数。Rb在±15内饱和，为负时进行算术左位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vasrh Rc,Ra,Rb	32bit	—	—
备注			

图 653

vaslb Rc,Ra,Rb

操作	<pre> tmp[3:0] <- Rb[3:0]; tmp[3:0] <- 7, if (Rb > 0x0000_0007); tmp[3:0] <- -7, if (Rb < 0xffff_fff9); Rc[31:24] <- Ra[31:24] << abs(tmp[3:0]); Rc[31:24] <- Ra[31:24] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[23:16] <- Ra[23:16] << abs(tmp[3:0]); Rc[23:16] <- Ra[23:16] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[15: 8] <- Ra[15: 8] << abs(tmp[3:0]); Rc[15: 8] <- Ra[15: 8] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[7: 0] <- Ra[7: 0] << abs(tmp[3:0]); Rc[7: 0] <- Ra[7: 0] >> abs(tmp[3:0]), if (tmp[3:0] < 0); </pre> <p>Ra与Rc以字节矢量形式处理。</p> <p>将Ra算术左位移Rb所示的位数。Rb在±7内饱和， 为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vaslb Rc,Ra,Rb	32bit	—	—
备注			

图654

vaslb Rb,Ra,I3

操作	Rb[31:24] <- Ra[31:24] << I3[2:0]; Rb[23:16] <- Ra[23:16] << I3[2:0]; Rb[15: 8] <- Ra[15: 8] << I3[2:0]; Rb[7: 0] <- Ra[7: 0] << I3[2:0]; 各寄存器以字节矢量形式处理。 Ra算术左位移当前值(I3)位数。		
汇编·助记	格式	影响标记	受影响的标记
vaslb Rb,Ra,I3	32bit	—	—
备注			
I3为无符号的值。			

图 655

vasrb Rc,Ra,Rb

操作	<pre> tmp[3:0] <- Rb[3:0]; tmp[3:0] <- 7, if (Rb > 0x0000_0007); tmp[3:0] <- -7, if (Rb < 0xffff_fff9); Rc[31:24] <- Ra[31:24] >> abs(tmp[3:0]); Rc[31:24] <- Ra[31:24] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[23:16] <- Ra[23:16] >> abs(tmp[3:0]); Rc[23:16] <- Ra[23:16] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[15: 8] <- Ra[15: 8] >> abs(tmp[3:0]); Rc[15: 8] <- Ra[15: 8] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[7: 0] <- Ra[7: 0] >> abs(tmp[3:0]); Rc[7: 0] <- Ra[7: 0] << abs(tmp[3:0]), if (tmp[3:0] < 0); Ra与Rc以字节矢量形式处理。 将Ra算术左位移Rb所示的位数。Rb在±7内饱和， 为负时进行算术左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vasrb Rc,Ra,Rb	32bit	—	—
备注			

图656

vasrb Rb,Ra,I3

操作	Rb[31:24] <- Ra[31:24] >> I3[2:0]; Rb[23:16] <- Ra[23:16] >> I3[2:0]; Rb[15: 8] <- Ra[15: 8] >> I3[2:0]; Rb[7: 0] <- Ra[7: 0] >> I3[2:0]; 各寄存器以字节矢量形式处理。 Ra算术右位移当前值(I3)位数。		
汇编·助记	格式	影响标记	受影响的标记
vasrb Rb,Ra,I3	32bit	—	—
备注			
I3为无符号的值。			

图 657

vlsb Rc,Ra,Rb

操作	=> vlsb Rc,Ra,Rb		
	<pre> tmp[3:0] <- Rb[3:0]; tmp[3:0] <- 7, if (Rb > 0x0000_0007); tmp[3:0] <- -7, if (Rb < 0xffff_fff9); Rc[31:24] <- Ra[31:24] << abs(tmp[3:0]); Rc[31:24] <- Ra[31:24] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[23:16] <- Ra[23:16] << abs(tmp[3:0]); Rc[23:16] <- Ra[23:16] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[15: 8] <- Ra[15: 8] << abs(tmp[3:0]); Rc[15: 8] <- Ra[15: 8] >> abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[7: 0] <- Ra[7: 0] << abs(tmp[3:0]); Rc[7: 0] <- Ra[7: 0] >> abs(tmp[3:0]), if (tmp[3:0] < 0); </pre> <p>Ra与Rc以字节矢量形式处理。</p> <p>Ra逻辑左位移Rb所示的位数。Rb在±7内饱和，为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlsb Rc,Ra,Rb	相当于32bit	—	—
备注			
※ 位移量为负时，变为算术右位移。			

图 658

vlsb Rb,Ra,I3

操作	=> vlsb Rb,Ra,I3		
	Rb[31:24] <- Ra[31:24] << I3[2:0]; Rb[23:16] <- Ra[23:16] << I3[2:0]; Rb[15: 8] <- Ra[15: 8] << I3[2:0]; Rb[7: 0] <- Ra[7: 0] << I3[2:0]; 各寄存器以半字矢量形式处理。 Ra逻辑位移当前值(14)的位数。		
汇编·助记	格式	影响标记	受影响的标记
vlsb Rb,Ra,I3		—	—
备注			
I4为无符号的值。			

图 659

vlsrb Rc,Ra,Rb

操作	<pre> tmp[3:0] <- Rb[3:0]; tmp[3:0] <- 7, if (Rb > 0x0000_0007); tmp[3:0] <- -7, if (Rb < 0xffff_fff9); Rc[31:24] <- Ra[31:24] >>> abs(tmp[3:0]); Rc[31:24] <- Ra[31:24] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[23:16] <- Ra[23:16] >>> abs(tmp[3:0]); Rc[23:16] <- Ra[23:16] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[15: 8] <- Ra[15: 8] >>> abs(tmp[3:0]); Rc[15: 8] <- Ra[15: 8] << abs(tmp[3:0]), if (tmp[3:0] < 0); Rc[7: 0] <- Ra[7: 0] >>> abs(tmp[3:0]); Rc[7: 0] <- Ra[7: 0] << abs(tmp[3:0]), if (tmp[3:0] < 0); Ra与Rc以字节矢量形式处理。 Ra逻辑右位移Rb所示的位数。Rb在±7内饱和， 为负时进行逻辑左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vlsrb Rc,Ra,Rb	32bit	—	—
备注			

图660

vlsrb Rb,Ra,I3

操作	Rb[31:24] <- Ra[31:24] >>> I3[2:0]; Rb[23:16] <- Ra[23:16] >>> I3[2:0]; Rb[15: 8] <- Ra[15: 8] >>> I3[2:0]; Rb[7: 0] <- Ra[7: 0] >>> I3[2:0]; 各寄存器以字节矢量形式处理。 Ra逻辑右位移当前值(I3)的位数。		
汇编·助记	格式	影响标记	受影响的标记
vlsrb Rb,Ra,I3	32bit	—	—
备注			
I3为无符号的值。			

图661

vrolb Rc,Ra,Rb

操作	<pre> tmp[31:24] <- Ra[31:24] << Rb[2:0]; tmp2[31:24] <- Ra[31:24] >>> (8 - Rb[2:0]); Rc[31:24] <- tmp[31:24] tmp2[31:24]; tmp[23:16] <- Ra[23:16] << Rb[2:0]; tmp2[23:16] <- Ra[23:16] >>> (8 - Rb[2:0]); Rc[23:16] <- tmp[23:16] tmp2[23:16]; tmp[15: 8] <- Ra[15: 8] << Rb[2:0]; tmp2[15:8] <- Ra[15: 8] >>> (8 - Rb[2:0]); Rc[15: 8] <- tmp[15:8] tmp2[15: 8]; tmp[7: 0] <- Ra[7: 0] << Rb[2:0]; tmp2[7:0] <- Ra[7: 0] >>> (8 - Rb[2:0]); Rc[7: 0] <- tmp[7: 0] tmp2[7: 0]; </pre> <p>Ra和Rc以字节矢量形式处理。</p> <p>Ra向左旋转Rb所示的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
vrolb Rc,Ra,Rb	32bit	—	—
备注			

图662

vrolb Rb,Ra,I3

操作	<pre> tmp1[31:24] <- Ra[31:24] << I3[2:0]; tmp2[31:24] <- Ra[31:24] >>> (8-I3[2:0]); tmp1[23:16] <- Ra[23:16] << I3[2:0]; tmp2[23:16] <- Ra[23:16] >>> (8-I3[2:0]); tmp1[15: 8] <- Ra[15: 8] << I3[2:0]; tmp2[15: 8] <- Ra[15: 8] >>> (8-I3[2:0]); tmp1[7: 0] <- Ra[7: 0] << I3[2:0]; tmp2[7: 0] <- Ra[7: 0] >>> (8-I3[2:0]); Rb <- tmp1[31:0] tmp2[31:0]; </pre> <p>各寄存器以字节矢量形式处理。 Ra向左旋转当前值(I3)的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
vrolb Rb,Ra,I3	32bit	—	—
备注			
I3为无符号的值。			

图663

vrorb Rb,Ra,I3

操作	\Rightarrow vrolb Rb,Ra,(8-I3)&0x7 tmp1[31:24] <- Ra[31:24] << I3[2:0]; tmp2[31:24] <- Ra[31:24] >>> (8-I3[2:0]); tmp1[23:16] <- Ra[23:16] << I3[2:0]; tmp2[23:16] <- Ra[23:16] >>> (8-I3[2:0]); tmp1[15: 8] <- Ra[15: 8] << I3[2:0]; tmp2[15: 8] <- Ra[15: 8] >>> (8-I3[2:0]); tmp1[7: 0] <- Ra[7: 0] << I3[2:0]; tmp2[7: 0] <- Ra[7: 0] >>> (8-I3[2:0]); Rb <- tmp1[31:0] tmp2[31:0]; 各寄存器以字节矢量形式处理。 Ra向右旋转当前值(I3)的位数。		
	汇编·助记	格式	影响标记
vrorb Rb,Ra,I3		—	—
备注			
I3为无符号的值。			

图 664

vsl Mm,Ra,Mn,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] << abs(tmp[5:0]); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >> abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] << abs(tmp[5:0]); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] >> abs(tmp[5:0]), if (tmp[5:0] < 0); Ra <- MLm; Ra以半字矢量形式，Mm和Mn以字矢量形式处理。 Mn算术左位移Rb所示的位数。Rb在±31以内饱和， 为负时进行算术右位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vsl Mm,Ra,Mn,Rb	32bit	—	—
备注			

图 665

vasl Mm,Rb,Mn,I5

操作	$\text{MHm}[31:16]:\text{MLm}[31:16] \leftarrow \text{MHn}[31:16]:\text{MLn}[31:16] \ll 15[4:0];$ $\text{MHm}[15:0]:\text{MLm}[15:0] \leftarrow \text{MHn}[15:0]:\text{MLn}[15:0] \ll 15[4:0];$ $\text{Rb} \leftarrow \text{MLm};$ <p>Mm和Mn以字矢量形式处理，Rb以半字矢量形式处理。 Mn算术左位移当前值(15)的位数。</p>		
汇编·助记	格式	影响标记	受影响的标记
vasl Mm,Rb,Mn,I5	32bit	—	—
备注			
I5为无符号的值。			

图666

vastvw Mm,Ra,Mn,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[62:0] <- sext63(MHn[31:16]:MLn[31:16]) << abs(tmp[5:0]); tmp2[62:0] <- sext63(MHn[31:16]:MLn[31:16]) >> abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); tmp2[62:0] <- sext63(MHn[15:0]:MLn[15:0]) << abs(tmp[5:0]); tmp2[62:0] <- sext63(MHn[15:0]:MLn[15:0]) >> abs(tmp[5:0]); if (tmp[5:0] < 0); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); Ra <- MLm; </pre> <p>Ra以半字矢量形式，Mm和Mn以字矢量形式处理。</p> <p>Mn算术左位移Rb所示的位数。结果不能用带符号的32位数表示时为饱和。为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vastvw Mm,Ra,Mn,Rb	32bit	CFR.OVS	—
备注			

图 667

vaslww Mm,Rb,Mn,I5

操作	<pre> tmp[62:0] <- sext63(MHn[31:16]:MLn[31:16]) << I5[4:0]; MHm[31:16]:MLm[31:16] <- tmp[31:0], MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[62:0] < 0x7fff_ffff_8000_0000[62:0]); tmp[62:0] <- sext63(MHn[15: 0]:MLn[15: 0]) << I5[4:0]; MHm[15: 0]:MLm[15: 0] <- tmp[31:0], MHm[15: 0]:MLm[15: 0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[15: 0]:MLm[15: 0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[62:0] < 0x7fff_ffff_8000_0000[62:0]); Rb <- MLm; Mn和Mm以字矢量形式处理，Rb以半字矢量形式处理。 Mn算术左位移当前值(15)的位数。结果不能用带符号的32位 表示为饱和。 </pre>			
汇编·助记	格式	影响标记	受影响的标记	
vaslww Mm,Rb,Mn,I5	32bit	CFR,OVS	—	
备注				
I5为无符号的值。				

图668

vasr Mm,Ra,Mn,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ff01); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >> abs(tmp[5:0]); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] << abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] >> abs(tmp[5:0]); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] << abs(tmp[5:0]), if (tmp[5:0] < 0); Ra <- MLm; Ra以半字矢量形式，Mm和Mn以字矢量形式处理。 Mn算术右位移Rb所示的位数。Rb在±31以内饱和， 为负时进行算术左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vasr Mm,Ra,Mn,Rb	32bit	—	—
备注			

图 669

vasr Mm,Rb,Mn,I5

操作	<pre> MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >> I5[4:0]; MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] >> I5[4:0]; Rb <- MLm; Mn和Mm以字矢量形式处理，Rb以半字矢量形式处理。 Mn算术右位移当前值(15)的位数。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vasr Mm,Rb,Mn,I5	32bit	—	—
备注			
I5为无符号的值。			

图 670

vasrvw Mm,Ra,Mn,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); tmp2[62:0] <- sext63(MHn[31:16]:MLn[31:16]) >> abs(tmp[5:0]); tmp2[62:0] <- sext63(MHn[31:16]:MLn[31:16]) << abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); tmp2[62:0] <- sext63(MHn[15:0]:MLn[15:0]) >> abs(tmp[5:0]); tmp2[62:0] <- sext63(MHn[15:0]:MLn[15:0]) << abs(tmp[5:0]); if (tmp[5:0] < 0); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[62:0] > 0x0000_0000_7fff_ffff[62:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[62:0] < 0x7fff_ffff_8000_0000[62:0]); Ra <- MLm; Ra以半字矢量形式，Mm和Mn以字矢量形式处理。 Mn算术右位移Rb所示的位数。结果不能用带符号的32位数 表示时为饱和。为负时进行算术左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vasrvw Mm,Ra,Mn,Rb	32bit	CFR.OVS	—
备注			

图671

vsl Mm,Ra,Mn,Rb

操作	<p>=> vsl Mm,Ra,Mn,Rb</p> <pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] << abs(tmp[5:0]); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >> abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0] << abs(tmp[5:0]); MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0] >> abs(tmp[5:0]), if (tmp[5:0] < 0); Ra <- MLm; </pre> <p>Ra以半字矢量形式，Mm和Mn以字矢量形式处理。</p> <p>Mn逻辑左位移Rb所示的位数。Rb在±31以内饱和，为负时进行算术右位移。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsl Mm,Ra,Mn,Rb		-	-
备注			
<p>※ 位移量为负时，变为算术右位移。</p>			

图672

vsl Mm,Rb,Mn,I5

操作	=> vsl Mm,Rb,Mn,I5 MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] << I5[4:0]; MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0] << I5[4:0]; Rb <- MLm; Mn和Mm以字矢量形式处理，Rb以半字矢量形式处理。 Mn逻辑左移当前值(I5)的位数。		
汇编·助记	格式	影响标记	受影响的标记
vsl Mm,Rb,Mn,I5		-	-
备注			
I5为无符号的值。			

图673

vlsr Mm,Ra,Mn,Rb

操作	<pre> tmp[5:0] <- Rb[5:0]; tmp[5:0] <- 31, if (Rb > 0x0000_001f); tmp[5:0] <- -31, if (Rb < 0xffff_ffe1); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >>> abs(tmp[5:0]); MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] << abs(tmp[5:0]), if (tmp[5:0] < 0); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] >>> abs(tmp[5:0]); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] << abs(tmp[5:0]), if (tmp[5:0] < 0); Ra <- MLm; Mn逻辑右位移Rb表示的位数，Rb在±31内饱和， 为负时变为逻辑左位移。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vlsr Mm,Ra,Mn,Rb	32bit	—	—
备注			

图674

图675

vlsr Mm,Rb,Mn,I5

操作	MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] >>> I5[4:0]; MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0] >>> I5[4:0]; Rb <- MLm; Mm和Mn以字矢量形式处理，Rb以半字矢量形式处理。 Mn算术左移当前值(15)位数。		
汇编·助记	格式	影响标记	受影响的标记
vlsr Mm,Rb,Mn,I5	32bit	—	—
备注			
I1为无符号的值。			

vsath Mm,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16]; MHm[31:16]:MLm[31:16] <- 0x0000_7fff, if (MHn[31:16]:MLn[31:16] > 0x0000_7fff[31:0]); MHm[31:16]:MLm[31:16] <- 0xffff_8000, if (MHn[31:16]:MLn[31:16] < 0xffff_8000[31:0]); Rb[31:16] <- MLm[31:16]; MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0]; MHm[15:0]:MLm[15:0] <- 0x0000_7fff, if (MHn[15:0]:MLn[15:0] > 0x0000_7fff[31:0]); MHm[15:0]:MLm[15:0] <- 0xffff_8000, if (MHn[15:0]:MLn[15:0] < 0xffff_8000[31:0]); Rb[15:0] <- MLm[15:0]; Mn和Mm以字矢量形式处理，Rb以半字矢量形式处理。 Mn带符号32位值饱和，存储到Rb。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vsath Mm,Rb,Mn	32bit	—	—
备注			

图676

vsath12 Rb,Ra

操作	<p> Rb[15:0] <- Ra[15:0]; Rb[15:0] <- 0x07ff, if (Ra[15:0] > 0x07ff); Rb[15:0] <- 0xf800, if (Ra[15:0] < 0xf800); Rb[31:16] <- Ra[31:16]; Rb[31:16] <- 0x07ff, if (Ra[31:16] > 0x07ff); Rb[31:16] <- 0xf800, if (Ra[31:16] < 0xf800); </p> <p>各寄存器以半字矢量形式处理。</p> <p>Ra带符号12位值饱和和存储到Rb。</p>		
汇编·助记	格式	影响标记	受影响的标记
vsath12 Rb,Ra	32bit	—	—
备注			

图677

vsath9 Rb,Ra

操作	Rb[15: 0] <- Ra[15: 0]; Rb[15: 0] <- 0x00ff, if (Ra[15: 0] > 0x00ff); Rb[15: 0] <- 0xff00, if (Ra[15: 0] < 0xff00); Rb[31:16] <- Ra[31:16]; Rb[31:16] <- 0x00ff, if (Ra[31:16] > 0x00ff); Rb[31:16] <- 0xff00, if (Ra[31:16] < 0xff00); 各寄存器以半字矢量形式处理。 Ra带符号9位值饱和存储到Rb。		
汇编·助记	格式	影响标记	受影响的标记
vsath9 Rb,Ra	32bit	—	—
备注			

图678

vsath8 Rb,Ra

操作	Rb[15: 0] <- Ra[15: 0]; Rb[15: 0] <- 0x007f, if (Ra[15: 0] > 0x007f); Rb[15: 0] <- 0xff80, if (Ra[15: 0] < 0xff80); Rb[31:16] <- Ra[31:16]; Rb[31:16] <- 0x007f, if (Ra[31:16] > 0x007f); Rb[31:16] <- 0xff80, if (Ra[31:16] < 0xff80);		
各寄存器以半字矢量形式处理。			
Ra带符号8位值饱和存储到Rb。			
汇编·助记	格式	影响标记	受影响的标记
vsath8 Rb,Ra	32bit	—	—
备注			

图679

vsath8u Rb,Ra

操作	Rb[15: 0] <- Ra[15: 0]; Rb[15: 0] <- 0x00ff, if (Ra[15: 0] > 0x00ff); Rb[15: 0] <- 0x0000, if (Ra[15: 0] < 0x0000); Rb[31:16] <- Ra[31:16]; Rb[31:16] <- 0x00ff, if (Ra[31:16] > 0x00ff); Rb[31:16] <- 0x0000, if (Ra[31:16] < 0x0000);		
各寄存器以半字矢量形式处理。			
Ra不带符号8位值饱和存储到Rb。			
汇编·助记	格式	影响标记	受影响的标记
vsath8u Rb,Ra	32bit	—	—
备注			

图680

vrndvh Rb,Mn

操作	<pre> tmp[17:0] <- sext18(MHn[31:16]:MLn[31:31])+sext18(1); Rb[31:16] <- tmp[16:1]; Rb[31:16] <- 0x7fff, CFR.OVS <- 1, if (tmp[17:1] > 0x0_7fff[16:0]); tmp[17:0] <- sext18(MHn[15:0]:MLn[15:15])+sext18(1); Rb[15: 0] <- tmp[16:1]; Rb[15: 0] <- 0x7fff, CFR.OVS <- 1, if (tmp[17:1] > 0x0_7fff[16:0]); </pre> <p>mn以字矢量形式处理。Rb以半字矢量形式处理。</p> <p>进从32位值到16位值的四舍五入运算，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vrndvh Rb,Mn	32bit	CFR.OVS	—
备注			
不会发生负的溢出。			

图681

图682

vabssumb Rc,Ra,Rb

操作	$Rc \leftarrow Rb + \text{uext}(\text{abs}(\text{VC3:Ra}[31:24]))$ $+ \text{uext}(\text{abs}(\text{VC2:Ra}[23:16]))$ $+ \text{uext}(\text{abs}(\text{VC1:Ra}[15: 8]))$ $+ \text{uext}(\text{abs}(\text{VC0:Ra}[7: 0]));$		
	<p>用于图像处理的动作预测的差分绝对值和与vasubb组合使用。</p> <p>Ra以字节矢量形式处理。</p> <p>Ra的4个8位值的绝对值相加，再与Ra相加。Ra的4个8位值的符号位分别存储到VC0~VC3。</p>		
汇编·助记	格式	影响标记	受影响的标记
vabssumb Rc,Ra,Rb	32bit	—	VC0,VC1,VC2,VC3
备注			

图683

vmul Mm,Rc,Ra,Rb

操作	$\text{MHm}[31:16]:\text{MLm}[31:16] \leftarrow \text{smul}(\text{Ra}[31:16],\text{Rb}[31:16]);$ $\text{Rc}[31:16] \leftarrow \text{MLm}[31:16], \text{if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$ $\text{MHm}[15: 0]:\text{MLm}[15: 0] \leftarrow \text{smul}(\text{Ra}[15: 0],\text{Rb}[15: 0]);$ $\text{Rc}[15: 0] \leftarrow \text{MLm}[15: 0], \text{if } (\text{reg}\#(\text{Rc}) \neq 30 \ \&\& \ \text{reg}\#(\text{Rc}) \neq 31);$		
	<p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。</p> <p>Ra和Rb整数相乘(SIMD straight)结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vmul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 684

vxmul Mm,Rc,Ra,Rb

操作	MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[15: 0]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- smul(Ra[15: 0],Rb[31:16]); Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb整数相乘 (SIMD cross) 结果存储到Mm和Rc。		
汇编·助记	格式	影响标记	受影响的标记
vxmul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 685

vhmul Mm,Rc,Ra,Rb

操作	MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- smul(Ra[15: 0],Rb[31:16]); Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb整数相乘 (SIMD high) 结果存储到Mm和Rc。		
汇编·助记	格式	影响标记	受影响的标记
vhmul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图686

vlmul Mm,Rc,Ra,Rb

操作	MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[15: 0]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- smul(Ra[15: 0],Rb[15: 0]); Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31);		
	Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。		
	Ra和Rb整数相乘 (SIMD low) 结果存储到Mm和Rc。		
汇编·助记	格式	影响标记	受影响的标记
vlmul Mm,Rc,Ra,Rb	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

vfmulh Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb整数相乘(SIMD straight)结果存储到Mm和Rc。 结果不能用带符号的32位表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vfmulh Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图687

vxfmulh Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]) MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]) Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]) MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]) Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb整数相乘(SIMD cross)结果存储到Mm和Rc。 结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmulh Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 688

vhfmulh Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)从32位到16位四舍五入结果存到Mm和Rc，结果不能用带符号的16位表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vhfmulh Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP=1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。			

图 689

vfmulh Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] > 0x1_8000_0000[32:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)结果存储到Mm和Rc。 结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmulh Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图690

vfmulhr Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)从32位到16位四舍五入 结果存到Mm和Rc，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmulhr Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

图691

图692

vxfmulhr Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_fff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_fff, CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)从32位到16位四舍五入 结果存到Mm和Rc。结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmulhr Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

图 693

vhfmulhr Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[32:0]) + sext34(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)从32位到16位四舍五入 结果存到Mm和Rc，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmulhr Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。			

图694

vlfmulhr Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[31:0]) + sext34(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(tmp[31:0]) + sext34(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7fff_0000, CFR.OVS <- 1, if (tmp2[33:16] > 0x0_7fff[17:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:16] < 0x3_8000[17:0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)从32位到16位四舍五入 结果存到Mm和Rc，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlfmulhr Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

vfmulw Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext133(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000); Rc[31:0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext133(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Ra、Rb、以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight) 结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vfmulw Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果中，只有一个存储到Rc，另一个相乘结果，请从vunpk2Rb、Mn等中取出。 Rc指定为r30或r31时，不能写入Rc。			

图 695

图 696

vxfmulw Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_fff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_fff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); </pre> <p>Ra、Rb、以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross) 结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmulw Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果中，只有一个存储到Rc、另一个相乘结果， 请从vunpk2 Rb. Mn等中取出。 Rc指定为r30或r31时，不能写入Rc。			

图697

vhfmulw Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7fff_ffff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Ra、Rb、以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)结果 存储到Mm和Rc，结果不能用带符号的32位表示时饱和。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vhfmulw Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果中，只有一个存储到Rc、另一个相乘结果，请从vunpk2Rb、Mn等中取出。 Rc指定为r30或r31时，不能写入Rc。			

图698

vifmulw Mm,Rc,Ra,Rb

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)) tmp[32:0] <- 0x0_fff_fff, CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)) MHm[31:16]:MLm[31:16] <- tmp[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); Rc[31:0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)) tmp[32:0] <- 0xffff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)) MHm[15:0]:MLm[15:0] <- tmp[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[32:0] > 0x0_7ff_fff[32:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[32:0] < 0x1_8000_0000[32:0]); </pre> <p>Ra、Rb、以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)结果 存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vifmulw Mm,Rc,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果中，只有一个存储到Rc、另一个相乘结果， 请从vunpk2Rb、Mn等中取出。			
Rc指定为r30或r31，不能写入Rc。			

图 699

vpfmulhww Mm,Rc:Rc+1,Ra,Rb

操作	<pre> tmp[48: 0] <- sext49(smul(Ra,Rb[31:16]))<<FXP_SHIFT; tmp[48: 0] <- 0x0_7ff_fff_fff[48:0], CFR.OVS <- 1, if ((Ra[31: 0] == 0x8000_0000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[48: 0] <- 0x0_fff_fff_fff[48:0], CFR.OVS <- 1, if ((Ra[31: 0] == 0x8000_0000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); MHm[31:16]:MLm[31:16] <- tmp[47:16]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[48:16] > 0x0_7ff_fff[32:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[48:16] < 0x1_8000_0000[32:0]); Rc <- MHm[31:16]:MLm[31:16], if (reg#(Rc) & 0b11110 != 0b11110); tmp[48: 0] <- sext49(smul(Ra,Rb[15: 0])) << FXP_SHIFT; tmp[48: 0] <- 0x0_7ff_fff_fff[48:0], CFR.OVS <- 1, if ((Ra[31: 0] == 0x8000_0000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[48: 0] <- 0x0_fff_fff_fff[48:0], CFR.OVS <- 1, if ((Ra[31: 0] == 0x8000_0000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); MHm[15: 0]:MLm[15: 0] <- tmp[47:16]; MHm[15: 0]:MLm[15: 0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp[48:16] > 0x0_7ff_fff[32:0]); MHm[15: 0]:MLm[15: 0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp[48:16] < 0x1_8000_0000[32:0]); Rc+1 <- MHm[15: 0]:MLm[15: 0], if (reg#(Rc+1) & 0b11110 != 0b11110); </pre> <p>Ra、Rb、以半字矢量形式处理，Mm以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD pair)结果 存储到Mm和Rc和Rc+1，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vpfmulhww Mm,Rc:Rc+1,Ra,Rb	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31，不能写入Rc。			

图 700

vmac Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[31:16]) +MHn[31:16]:MLn[31:16]; Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15:0]:MLm[15:0] <- smul(Ra[15:0],Rb[15:0]) +MHn[15:0]:MLn[15:0]; Rc[15:0] <- MLm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD straight)，与Mn相加，结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vmac Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31，不能写入Rc。			

图701

vmac M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) + smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15:0]:ML0[15:0] <- sext32(Rx[15:0]) + smul(Ra[15:0],Rb[15:0]); Rc[15:0] <- ML0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc、Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD straight)，与Rx相加，结果存储到M0和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vmac M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 702

vxmacc Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[15: 0]) +MHn[31:16]:MLn[31:16]; Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- smul(Ra[15: 0],Rb[31:16]) +MHn[15: 0]:MLn[15: 0]; Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。</p> <p>Ra和Rb整数相乘 (SIMD cross)，与Mn相加，结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxmacc Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 703

vxmlac M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) + smul(Ra[31:16],Rb[15:0]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15:0]:ML0[15:0] <- sext32(Rx[15:0]) + smul(Ra[15:0],Rb[31:16]); Rc[15:0] <- ML0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc、Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD cross)，与Rx相加，结果存储到M0和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxmlac M0,Rc,Ra,Rb,Rx	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图704

vhmac Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[31:16]) +MHn[31:16]:MLn[31:16]; Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15:0]:MLm[15:0] <- smul(Ra[15:0],Rb[31:16]) +MHn[15:0]:MLn[15:0]; Rc[15:0] <- MLm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD high)，与Mn相加，结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhmac Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 705

vhmac M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) + smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15:0]:ML0[15:0] <- sext32(Rx[15:0]) + smul(Ra[15:0],Rb[31:16]); Rc[15:0] <- ML0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc、Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD high)，与Rx相加，结果存储到M0和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhmac M0,Rc,Ra,Rb,Rx	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 706

vlmac Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- smul(Ra[31:16],Rb[15: 0]) +MHn[31:16]:MLn[31:16]; Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- smul(Ra[15: 0],Rb[15: 0]) +MHn[15: 0]:MLn[15: 0]; Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra、Rb、Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD low)，与Mn相加，结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vlmac Mm,Rc,Ra,Rb,Mn	32bit	—	—
备注			
Rc指定为r30或r31时，不能写入Rc。			

图707

vlmac M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) + smul(Ra[31:16],Rb[15: 0]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15: 0]:ML0[15: 0] <- sext32(Rx[15: 0]) + smul(Ra[15: 0],Rb[15: 0]); Rc[15: 0] <- ML0[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra, Rb, Rc, Rx以半字矢量形式处理, M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD low), 与Mn相加, 结果存储到Mm和Rc。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vlmac M0,Rc,Ra,Rb,Rx	32bit	—	—
备注			
Rc指定为r30或r31时, 不能写入Rc。			

vfmach Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[31:16]:MLn[31:16])+sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext34(smul(Ra[15:0],Rb[15:0])) << FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000), && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[15:0]:MLn[15:0])+ sext34(tmp[32:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight), 与Mn相加, 结果存储到Mm和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmach Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图 708

vfmach M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000 && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000 && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) +sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000 && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000 && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) + sext34(tmp[32:0]) MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc, Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)，与Rx相加， 结果存储到M0和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmach M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 709

vxfmach Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15:0])) << FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[31:16]:MLm[31:16]) + sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[31:16])) << FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[15:0]:MLm[15:0]) + sext34(tmp[31:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross), 与Mn相加, 结果存储到Mm和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmach Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图710

vxfmach M0,Rc,Ra,Rb,Rx

```

tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT;
tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1,
    if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000)
        && (PSR.FXP == 0));
tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1,
    if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000)
        && (PSR.FXP == 1));
tmp2[33:0] <- (sext34(Rx[31:16])<<16) + sext34(tmp[32:0]);
MH0[31:16]:ML0[31:16] <- tmp2[31:0];
MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1,
    if (tmp2[33:0] > 0x0_7ff_fff[33:0]);
MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1,
    if (tmp2[33:0] < 0x3_8000_0000[33:0]);
Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31);
tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT;
tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1,
    if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000)
        && (PSR.FXP == 0));
tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1,
    if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000)
        && (PSR.FXP == 1));
tmp2[33:0] <- (sext34(Rx[15:0])<<16) + sext34(tmp[32:0]);
MH0[15:0]:ML0[15:0] <- tmp2[31:0];
MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1,
    if (tmp2[33:0] > 0x0_7ff_fff[33:0]);
MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1,
    if (tmp2[33:0] < 0x3_8000_0000[33:0]);
Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31);

```

Ra, Rb, Rc, Rx以半字矢量形式处理，M0以字矢量形式处理。
Ra和Rb以固定小数点相乘(SIMD cross)，与Rx相加，
结果存储到M0和Rc。结果不能用带符号的32位表示时饱和。

操作

汇编·助记	格式	影响标记	受影响的标记
vxfmach M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图711

vhfmach Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x_0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x_0_ffff_ffff[32:0], if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[31:16]:MLn[31:16])+ sext34(tmp[31:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; mp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); mp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); mp2[33:0] <- sext34(MHn[15:0]:MLn[15:0])+ sext34(tmp[32:0]); Hm[15:0]:MLm[15:0] <- tmp2[31:0]; Hm[15:0]:MLm[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); Hm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); c[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra, Rb, Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high), 与Mn相加, 结果存储到Mm和Rc。结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmach Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图712

vfmach M0,Rc,Ra,Rb,Rx

操作:	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) + sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext33(Rx[15:0])<<16) + sext33(tmp[32:0]); MH0[16:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7fff_ffff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)与Rx相加， 结果存储到M0和Rc，结果不能用带符号的32位表示时 饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmach M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图713

vifmach Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[31:16]:MLn[31:16])+ sext34(tmp[31:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7fff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ffff_ffff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[15:0]:MLn[15:0])+ sext34(tmp[31:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_ffff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)与Mn相加， 结果存储到Mm和Rc，结果不能用带符号的32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vifmach Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 714

vifmach M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) +sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) + sext34(tmp[32:0]); MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)与Rr相加， 结果存储到M0和Rc，结果不能用带符号的32位表示 时饱和。</p>		
	汇编·助记	格式	影响标记
vifmach M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图715

vfmachr Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&R b[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_ff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&R b[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + sext35(MHm[31:16]:MLn[31:16]) + sext35(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 &&R b[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 &&R b[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + sext35(MHm[15: 0]:MLn[15: 0]) + sext35(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18: 0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1); Rc[15: 0] <- MHm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)与Mn相加，结果从32位到16位四舍五入，结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmachr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

图716

vfmachr M0,Rc,Ra,Rb,Rx

操作	<pre> tmp32[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp32[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp32[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- sext35(tmp32[32:0]) + (sext35(Rx[31:16])<<16) + sext35(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1, If (tmp2[34:16] > 0x0_7ff[18:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[34:16] < 0x7_8000[18:0]); MHm[31:16]:MLm[31:16] <- UD, If (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], If (reg#(Rc) != 30 && reg#(Rc) != 31); tmp32[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp32[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp32[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- sext35(tmp32[32:0]) + (sext35(Rx[15:0])<<16) + sext35(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1, If (tmp2[34:16] > 0x0_7ff[18:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[34:16] < 0x7_8000[18:0]); MHm[15:0]:MLm[15:0] <- UD, If (PSR.FXP == 1) Rc[15:0] <- MHm[15:0], If (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)与Rx相加，结果从32位到16位四舍五入，结果存储到M0，结果不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmachr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话，未组装Mm，Rc变为UD。 Rc指定为r30或r31时，不能写入Rc。			

图717

vxfmachr Mm,Rc,Ra,Rb,Mn

```

tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1,
  If ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000)
    && (PSR.FXP == 0));
tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1,
  If ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000)
    && (PSR.FXP == 1));
tmp2[34: 0] <- sext35(tmp[32: 0])
  + sext35(MHn[31:16]:MLn[31:16])
  + sext35(0x0_8000);
MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0;
MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1,
  If (tmp2[34:16] > 0x0_7ff[16:0]);
MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1,
  If (tmp2[34:16] < 0x7_8000[16:0]);
MHm[31:16]:MLm[31:16] <- UD, If (PSR.FXP == 1);
Rc[31:16] <- MHm[31:16], If (reg#(Rc) != 30 && reg#(Rc) != 31);

tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1,
  If ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000)
    && (PSR.FXP == 0));
tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1,
  If ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000)
    && (PSR.FXP == 1));
tmp2[34: 0] <- sext35(tmp[32: 0])
  + sext35(MHn[15: 0]:MLn[15: 0])
  + sext35(0x0_8000);
MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0;
MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1,
  If (tmp2[34:16] > 0x0_7ff[16: 0]);
MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1,
  If (tmp2[34:16] < 0x3_8000[16: 0]);
MHm[15:0]:MLm[15:0] <- UD, If (PSR.FXP == 1)
Rc[15: 0] <- MHm[15: 0], If (reg#(Rc) != 30 && reg#(Rc) != 31);
Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。
Ra和Rb以固定小数点相乘(SIMD cross)与Mn相加，结果
从32位到16位四舍五入，结果存储到Mm和Rc，结果
不能用带符号的32位表示时饱和。
    
```

操作

汇编助记	格式	影响标记	受影响的标记
vxfmachr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP

备注

PSR.FXP=1的话，未组装Mm，Rc变为UD。
Mm的下位16位，经常变为0。
Rc指定为r30或r31时，不能写入Rc。

图718

vxfmachr M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_ff[32:0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ff_ff[32:0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- sext35(tmp[32:0]) + (sext35(Rx[31:16])<<16) + sext35(0x0_8000); Mm[31:15]:MLm[31:16] <- tmp2[31:16]:16'b0; Mm[31:15]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1, If (tmp2[34:16] > 0x0_7ff[18:0]); Mm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[34:16] < 0x7_8000[18:0]); Mm[31:16]:MLm[31:16] <- UD, If (PSR.FXP == 1); Rc[31:16] <- Mm[31:16], If (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_ff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_ff_ff[32:0], CFR.OVS <- 1, If ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[34:0] <- sext36(tmp[32:0]) + (sext36(Rx[15:0])<<16) + sext36(0x0_8000); Mm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; Mm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1, If (tmp2[34:16] > 0x0_7ff[18:0]); Mm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[34:16] < 0x7_8000[18:0]); Mm[15:0]:MLm[15:0] <- UD, If (PSR.FXP == 1); Rc[15:0] <- Mm[15:0], If (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)与Rx相加，结果从32位到16位四舍五入，结果存储到M0和Rc，结果不能用带符号16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmachr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。			

图719

vhfmachr Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smull(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_ff_f[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + sext35(MHm[31:16]:MLn[31:16]) + sext35(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smull(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_ff_f[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + sext35(MHm[15: 0]:MLn[15: 0]) + sext35(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18: 0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1) Rc[15: 0] <- MHm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)与Mn相加，结果从32位到16位四舍五入，结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmachr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

图720

vhfmachr M0,Rc,Ra,Rb,Rx

```

tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1,
  if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 0));
tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1,
  if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 1));
tmp2[34: 0] <- sext36(tmp[32: 0])
  + (sext36(Rb[31:16])<<16)
  + sext36(0x0_8000);
MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0;
MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1,
  if (tmp2[34:16] > 0x0_7ff[18:0]);
MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[34:16] < 0x7_8000[18:0]);
MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1);
Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31);
tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1,
  if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 0));
tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1,
  if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 1));
tmp2[34: 0] <- sext36(tmp[32: 0])
  + (sext36(Ra[15: 0])<<16)
  + sext36(0x0_8000);
MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0;
MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1,
  if (tmp2[34:16] > 0x0_7ff[18:0]);
MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[34:16] < 0x7_8000[18:0]);
MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1)
Rc[15: 0] <- MHm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31);

```

操作

Ra,Rb,Rc,Rx以半字矢量形式处理，M0以字矢量形式处理。
Ra和Rb以固定小数点相乘(SIMD high)与Rx相加，结果从32位到16位四舍五入，结果存储到M0和Rc，结果不能用带符号16位表示时饱和。

汇编·助记	格式	影响标记	受影响的标记
vhfmachr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。			

图721

vlfmchr Mm,Rc,Ra,Rb,Mn

```

tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1,
  if ((Ra[31:16] == 0x8000 &&Rb[15: 0] == 0x8000)
      && (PSR.FXP == 0));
tmp[32: 0] <- 0x0_ff_ff[32: 0], CFR.OVS <- 1,
  if ((Ra[31:16] == 0x8000 &&Rb[15: 0] == 0x8000)
      && (PSR.FXP == 1));
tmp2[34: 0] <- sext35(tmp[32: 0])
  + sext35(MHn[31:16]:MLn[31:16])
  + sext35(0x0_8000);
MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0;
MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1,
  if (tmp2[34:16] > 0x0_7ff[18:0]);
MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[34:16] < 0x7_8000[18:0]);
MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1);
Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31);
tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[15: 0]))<<FXP_SHIFT;
tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1,
  if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000)
      && (PSR.FXP == 0)); CFR.OVS <- 1,
tmp[32: 0] <- 0x0_ff_ff[32: 0],
  if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000)
      && (PSR.FXP == 1));
tmp2[34: 0] <- sext35(tmp[32: 0])
  + sext35(MHn[15: 0]:MLn[15: 0])
  + sext35(0x0_8000);
MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0;
MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1,
  if (tmp2[34:16] > 0x0_7ff[18: 0]);
MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[34:16] < 0x7_8000[18: 0]);
MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1)
Rc[15: 0] <- MHm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31);
Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。
Ra和Rb以固定小数点相乘(SIMD low)与Mn相加, 结果
从32位到16位四舍五入, 结果存储到Mm和Rc, 结果
不能用带符号的32位表示时饱和。
    
```

操作

汇编·助记	格式	影响标记	受影响的标记
vlfmchr Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP

备注

PSR.FXP==1的话, 未组装Mm, Rc变为UD。
Mm的下位16位, 经常变为0。
Rc指定为r30或r31时, 不能写入Rc。

图722

vfmachr M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32: 0] <- sext33(snu(Ra[31:16],Rb[16: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[16: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_ff_f[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[16: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + (sext35(Rx[31:16])<<16) + sext35(0x0_8000); MHm[31:16]:MLm[31:16] <- tmp2[31:16]:16'b0; MHm[31:16]:MLm[31:16] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MHm[31:16]:MLm[31:16] <- UD, if (PSR.FXP == 1); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(snu(Ra[16: 0],Rb[16: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_ff[32: 0], CFR.OVS <- 1, if ((Ra[16: 0] == 0x8000 && Rb[16: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_ff_f[32: 0], CFR.OVS <- 1, if ((Ra[16: 0] == 0x8000 && Rb[16: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[34: 0] <- sext35(tmp[32: 0]) + (sext35(Rx[16: 0])<<16) + sext35(0x0_8000); MHm[15:0]:MLm[15:0] <- tmp2[31:16]:16'b0; MHm[15:0]:MLm[15:0] <- 0x7ff_0000, CFR.OVS <- 1, if (tmp2[34:16] > 0x0_7ff[18:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[34:16] < 0x7_8000[18:0]); MHm[15:0]:MLm[15:0] <- UD, if (PSR.FXP == 1) Rc[15: 0] <- MHm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)与Mn相加，结果 从32位到16位四舍五入，结果存储到M0和Rc，结果 不能用带符号的16位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmachr M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
<p>PSR.FXP==1的话，未组装Mm，Rc变为UD。 Mm的下位16位，经常变为0。 Rc指定为r30或r31时，不能写入Rc。</p>			

图723

vfmacw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 &&Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[31:16]:MLn[31:16]) + sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[15: 0]:MLn[15: 0]) + sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)与Mn相加， 结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmacw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果，只有一个存储到Rc，另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。			

图724

vxfmacw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[31:16]:MLn[31:16]) + sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[15: 0]:MLn[15: 0]) + sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)与Mn相加， 结果存储到Mm和Rc，结果不能用带符号的32位表示时饱和。</p>			
汇编·助记	格式	影响标记	受影响的标记	
vxfmacw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP	
备注				
SIMD的2个相乘结果，只有一个存储到Rc，另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。				

图725

vhfmacw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[31:16]:MLn[31:16]) + sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[15: 0]:MLn[16: 0]) + sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)与Mn相加, 结果存储到Mm和Rc, 结果不能用带符号32位表示时饱和。</p>		
汇编助记	格式	影响标记	受影响的标记
vhfmacw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果, 只有一个存储到Rc, 另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时, 不能写入Rc。			

图726

vlfmacw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, If ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[31:16]:MLn[31:16]) + sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, If (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], If (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, If ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, If ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[15: 0]:MLn[15: 0]) + sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, If (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, If (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)与Mn相加, 结果存储到Mm和Rc, 结果不能用带符号32位表示时饱和。</p>		
汇编助记	格式	影响标记	受影响的标记
vlfmacw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果, 只有一个存储到Rc, 另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时, 不能写入Rc。			

图727

vpmachww Mm,Rc:Rc+1,Ra,Rb,Mn

操作

```

tmp[48:0] <- sext48(emu1(Ra,Rb[31:16]))<<FXP_SHIFT;
tmp[48:0] <- 0x0_7fff_fff[48:0], CFR.OVS <- 1,
  if ((Ra[31:0] == 0x8000_0000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 0));
tmp[48:0] <- 0x0_fff_fff_fff[48:0], CFR.OVS <- 1,
  if ((Ra[31:0] == 0x8000_0000 && Rb[31:16] == 0x8000)
      && (PSR.FXP == 1));
tmp2[33:0] <- sext34(MHn[31:16]:MLn[31:16])
  + sext34(tmp[48:16]);
MHm[31:16]:MLm[31:16] <- tmp2[31:0];
MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1,
  if (tmp2[33:0] > 0x0_7fff_fff[33:0]);
MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[33:0] < 0x3_8000_0000[33:0]);
Rc <- MHm[31:16]:MLm[31:16],
  if (reg#(Rc) & 0b11110 != 0b11110);
tmp[48:0] <- sext48(emu1(Ra,Rb[15:0]))<<FXP_SHIFT;
tmp[48:0] <- 0x0_7fff_fff[48:0], CFR.OVS <- 1,
  if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000)
      && (PSR.FXP == 0));
tmp[48:0] <- 0x0_fff_fff_fff[48:0], CFR.OVS <- 1,
  if ((Ra[31:0] == 0x8000_0000 && Rb[15:0] == 0x8000)
      && (PSR.FXP == 1));
tmp2[33:0] <- sext34(MHn[15:0]:MLn[15:0])
  + sext34(tmp[48:16]);
MHm[15:0]:MLm[15:0] <- tmp2[31:0];
MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1,
  if (tmp2[33:0] > 0x0_7fff_fff[33:0]);
MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1,
  if (tmp2[33:0] < 0x3_8000_0000[33:0]);
Rc+1 <- MHm[15:0]:MLm[15:0],
  if (reg#(Rc+1) & 0b11110 != 0b11110);
    
```

Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。
Ra和Rb以固定小数点相乘(SIMD pair)与Mn相加，
结果存储到Mm,Rc,Rc+1，结果不能用带符号32位
表示时饱和。

汇编·助记	格式	影响标记	受影响的标记
vpmachww Mm,Rc:Rc+1,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图728

vmsu Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] -smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0]; -smul(Ra[15:0],Rb[15:0]) Rc[15:0] <- MLm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。</p> <p>Ra和Rb整数相乘(SIMD straisht)从Mn减去, 结果存储到Mm和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vmsu Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图729

vmsu M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) - smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15:0]:ML0[15:0] <- sext32(Rx[15:0]) - smul(Ra[15:0],Rb[15:0]); Rc[15:0] <- ML0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra,Rb,Rc,Rx以半字矢量形式处理, M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD straisht)从Rx减去, 结果存储到M0和Rc。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vmsu M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图730

vxmsu Mm,Rc,Ra,Rb,Mn

操作	$MHm[31:16]:MLm[31:16] \leftarrow MHn[31:16]:MLn[31:16]$ $\quad \quad \quad -smul(Ra[31:16],Rb[15:0]);$ $Rc[31:16] \leftarrow MLn[31:16], \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ $MHm[15:0]:MLm[15:0] \leftarrow MHn[15:0]:MLn[15:0]$ $\quad \quad \quad -smul(Ra[15:0],Rb[31:16]);$ $Rc[15:0] \leftarrow MLm[15:0], \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD cross)从Mn减去, 结果存储到Mm和Rc。			
	汇编·助记	格式	影响标记	受影响的标记
vxmsu Mm,Rc,Ra,Rb,Mn		32bit	-	-
备注				
Rc指定为r30或r31时, 不能写入Rc。				

图731

vxmsu M0,Rc,Ra,Rb,Rx

操作	$MH0[31:16]:ML0[31:16] \leftarrow sext32(Rx[31:16])$ $\quad \quad \quad -smul(Ra[31:16],Rb[15:0]);$ $Rc[31:16] \leftarrow ML0[31:16], \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ $MH0[15:0]:ML0[15:0] \leftarrow sext32(Rx[15:0])$ $\quad \quad \quad -smul(Ra[15:0],Rb[31:16]);$ $Rc[15:0] \leftarrow ML0[15:0], \text{ if } (reg\#(Rc) \neq 30 \ \&\& \ reg\#(Rc) \neq 31);$ Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。			
	汇编·助记	格式	影响标记	受影响的标记
vxmsu M0,Rc,Ra,Rb,Rx		32bit	-	-
备注				
Rc指定为r30或r31时, 不能写入Rc。				

图732

vhmsu Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] -emul(Ra[31:16],Rb[31:16]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15:0]:MLm[15:0] <- MHn[15:0]:MLn[15:0] -emul(Ra[15:0],Rb[31:16]); Rc[15:0] <- MLm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD high)从Mn减去, 结果存储到Mm和Rc。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vhmsu Mm,Rc,Ra,Rb,Mn	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图733

vhmsu M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) - smul(Ra[31:16],Rb[31:16]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15:0]:ML0[15:0] <- sext32(Rx[15:0]) - smul(Ra[15:0],Rb[31:16]); Rc[15:0] <- ML0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); Ra,Rb,Rc,Rx以半字矢量形式处理, M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD high)从Rx减去, 结果存储到M0和Rc。 </pre>		
汇编·助记	格式	影响标记	受影响的标记
vhmsu M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时,不能写入Rc。			

图734

vmsu Mm,Rc,Ra,Rb,Mn

操作	<pre> MHm[31:16]:MLm[31:16] <- MHn[31:16]:MLn[31:16] -smul(Ra[31:16],Rb[15: 0]); Rc[31:16] <- MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MHm[15: 0]:MLm[15: 0] <- MHn[15: 0]:MLn[15: 0] -smul(Ra[15: 0],Rb[15: 0]); Rc[15: 0] <- MLm[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理, Mm和Mn以字矢量形式处理。 Ra和Rb整数相乘(SIMD low)从Mn减去, 结果存储到Mm和Rc。</p>			
	汇编助记	格式	影响标记	受影响的标记
vmsu Mm,Rc,Ra,Rb,Mn	32bit	-	-	
备注				
Rc指定为r30或r31时, 不能写入Rc。				

图735

vimsu M0,Rc,Ra,Rb,Rx

操作	<pre> MH0[31:16]:ML0[31:16] <- sext32(Rx[31:16]) - smul(Ra[31:16],Rb[15: 0]); Rc[31:16] <- ML0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); MH0[15: 0]:ML0[15: 0] <- sext32(Rx[15: 0]) - smul(Ra[15: 0],Rb[15: 0]); Rc[15: 0] <- ML0[15: 0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以半字矢量形式处理, M0以字矢量形式处理。 Ra和Rb整数相乘(SIMD low)从Rx减去, 结果存储到M0和Rc。</p>		
汇编·助记	格式	影响标记	受影响的标记
vimsu M0,Rc,Ra,Rb,Rx	32bit	-	-
备注			
Rc指定为r30或r31时，不能写入Rc。			

图736

vfmsuh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[31:16]:MLn[31:16])-sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[15:0]:MLn[15:0])-sext34(tmp[32:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD Straight)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmsuh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图737

vfmsub M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) - sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[32:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[32:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0xffff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) - sext34(tmp[32:0]); MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[32:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[32:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD Straight)从Rx减去， 结果存储到M0和Rc，结果不能用带符号32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmsub M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图738

vxfmsuh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[31:16]:MLn[31:16]) - sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[15:0]:MLn[15:0]) - sext34(tmp[32:0]); MHm[15: 0]:MLm[15: 0] <- tmp2[31:0]; MHm[15: 0]:MLm[15: 0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[15: 0]:MLm[15: 0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmsuh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 739

vxfmsub M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) - sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) - sext34(tmp[32:0]); MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc, Rx以半字矢量形式处理, M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)从Rx减去, 结果存储到M0和Rc, 结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmsuh M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时, 不能写入Rc。			

图740

vhfmsuh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[31:16]:MLn[31:16]) - sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_fff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHn[15:0]:MLn[15:0]) - sext34(tmp[31:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7fff_fff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmsuh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图741

vhfmsuh M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) - sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[31:16]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) - sext34(tmp[32:0]); MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc,Rx以半字矢量形式处理，M0以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)从Rx减去， 结果存储到M0和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmsuh M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图 742

vifmsuh Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32:0] <- sext33(smul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[31:16]:MLm[31:16]) - sext34(tmp[32:0]); MHm[31:16]:MLm[31:16] <- tmp2[31:0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[31:16] <- MHm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(smul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- sext34(MHm[15:0]:MLm[15:0]) - sext34(tmp[32:0]); MHm[15:0]:MLm[15:0] <- tmp2[31:0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MHm[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vifmsuh Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图743

vifmsub M0,Rc,Ra,Rb,Rx

操作	<pre> tmp[32:0] <- sext33(umul(Ra[31:16],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[31:16])<<16) - sext34(tmp[32:0]); MH0[31:16]:ML0[31:16] <- tmp2[31:0]; MH0[31:16]:ML0[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_0_7ff_fff[33:0]); MH0[31:16]:ML0[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[32:0]); Rc[31:16] <- MH0[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32:0] <- sext33(umul(Ra[15:0],Rb[15:0]))<<FXP_SHIFT; tmp[32:0] <- 0x0_7ff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 0)); tmp[32:0] <- 0x0_fff_fff[32:0], CFR.OVS <- 1, if ((Ra[15:0] == 0x8000 && Rb[15:0] == 0x8000) && (PSR.FXP == 1)); tmp2[33:0] <- (sext34(Rx[15:0])<<16) - sext34(tmp[32:0]); MH0[15:0]:ML0[15:0] <- tmp2[31:0]; MH0[15:0]:ML0[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33:0] > 0x0_7ff_fff[33:0]); MH0[15:0]:ML0[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33:0] < 0x3_8000_0000[33:0]); Rc[15:0] <- MH0[15:0], if (reg#(Rc) != 30 && reg#(Rc) != 31); </pre> <p>Ra,Rb,Rc以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vifmsub M0,Rc,Ra,Rb,Rx	32bit	CFR.OVS	PSR.FXP
备注			
Rc指定为r30或r31时，不能写入Rc。			

图744

vfmsuw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[31:16])<<FXP_SHIFT; tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[31:16]:MLn[31:16]) - sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7fff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[15: 0])<<FXP_SHIFT; tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[15: 0]:MLn[15: 0]) - sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7fff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD straight)从Mn减去 结果存储到Mm和Rc，结果不能用带符号32位表示时 饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vfmsuw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果，只有一个存储到Rc， 另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。			

图745

vxfmsuw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(smul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[31:16]:MLn[31:16]) - sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(smul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[15: 0]:MLn[15: 0]) - sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD cross)从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vxfmsuw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果，只有一个存储到Rc， 另一个相乘结果请从vunpk2.Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。			

图746

vhfmsuw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[31:16]:MLn[31:16]) - sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[31:16]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7ff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[31:16] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHn[15: 0]:MLn[15: 0]) - sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7ff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7ff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD high)，从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示 时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vhfmsuw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果，只有一个存储到Rc， 另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。			

图 747

vifmsuw Mm,Rc,Ra,Rb,Mn

操作	<pre> tmp[32: 0] <- sext33(umul(Ra[31:16],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[31:16] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[31:16]:MLn[31:16]) - sext34(tmp[32: 0]); MHm[31:16]:MLm[31:16] <- tmp2[31: 0]; MHm[31:16]:MLm[31:16] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7fff_fff[33: 0]); MHm[31:16]:MLm[31:16] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); Rc[31: 0] <- MHm[31:16]:MLm[31:16], if (reg#(Rc) != 30 && reg#(Rc) != 31); tmp[32: 0] <- sext33(umul(Ra[15: 0],Rb[15: 0]))<<FXP_SHIFT; tmp[32: 0] <- 0x0_7fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 0)); tmp[32: 0] <- 0x0_fff_fff[32: 0], CFR.OVS <- 1, if ((Ra[15: 0] == 0x8000 && Rb[15: 0] == 0x8000) && (PSR.FXP == 1)); tmp2[33: 0] <- sext34(MHm[15: 0]:MLn[15: 0]) - sext34(tmp[32: 0]); MHm[15:0]:MLm[15:0] <- tmp2[31: 0]; MHm[15:0]:MLm[15:0] <- 0x7fff_fff, CFR.OVS <- 1, if (tmp2[33: 0] > 0x0_7fff_fff[33: 0]); MHm[15:0]:MLm[15:0] <- 0x8000_0000, CFR.OVS <- 1, if (tmp2[33: 0] < 0x3_8000_0000[33: 0]); </pre> <p>Ra,Rb以半字矢量形式处理，Mm和Mn以字矢量形式处理。 Ra和Rb以固定小数点相乘(SIMD low)，从Mn减去， 结果存储到Mm和Rc，结果不能用带符号32位表示时饱和。</p>		
汇编·助记	格式	影响标记	受影响的标记
vifmsuw Mm,Rc,Ra,Rb,Mn	32bit	CFR.OVS	PSR.FXP
备注			
SIMD的2个相乘结果，只有一个存储到Rc，另一个相乘结果请从vunpk2 Rb.Mn写出。 Rc指定为r30或r31时，不能写入Rc。			

图748