[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04N 7/50
H03M 7/42

[12] 发明专利说明书

[21] ZL 专利号 99806441.6

[45] 授权公告日 2005 年 4 月 27 日

[11] 授权公告号 CN 1199472C

[22] 申请日 1999.5.14 [21] 申请号 99806441.6
[30] 优先权
[32] 1998.5.18 [33] US [31] 60/085,797
[32] 1999.3.29 [33] US [31] 09/280,437
[86] 国际申请 PCT/US1999/010659 1999.5.14
[87] 国际公布 WO1999/060521 英 1999.11.25
[85] 进入国家阶段日期 2000.11.20
[71] 专利权人 索尼电子有限公司
地址 美国新泽西州
[72] 发明人 M·布布利尔 S·波瑟 S·C·加德雷
审查员 郎亦虹

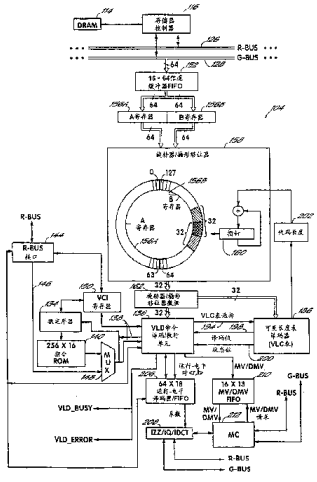
[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 王 岳 王忠忠

权利要求书 4 页 说明书 24 页 附图 7 页

[54] 发明名称 对数字编码视频信号进行译码的可变长度译码器

[57] 摘要

这里公开了一种用于对依从 MPEG-1 和-2 语法的视频位流进行译码的可变长度译码器(VLD)(104)。VLD(104)包括一个微定序器(134)和用于采用一个新颖指令集(214a-214g)控制 MPEG 译码过程的 VLD 命令译码/执行单元(136)。指令集(214a-214g)包括一个用于译码视频数据的命令集和一个流程控制指令集。旋转器/桶形移位器(158)用于使得来自视频位流的预定数目的编码位可用于 VLD(104)和采用 MPEG 标准可变长度代码(VLC)表进行可变长度译码的可变长度表译码器(186)。可变长度表译码器(186)在所有 VLC 表之间共享一个前缀模式匹配方案，并将可变长度代码组织成一系列子表。每个子表对应于一个唯一的前缀模式。通过识别视频数据位流中的领先模式并且并行地访问对应于该领先模式的子表来对可变长度代码进行译码。将运行-长度和幅度电平 DCT 系数码元以压缩形式存储，并由反变换单元(208)根据需要译码。运动矢量也被一直存储到运动补偿单元(212)需要为止。



ISSN1008-4274

1.一种用于对压缩视频数据流进行译码的可变长度译码器（104），其中，数据流包括表示所发送帧的图象区域的多个可变长度编码数据，所述可变长度译码器（104）包括:

存储器（114），用于存储可变长度编码数据;

与存储器相连的命令译码和执行电路(136)，用于从存储器(114)接收所选择的可变长度编码数据;

与命令译码和执行电路（136）相连的定序器（134），用于根据存储在该定序器中的一组指令，向命令译码和执行电路（136）提供命令，以便将可变长度编码数据变换成相应的译码值；以及

与命令译码和执行电路（136）相连的主控制器（112），用于与定序器（134）无关地向命令译码和执行电路（136）提供命令，以控制命令译码和执行电路（136）的译码操作;

与定序器（134）和主控制器（112）相连的多个命令指令寄存器（150），其中，定序器（134）和主控制器（112）读取和写入该命令指令寄存器（150）中的内容，该命令指令寄存器（150）包括那些至少构成了存储在该定序器中的该组指令的一部分的数据.

2.如权利要求 1 所述的可变长度译码器（104），其中，定序器(134)包括用于存储多个指令的存储器(140)和与指令存储器(140)相连的指令译码和控制电路（142），指令译码和控制电路（142）用于对来自存储器（140）的指令译码，并提供命令到命令译码和执行电路（136），以便将可变长度编码数据变换成相应的译码值.

3.如权利要求 1 所述的可变长度译码器（104），其中，所述命令指令寄存器（150）与提供给命令译码和执行电路（136）的指令相对应。

4.如权利要求 1 所述的可变长度译码器（104），还包括与命令译码和执行电路（136）相连的可变长度表译码器（186），用于接收可变长度编码数据并提供相应的译码值到命令译码和执行电路（136）。

5.如权利要求 1 所述的可变长度译码器（104），其中，命令译码和执行电路（136）可用于对符合运动图象专家组语法的可变长度编码数据进行译码.

6.如权利要求 5 所述的可变长度译码器（104），其中，命令译码和执行电路（136）可用于将可变长度编码数据译码成离散余弦变换系数码元，每个离散余弦变换系数码元包括一个运行-长度值和一个幅度电平值。

7.如权利要求 6 所述的可变长度译码器（104），还包括与命令译码和执行电路（136）相连的先进先出存储器和译码器（206），用于将离散余弦变换系数码元存储为压缩的运行-长度和幅度电平对。

8.如权利要求 7 所述的可变长度译码器（104），还包括与先进先出存储器和译码器（206）相连的离散余弦反变换电路（208），其中，先进先出存储器和译码器（206）可用于将运行-长度和幅度电平对解压缩成离散余弦变换系数，以供离散余弦反变换电路（208）在重构所发送帧的图象数据时使用。

9.如权利要求 5 所述的可变长度译码器（104），其中，命令译码和执行电路（136）可用于将可变长度编码数据译码成运动矢量值。

10.如权利要求 9 所述的可变长度译码器（104），还包括与命令译码和执行电路（136）相连的用于存储运动矢量值的先进先出存储器（210）。

11.如权利要求 10 所述的可变长度译码器（104），还包括与先进先出存储器（210）相连的运动补偿电路（212），其中，先进先出存储器（210）可用于在重构所发送帧的图象数据时向运动补偿电路（212）提供运动矢量值。

12.一种用于采用可变长度代码表来对压缩视频数据进行译码的可变长度表译码器（186），其中，压缩视频数据包括表示一发送帧的图象区域的可变长度编码数据，该可变长度表译码器（186）被耦合到存储器（156A，156B）和与该存储器（156A，156B）相联系的移位器电路（164），其中该存储器（156A，156B）存储可变长度编码数据，而该移位器电路（164）使该可变长度编码数据的预定数目的位（162）可见，所述可变长度表译码器（186）包括:

与存储器（156A，156B）和移位器电路（164）相连的模式匹配电路（188），用于识别由移位器电路（164）使其可见的可变长度编码数据（162）中的唯一的前缀模式，每个唯一的前缀模式具有一前缀模式长度;

可变长度代码表数据包括与多个可变长度代码中的每一个相联系的译码值，其中，该可变长度代码表数据在多个子表数据电路中配置，每个子表数据电路与可变长度代码中的唯一的前缀模式相联系，并且和该前缀模式长度无关；

将由移位器电路（164）使其可见的可变长度编码数据（162）连到模式匹配电路（188）和每个子表数据电路的数据路径（196），用于将可变长度编码数据（162）同时加到模式匹配电路（188）和每个子表数据电路；

响应模式匹配电路（188）的控制电路（190,192），用于从与可变长度代码中的唯一的前缀模式以及在唯一的前缀模式之后的可变长度代码中的附加数据相联系的子表数据电路中获得一个译码值（198），其中所述唯一的前缀模式与可变长度编码数据中所识别的前缀模式相匹配，所述附加数据与在所识别的前缀模式之后的可变长度编码数据中的附加数据相匹配。

13. 如权利要求 12 所述的可变长度表译码器（186），其中，每个可变长度代码包括一个相关代码长度，其中，控制电路（190,192）响应模式匹配电路（188），用于从与可变长度代码中的唯一的前缀模式以及在唯一的前缀模式之后的可变长度代码中的附加数据相联系的子表数据电路中获得一个代码长度（202），其中所述唯一的前缀模式与可变长度编码数据中所识别的前缀模式相匹配，所述附加数据与在所识别的前缀模式之后的可变长度编码数据中的附加数据相匹配。

14. 如权利要求 12 所述的可变长度表译码器（186），其中，移位器电路（164）包括多个选择器电路，选择器电路用于有选择地移位存储器（156A，156B）中的可变长度编码数据。

15. 如权利要求 14 所述的可变长度表译码器（186），其中，将多个选择器电路安排在多个选择器级中，其中，每个选择器级可用于以 2 的幂移位存储器（156A，156B）中的可变长度编码数据或不对可变长度编码数据提供移位。

16. 如权利要求 15 所述的可变长度表译码器（186），其中，至少一个选择器级包括比前一选择器级更少的选择器电路。

17. 如权利要求 15 所述的可变长度表译码器（186），其中，每

个选择器级包括比其前一选择器级更少的选择器电路。

18. 如权利要求 14 所述的可变长度表译码器（186），其中，移位器电路（164）响应最后获得的代码值，用于以等于所获得的代码值的位数来移位存储器（156A，156B）中的可变长度编码数据。

19. 一种用于采用可变长度代码表来对压缩视频数据流进行译码的方法，其中，数据流包括表示所发送帧的图象区域的多个可变长度编码数据，所述方法包括下列步骤：

定义一个包括与多个可变长度代码中的每一个相联系的译码值的可变长度代码表；

定义与可变长度代码相联系的多个唯一的前缀模式，每个唯一的前缀模式具有一前缀模式长度；

定义多个子表，其中，每个子表与可变长度代码中的唯一的前缀模式中的一个相联系，并与该前缀模式长度无关，每个子表包括至少一个与唯一的前缀模式以及在唯一的前缀模式之后的可变长度代码中的附加数据相联系的译码值；

识别可变长度编码数据中的与可变长度代码相联系的唯一的前缀模式中的一个；

将可变长度编码数据同时加到多个子表的每一个上；以及

从与可变长度代码中的唯一的前缀模式以及在唯一的前缀模式之后的可变长度代码中的附加数据相联系的子表中获得一个译码值（198），其中所述唯一的前缀模式与可变长度编码数据中所识别的前缀模式相匹配，所述附加数据与在所识别的前缀模式之后的可变长度编码数据中的附加数据相匹配。

20. 如权利要求 19 所述的方法，还包括下列步骤：

将代码长度与多个可变长度代码中的每一个相联系；以及

获得与具有一个前缀模式和在前缀模式之后的附加数据的可变长度代码相联系的代码长度（202），其中，所述前缀模式与可变长度编码数据中所识别的前缀模式相匹配，所述附加数据与在所识别的前缀模式之后的可变长度编码数据中的附加数据相匹配。

对数字编码视频信号进行译码的可变长度译码器

交叉参考

本申请要求在 1998 年 5 月 18 日申请的共同未决的临时申请 No.60/085,797 的申请日，其公开文本在这里作为参考。

发明领域

本发明涉及数字编码视频信号的译码，尤其涉及用于对已经用固定长度值和可变长度码编码的视频数据和控制信息进行译码的译码器。

发明背景

近年来视频信号的数字传输已经被更加广泛地使用，特别是用在消费者电子工业中。数字视频信号传输以及在数字通用盘（DVD）播放器和数字视频广播（DVB）机顶盒应用中的接收的使用方面的增长例如已经导致在图象的发射序列中的图象质量的提高以及更有效地控制视频信号在现有 NTSC 和 PAL 模拟传输系统上的存储、管理和显示的能力。在这些发展的促进下，由国际标准化组织（ISO）特许设立的行业主办的运动图象专家组（MPEG）具有用于数字视频压缩的特定格式，即用于编码视频位流的语法，该语法在两个标准、ISO-11172-2(MPEG-1)和 ISO-13818-2(MPEG-2)中进行了定义。因此，在下面的讨论中，参考 ISO-11172-2(MPEG-1)和 ISO-13818-2(MPEG-2)，以便更详细地说明用于根据这些标准对视频信号进行数字编码的位流语法。因此，这些标准中的每一个在这里都全部作为参考。

由 MPEG-1 和-2 标准定义的位流语法涉及位流中的三种一般类型的信息或数据，即定义位流所必须的控制信息、正确地解压缩和复制图象的发射序列的控制信息、以及编码视频数据。位流控制信息可以识别位流是被分组的视频或音频数据，或者例如识别出位流是一个用 MPEG-1 或-2 标准编码的视频信号。图象控制信息例如可以包括帧水平和垂直尺寸、即每行的图象象素（pels）数以及每帧中的行数、帧频或场频、以及画面高宽比。如同下面将要详细说明的，编码视频数据表示重新生成每一帧或场所必须的 DCT 变换和量化的色度和亮度象素值。

MPEG-1 和-2 标准分别指定被设计为通过去除空间和时间冗余的方法来提高信息密度和编码效率的位流语法．将每个 MPEG 图象分成一系列宏块，该宏块由 16×16 个亮度象素（Y）或四个 8×8 变换块象素的 2×2 阵列组成．每个宏块进一步包括 8×16 个色度象素（U，V）或两个 8×8 象素块的 1×2 阵列．在编码过程中，通过采用对 8×8 象素块的离散余弦变换（DCT）编码、随后进行量化、之字形扫描以及运行零（运行长度）和幅度电平的可变长度编码来去除空间冗余．采用运动补偿预测来去除时间冗余．

对于视频，MPEG 考虑内（I-）帧、预测（P-）帧和双向预测（B-）帧．I-帧是独立编码的，是三种帧类型中编码效率最低的．P-帧比 I-帧的编码效率高，相对于前面编码的 I-或 P-帧进行编码．B-帧是三种帧类型中编码效率最高的，相对于前面和后面的 I-或 P-帧进行编码．在 MPEG 系统中的帧的编码顺序不必与帧的出现顺序相同．位流的首部提供由译码器用来正确地对帧的时间和序列译码以展示一个活动图象所用的信息．

用于对数字发送的视频位流进行译码的典型的视频译码器具有一个用于控制可变长度译码器（VLD）的微控制器或定序器，所述可变长度译码器被设计为对位流进行语法分析，以便采用 MPEG 可变长度代码表（VLC）对量化的 DCT 系数和运动矢量进行译码．反变换处理器用于将每块量化系数值变换成一个表示该块的反之字形的值流，并对这些值进行去量化．将去量化的 DCT 系数传送到执行反 DCT 变换操作的反离散余弦变换（IDCT）处理器，以恢复色度和亮度象素值．然后将这些值与译码后的运动矢量结合起来送到一个运动补偿（MC）处理器，执行 MPEG 解压缩，将 I-、P-和 B-帧变换成全视频帧．

在典型的用于执行依从 MPEG 语法的位流语法分析和译码的 VLD 结构中，将视频位流的预定数目的未译码位存储在一个或多个寄存器中．VLD 从寄存器中提取出这些位中的较小数目的位，将最左边的位总是排列为 VLD 提取的第一位．VLD 然后在一个 MPEG VLC 表中执行表查找，以便对可变长度编码的视频数据进行译码，并获得代码长度．在已经对提取出的位中的可变长度代码进行译码后，VLD 在寄存器中的位上执行 MASK/SHIFT/OR 操作，以重新排列寄存器的最左边位置上的第一未用位.VLC表一般包含在一个或多个大约具有 $2^n \times m$ 个存储单元的 PALS

或 ROM 中，其中“n”表示在每个 VLC 表中的最大可能可变代码长度，“m”表示唯一 VLC 表的数目.

本领域普通技术人员应该理解，在特定 VLD 结构中的译码过程之后排列未使用的位所需的 SHIFT/MASK/OR 操作将显著地影响 VLD 的整个译码速度. 由于这些操作中的每一个可能都需要一个或多个周期，所以当 VLD 需要多个周期来对每个 DCT 系数码元（即，每个运行长度和幅度电平对）进行译码并且然后重新排列未使用的位时，VLD 的译码效率显著降低. 另外，由于每个唯一的 VLC 表的每个可变长度代码都存储在一个单独的存储单元中，所以特定 VLD 结构中的 VLC 表结构增加 VLD 的成本和复杂性.

因此，需要一种对已经根据 MPEG-1 或-2 标准编码的可变长度 DCT 系数和运动矢量进行高效译码的 VLD. 还需要一种使得译码各种 MPEG 可变长度代码所需的存储量最小的 VLD. 还需要一种能够根据预定的指令集从微定序器接收指令并进一步从主控制器接收指令的 VLD.

US-A-5,502,493 公开了一种用于 MPEG 信号的译码器，包括可变长度译码器.

1995 年 7 月的信息技术-移动图象和相关音频信息的一般编码: 系统的 ITI-T 推荐标准 H.222.0（与 ISO/IEC 国际标准 13818-1 相同）公开了一种在译码之前对音频和视频数据进行多路分解的系统.

US-A-5,604,499 公开了一种用于采用可变长度代码表对压缩视频数据进行译码的可变长度表译码器.

发明概述

实施本发明的可变长度视频译码器特别适合于对依从 MPEG-1 和-2 语法的视频位流进行译码. 将视频译码器设计为一个用于对单个时间片的宏块和块层进行译码以产生 DCT 系数值和运动矢量的单个事件每周期时间片语法分析机（single event per cycle slice parsing engine）.

视频译码器包含一个微定序器，微定序器与 VLD 命令译码/执行单元相连系，以便根据 MPEG 标准控制可变长度译码过程. 在译码过程中，微定序器或者向 VLD 命令译码/提取单元发出命令来执行可变长度译码，或者控制通过存储在指令 ROM 中的指令集提供的程序流. 视频译码器还能够从一个负责上层语法分析和控制整个译码过程以便重构译

码的图象序列的 RISC CPU 接收译码指令.

将编码视频数据存储在 DRAM 存储器中，并通过一个信道缓冲器 FIFO 使其可用于视频译码器. 依据本发明的一个方面，通过使用旋转器/桶形移位器和指针寄存器，使得预定数目的编码视频数据位对于视频译码器和可变长度表译码器来说是可见的. 桶形移位器和指针寄存器使得从指针地址到指针地址 + 31 的位作为旋转器/桶形移位器数据对于视频译码器和可变长度表译码器来说都是可见的. 视频译码器负

责对旋转器/桶形移位器数据中的可变长度代码进行译码，以获得每个时间片的所需 DCT 系数和运动矢量。在对可变长度代码译码之后，旋转器/桶形移位器的指针寄存器递增，准备下一个译码周期。

依据本发明的另一个方面，提供了一种使得视频译码器能访问每个 MPEG VLC 表、以获得所需的译码值的新颖的方案。根据每个表中所标识的唯一的前缀模式的限定，每个 MPEG VLC 表被分成一系列子表。在可变长度译码过程中，将提取出的 32 位的旋转器/桶形移位器数据运用到可变长度表译码器中的一个模式匹配逻辑和 MUX 控制上，以识别旋转器/桶形移位器数据中的唯一的前缀模式。将前缀模式后面的位并行地加到每个 MPEG VLC 表中的所有子表上。在已经对可变长度编码数据进行了译码之后，可变长度表译码器提供译码值和一个有效代码状态位。可变长度表译码器还向旋转器/桶形移位器的指针寄存器提供一个代码长度信号，以该代码长度递增该指针寄存器。

依据本发明的又一个方面，将译码的 DCT 系数作为压缩的运行长度和幅度电平对存储在运行-电平译码器/FIFO 中。根据反变换单元的需要，运行-电平译码器/FIFO 将运行长度和幅度电平对解压缩成 DCT 系数。这允许将对霍夫曼编码的可变长度对进行的译码与对以前译码的运行-电平对进行的运行-电平译码并行地进行。将运动矢量存储在 mv/dmv FIFO 中，直到运动补偿单元需要。

从下面的附图及其说明中，本发明的上述和其他方面、目的和优点将变得显而易见。

附图简要说明

附图被包含在说明书中，并构成说明书的一部分，附图显示了本发明的实施例，并与上面给出的本发明的总体说明以及下面给出的实施例的详细说明一起用于解释本发明的原理。

图 1 是依据本发明原理的用于执行数字音频/视频信号数据解压缩和输出的视频/音频译码器的方框图；

图 2 是图 1 所示的译码系统中的视频译码器的方框图，包括用于对数字编码视频数据和控制信息进行译码的可变长度译码器或 VLD；

图 2A 是图 2 所示的视频译码器的微定序器和 VLD 命令/译码执行单元的方框图；

图 3 是用于将 32 位增量的编码视频数据提供到图 2 的可变长度译

码器（VLD）命令译码/执行单元和可变长度表译码器的旋转器/桶形移位器电路的示意图；

图 4 是在由图 2 的可变长度译码器（VLD）命令译码/执行单元访问的“macroblock_address_increment”可变长度代码（VLC）表的模式匹配结构的表示图；

图 5 是由图 2 的可变长度译码器（VLD）在视频数据解压缩过程中访问可变长度代码（VLC）时使用的译码逻辑电路的方框图；

图 6A 是用于 TBIT 指令的微定序器指令格式；

图 6B 是用于 ICMD、CMDI 指令的微定序器指令格式；

图 6C 是用于 COMP 指令的微定序器指令格式；

图 6D 是用于 BRANCH 指令的微定序器指令格式；

图 6E 是用于 SETF 指令的微定序器指令格式；

图 6F 是用于 INCM 指令的微定序器指令格式；以及

图 6G 是用于 HALT 指令的微定序器指令格式。

最佳实施例的详细说明

参考附图，特别参考图 1，图 1 显示了一个译码系统 100，包括用于执行数字编码视频信号和解压缩的视频图象的显示的音频/视频译码的各种功能模块 102 至 112。优选地，译码系统 100 被构造为一个专用集成电路（ASIC），用于例如数字通用盘（DVD）和数字视频广播（DVB）机顶盒（STB）应用中的数字音频/视频接收。应该理解，图 1 中所示的功能单元只是示例性的，而在实际的实施中，可以包括另外的功能单元。每个功能单元 102 至 112 以各种方式与一个大的芯片下（off-chip）存储器 114 进行交互，该存储器 114 是一个动态随机存取存储器或 DRAM。对 DRAM 114 的访问由 ASIC 100 上的存储器控制器 116 控制。

出于示例的目的，下面将说明 ASIC 100 的几个功能单元。这些功能单元包括一个可编程输入多路分解器 102，该可编程输入多路分解器 102 最好在线 118 上以直至 72Mbps 的数据速率接收数字编码视频/音频输入信号。数字编码视频/音频信号是一个分组数据的串行位流，通常被称为具有识别分组数据的配置的预先确定的位流语法的“分组载流线”或 PES 流。多路分解器 102 负责将输入的数字编码信号分成单个的视频、音频或其他数据分组，并用存储器控制器 116 将输入的

信号存储进 DRAM 114 中的缓冲器. 如同下面将要详细说明的, 依据本发明的原理的视频译码器 104 用于取还存储在 DRAM 114 中的缓冲器中的视频数据和图象控制信息, (用对 DRAM 存储器 114 的经常和重复的访问来)译码视频数据和控制信息, 然后将译码的视频信息传送到输出编码器 108, 以便通过总线 120 在监视器上输出. 输出编码器 108 最好是一个 NTSC/PAL 编码器,对于 NTSC 以 30fps 提供 720×480 象素的图象尺寸, 对于 PAL 以 25fps 提供 720×576 象素的图象尺寸. 音频译码器 106 从 DRAM 114 取还音频信息, 对音频信息进行译码以供输出, 并将要输出的音频信息通过总线 122 传送到扬声器. ASIC 100 还包括一个用于通过线 124 与主微控制器进行交互、以便允许主计算机监视和控制 ASIC 100 的操作的主接口 110.

除了所显示的功能单元之外, 可以有一个或多个另外的用于译码子照片视频信息的视频译码单元, 子照片视频信息可以包含伴随由视频译码器 104 译码的主视频的子标题或其他信息. 此外, 屏上显示可以由与 RBUS 126 和 GBUS 128 相连的 OSD 部分在 ASIC 100 内产生. 屏上显示可以响应于从主 CPU 接收的命令而产生, 以便提供关于主 CPU 的操作的反馈和/或使用 ASIC 100 的设备的重放或接收状态.

对专用集成电路 100 的控制是由一个精简指令集中央处理单元(RISC CPU) 112 提供的, 其控制和监视 ASIC 100 上的每个其他功能单元的操作. RISC CPU 112 响应存储在指令存储器 130 中的 16-位指令. 指令存储器 130 保存 4096 个 16-位指令, 这对于相对简单的 ASIC 100 程序来说是足够的. 对于可以由 ASIC 100 执行的相对复杂的程序来说, 可以将 4096 个指令的“指令页”从 DRAM 114 中的更大的指令缓冲器交换进或交换出程序存储器 130.

如图 1 所示, RISC CPU 112 通过两个主总线、即 RBUS 126 和 GBUS 128 与 ASIC 100 中的每个功能单元交互. 具体地说, 每个功能单元与 64 位 GBUS 128 相连, 通过 GBUS 128 可以取还或传送数据到存储器控制器 116, 因而取还或传送数据到 DRAM 116. 此外, 可以将数据块通过 GBUS 128 从一个功能单元传送到另一个功能单元.

将对存储器访问或传送以及各种其他命令的请求通过 RBUS 126 从一个功能单元传送到其他功能单元. RBUS 126 可以包括由对存储器进行经常访问的功能单元使用的一个或多个 32-位总线, 或由几个功能

单元共享的单个 8-位、时分多路复用总线. RBUS 控制器 132 接收使用 RBUS 126 的请求，根据需要在这些请求之间仲裁，并将对 RBUS 的访问传送到最高优先权请求功能单元。

当请求存储器访问时，请求功能单元 102 至 112 通过 RBUS 126 将一个虚拟地址传送到存储器控制器 116. 存储器访问请求可以请求传送单个存储单元，或者可以包括对要响应请求而被访问的多个存储单元的识别。存储器控制器 116 通过响应于请求来管理对 DRAM 114 中的识别单元的访问来响应请求. 如果对存储器访问的多个请求在任何时间是待决的，则存储器控制器 116 在待决请求之间仲裁，以便允许对最高优先权请求功能单元的访问。关于存储器控制器 116 响应请求的操作的更进一步的细节将在 1997 年 4 月 30 日申请的、题目为“用于数字视频的存储器地址产生”的共同未决的美国专利申请 No. 08/846,590 中找到，该申请在这里全部作为参考. 另外，关于 RISC CPU 112 及其精简指令集的操作可以在 1997 年 3 月 30 日申请的、题目为“用于数字音频视频译码的特定用途处理器”的共同未决的美国专利申请 No. 08/865,749 中找到，该申请在这里全部作为参考.

关于各种功能单元的状态的附加数据可以通过 RBUS 126 得到. 功能单元通过 RBUS 126 提供可以在识别出的特定地址访问的状态信息。因此，例如，为了访问一个来自视频译码器 104 的状态字，将一个识别 DEC_VALUE 地址的访问请求传送到 RBUS 控制器 132. 作为响应，RBUS 控制器 132 使得视频译码器 104 的状态字被传送到请求功能单元。

命令也通过 RBUS 126 发送到功能单元. 为了将命令传送到功能单元，通过 RBUS 将命令传递到用于功能单元的一特定地址。因此，例如，为了将命令传送到视频译码器 104，将一个识别 VLD_CMD 地址的访问请求传送到 RBUS 控制器 132。作为响应，RBUS 控制器 132 使得请求功能单元能够将命令传送给 RBUS 126，并使得视频译码器 104 将命令接收进其命令缓冲器.

视频译码器 104 是本发明的焦点，其操作和特征在图 2-6G 中显示得更清楚. 依据本发明的原理，视频译码器 104 特别适合于执行依从 MPEG-1 和主简档@主级别 MPEG-2 语法的数字编码视频信号的视频译码. 如整体在这里作为参考的 ISO/IEC 13182-2:1995 (E) 所述，MPEG-2 语法定义了视频数据和控制信息的压缩位流，以六层表示一个图象序

列，具体为：序列层，照片组层，照片层，时间片层，宏块层，以及块层。编码位流中的这些层中的每一层包括一个唯一的起始代码，该起始代码识别该层，并提供固定长度数据值或可变长度霍夫曼编码数据或二者兼而有之，这些数据必须由视频译码器 104 进行语法分析和译码。在时间片层之上的图象控制信息的上层语法分析由 RISC CPU 112 处理。这样，RISC CPU 112 从压缩视频位流获得足够的信息来控制视频位流中的所发送图象序列的解压缩、重构和展示。

现在参看图 2 和图 2A，详细显示了视频译码器 104 的方框图。如同下面将要更加详细地说明的，视频译码器 104 的主要功能是一个用于对各个时间片的宏块和块层进行译码的单个有序码元每周期时间片语法分析机。当达到一个时间片的结束时，向 RISC CPU 112 发送一个中断，执行各种存储器检查，然后指示下一个时间片的处理。视频译码器 104 的操作主要通过微定序器 134 来控制，微定序器 134 通过线 138 与 VLD 命令/译码执行单元 136 接口。微定序器 134 包括一个 256×16 的指令 ROM 140 和一个指令译码/控制单元 142（参看图 2A）。在每个时间片的宏块和块层的译码过程中，将微定序器 134 编程为或者向 VLD 命令/译码执行单元 136 发出执行可变长度译码的命令，或者控制通过其指令集提供的程序流。另外，VLD 命令/译码执行单元 136 可以通过 RBUS 接口 144、线 146 和多路复用器 148 直接从 RISC CPU 112 接收指令。如同下面将要更加详细地说明的，RISC CPU 112、微定序器 134 和 VLD 命令译码/执行单元 136 能够读和写形成 VLD 体系数据路径的一部分的一系列 VLD 命令指令寄存器（VCI 寄存器）150 的内容。

如上所述，编码视频数据最初存储在 DRAM 114 中。编码视频数据通过存储器控制器 116 经 GBUS 128 对于视频译码器 104 是可用的。视频译码器 104 包括一个 16×64 信道缓冲器 FIFO 形式的 GBUS 接口 152，用于存储足够量的编码视频位流，以确保缓冲器 FIFO 152 在译码过程中不会空或过满。

视频译码器 104 的主要功能之一是对视频位流中的可变长度编码数据进行译码。可变长度数据是根据 MPEG-1 和-2 标准中的 VLC 表进行霍夫曼编码的。本发明的 VLD 支持至少十一个 MPEG VLC 表，包括：macroblock_type_I； macroblock_type_P； macroblock_type_B；

macroblock_type_D ; macroblock_address_increment; dct_dc_size_luma; dct_dc_size_chrominance; ac_table(table 0 和 table 1); coded_block_pattern;motion_code; 以及 dmvector.

如图 2 所示，信道缓冲器 FIFO 152 通过相应的 64-位总线从信道缓冲器 FIFO 152 向一对 A 和 B 寄存器 156A 和 156B 中的每一个提供 64 位的视频数据。依据本发明的一个方面，提供旋转器/桶形移位器 158 和指针寄存器 160 来使得从指针地址到包括指针地址 + 31 在内的位对于 VLD 命令译码/执行单元 136 是可见的。将这 32 位作为旋转器/桶形移位器数据 162 提供给 VLD 命令译码/执行单元 136。

参看图 2 和 3，A 和 B 寄存器 156A 和 156B 以一个 128-位的环形相连，以允许指针的边界交叉从 A 寄存器中的位 63 到 B 寄存器中的位 64，从 B 寄存器中的位 127 到 A 寄存器中的位 0。当指针从 A 寄存器到 B 寄存器时，A 寄存器的内容以新数据更新。同样，当指针从 B 寄存器到 A 寄存器时，B 寄存器的内容以新数据更新。为了允许指针绕环形的运动，并使得 128 个位中的每一位对于指针都是可达到的，图 3 所示的桶形移位器 164 具有七个选择器级，提供数据向左的以 2 为幂的各种增量移位或不移位。标为“级 64/0”的第一级 166 具有 128 个选择器（D_0-D_{127}）来允许 128 个位中的每一位左移 64 位或不移位。标为“级 32/0”的第二级 168 具有 95 个选择器来提供左移 32 位或不移位。标为“级 16/0”的第三级 170 具有 63 个选择器来提供左移 16 位或不移位。标为“级 8/0”的第四级 172 具有 47 个选择器来提供左移 8 位或不移位。标为“级 4/0”的第五级 174 具有 39 个选择器来提供左移 4 位或不移位。标为“级 2/0”的第六级 176 具有 35 个选择器来提供左移 2 位或不移位。标为“级 1/0”的第七级也就是最后一级 178 具有 33 个选择器来提供左移 1 位或不移位。

采用桶形移位器 164 的结构，可以使 A 和 B 寄存器 156A 和 156B 中的 128 位中的任何 32 位作为旋转器/桶形移位器数据 162 可用于 VLD 命令译码/执行单元 136。在级 166-178 的每一级中，左移由箭头 180 表示，不移动由箭头 182 表示。本领域普通技术人员将会理解，如同这里相对于桶形移位器 164 所用的，术语“选择器”意味着提供上面详细说明的左移或不移动操作的任何电路。通过依据本发明的桶形移位器 164 的操作，可以从几级 168-178 中去除如图 3 中的虚三角框所

概略地表示的各种选择器。通过将桶形移位器 164 截成一个梯形而不是对称的长方形，可以理解，可以去除不必要的选择器，以节约花费和硬件不动产。

如图 2 和 5 所更清楚地显示的，使得旋转器/桶形移位器 162 的 32 位对于包含 MPEG 标准可变长度代码（VLC）表的可变长度表译码器 184 是可用的。依据本发明的另一个方面，提供了一个新颖的方案，使得 VLD 命令译码/执行单元 136 访问十一个 MPEG VLC 表中的每一个。如图 4 所示，在“macroblock_address_increment”的 VLC 表的例子中，在每个 MPEG VLC 表中识别一个前缀模式，该前缀模式定义在可变代码中的“1”第一次出现之前“0”的个数，以及产生一个所有唯一的前缀模式的集合所需的任何额外逻辑。

例如，在图 4 的“macroblock_address_increment”表中，每个可变长度代码具有一个被定义为 K_0、K_1、K_2、K_3、K_{4_0}、K_{4_1}、K_{5_1}、K_{5_01}、K_{5_001}、K_{5_000} 和 K_6 的规定前缀模式。K_0 表示在“1”第一次出现之前没有“0”，K_1 表示在“1”第一次出现之前有一个“0”，K_2、K_3 和 K_6 也依此类推。在几个可变长度代码的情况下，在“1”之前的领先“0”模式之后需要额外的逻辑、例如 K_{4_0}、K_{4_1}、K_{5_1}、K_{5_01}、K_{5_001} 和 K_{5_000} 来产生一个所有唯一的前缀模式的集合。这样，可以由每个 MPEG VLC 表中的前缀模式定义一个子表集合，每个子表对应唯一的前缀模式中的一个。此外，每个可变长度代码具有一个定义可变长度代码中的位数的代码长度。一个增量值与“macroblock_address_increment”的每个可变长度代码相联系，以便相对于照片的左沿或距离最近发送的宏块的差分增量值来限定一个时间片中的第一宏块的水平位置。本领域普通技术人员从这个例子中将会容易地理解如何为其他 MPEG VLC 表定义前缀模式和额外逻辑。

下面参考图 5 所示的可变长度表译码器 186，在可变长度译码过程中，将 32 位的旋转器/桶形移位器数据 162 加到模式匹配逻辑和'MUX 控制上，以识别旋转器/桶形移位器数据 162 中的唯一的前缀模式。所识别的前缀模式匹配、例如 K_0、K_1 等用作为信号“MUX CNTL”190 来控制十一个 MPEG VLC 表中的每一个中的每个 MUX 192 的输出。因此，由模式匹配逻辑和 MUX 控制 188 确定的前缀模式匹配在全部十一个 MPEG VLC 表、包括如图 5 所示的“macroblock_address_increment”

和“motion_code”VLC表之间共享，以控制其各个MUX 192的输出。

在任何给定时间被译码的VLC表是由VLD命令译码/执行单元136提供的“VLC TABLE SELECT”信号194确定的，这将在下面详细说明。在前缀模式匹配由模式匹配逻辑和MUX控制188进行识别以便限定要加到每个MUX 192的MUX CNTL信号190的同时，将在32位旋转器/桶形移位器数据162中的领先模式匹配之后的位同时加到十一个MPEG VLC表中的每一个的所有子表，如总线196所示。因此，如果VLC表“macroblock_address_increment”由“VLC TABLE SELECT”信号194选择，并且前缀模式匹配是K_{5_1}，并且通过总线196将一个“1”加到子表K_{5_1}，则MUX 192将输出一个译码值14（表示增量值）、一个代码长度8（表示可变长度代码中的位数）和一个有效状态位（表示代码的有效性）到VLD命令译码/执行单元136。因此，在完成译码过程之后，可变长度表译码器186将两个信号提供给VLD命令译码/执行单元136，包括译码值198和有效代码状态位200。可变长度表译码器186还将线204上的代码长度信号202提供给指针寄存器160，以便以代码长度递增指针寄存器。可变长度表译码器186可以是RAM或ROM，但它最好是硬连线优化随机逻辑。

现在参考图2和3，来自可变长度表译码器186的代码长度信号202用于递增指针寄存器160。代码长度信号202还由桶形移位器164用来控制桶形移位器中必需的移位模式。例如，如果代码长度信号202的值为13，则桶形移位器164的级172(8/0)、174(4/0)和178(1/0)将被允许引起左移13位，而其他所有级则没有移位。在由桶形移位器164执行了移位操作之后，使一个新的旋转器/桶形移位器数据162的集合对于VLD命令译码/执行单元136和可变长度表译码器186是可见的。本发明的可变长度表译码器186和桶形移位器164提供对MPEG可变长度代码的有效译码，同时减少了对VLC表的存储器需求。

依据MPEG标准，将可变长度编码的DCT系数译码为运行-长度和幅度电平对，如图2中的码元“<r,l>”所示。值“r”表示在具有由“l”表示的幅度电平的系数之前的零值系数的数目。例如，码元<5,2>表示在系数值2之前有5个零。依据本发明的又一方面，由于运行-长度和幅度电平对是由可变长度表译码器186译码的，所以将运行-长度和幅度电平对码元从VLD命令译码/执行单元136加到一个64×18的

运行-电平译码器/FIFO 206，将其存储为压缩对，直到反之字形、反量化和反 DCT 变换单元 208 需要。应该理解，由于压缩码元的数目少于 DCT 系数的数目，所以运行-电平译码器/FIFO 206 允许与前面译码的运行-电平对的运行-电平译码并行地对霍夫曼编码的可变长度对进行译码。

如图 2 所示，将译码的运动矢量“mv”和差分运动矢量“dmv”从 VLD 命令译码/执行单元 136 加到一个 16×13 的 mv/dmv FIFO 210，将其一直存储到运动补偿单元 212 需要为止。如同在 1997 年 12 月 30 日申请的、题目为“运动补偿数字视频译码及缓冲器存储器寻址”的共同未决的美国专利申请 No.09/001,122（该申请在这里全部作为参考）中详细说明的，译码视频数据值和运动矢量由运动补偿单元 212 进行合并，以形成用于显示的全 I-、P-和 B-帧。VLD 命令译码/执行单元 136 最好具有用于运动矢量、差分运动矢量、宏块增量地址、dc 系数和 ac 系数计算的状态机。

图 6A-6G 显示了由视频译码器 104 的微定序器 134 支持的指令集，而图 2A 显示了在微定序器 134 与 VLD 命令译码/执行单元 136 之间的全部程序流控制。特别地，微定序器指令集包括八个指令 214a-214g，每个指令使用三位操作码 216，而指令 214b 表示两个不同的指令。在图 6A-6G 所示的指令中，只有指令 214b 的两个指令 ICMD 和 CMDI 是执行命令。其他六个指令 214a 和 214c-214g 是流程控制指令，这将在下面更详细地说明。指令 214a-214g 中的大多数对在 VCI 寄存器 150 中找到的数据进行操作。

下面的表提供了对各种 VCI 寄存器 150 的描述。表 1 定义了 VCI 控制寄存器描述。表 2 定义了宏块和块层寄存器描述。表 3 定义了用于上层译码的照片层寄存器描述。表 3 寄存器是由 RISC CPU 112 写入的。表 4 定义了控制和状态寄存器描述，表 5 定义了预测器和状态机寄存器描述。

表1.VCI控制寄存器定义

名称	字段	类型	描述
vci_addr	[5:0]	r/w	VCI 间接寄存器地址
vci_data	[15:0]	r/w	VCI 间接数据

VCI 控制寄存器用于访问在下面的表 2-5 中定义的 VCI 间接寄存器。RISC CPU 112 将 VCI_addr 寄存器设置为读或写特定地址的 VCI 间接寄存器的内容. 这个读和写操作由 VCI_data 寄存器完成.

表2.宏块级的VCI间接寄存器定义

名称	地址	宽度	Dir	描述
vci_mba_x	0x00	7	r/w	宏块地址x维数
vci_mba_y	0x01	7	r/w	宏块地址y维数
vci_vld_out	0x02	16	r/w	VLD的临时输出
vci_q_scale	0x03	5	r/w	量化器比例信息
vci_cbp	0x04	8	r/w	编码块模式

表2.宏块级的VCI间接寄存器定义（续表）

	名称	地址	宽度	Dir	描述
	vci_mtype	0x05	5	r/w	宏块类型
	vci_motype	0x06	3	r/w	译码运动类型 [0] mv_count 0:1向量, 1:2向量 [1] mv_format 0:场,1: 帧 [2] dmv
5	vci_dct_type	0x07	1	r/w	DCT类型0: 帧, 1: 场
	vci_mvfsl	0x08	1	r/w	运动垂直场选择 -
	reserved	0x09	1		保留
	vci_temp_0	0x0a to 0x0f		r/w	保留

表3.照片级层 的VCI间接寄存器定义

名称	地址	宽度	目录	描述
vci_pic_init	0x10	26	r/w	[25:24]:DC 精度 -->00:8-bit, 01:9-bit, 10:10-bit -->11:11-bit [23:20]: 反向垂直帧代码 [19:16]: 反向水平帧代码 fcode [15:12]: 正向垂直帧代码 [11:8]: 正向水平帧代码 [7]: 隐藏运动矢量 [6]: 内vlc格式 [5]: frame pred frame dct [4:2]: 图象编码类型 (I,P,B,D) [1:0]: 图象结构（场，帧） -->[00]: 保留 -->[01]: 顶场 -->[10]: 底场 -->[11]: 帧
vci_seq_init	0x11	10	r/w	[10:4]: mb数×维数 [3]: 0: 常规，1: 专用，--blk类型 [2]: 0: mpeg2, 1:mpeg1 [1:0]: 色度格式 -->00: 保留 , 01: 420, 10:422, -->11: 444
vci_conceal	0x12	3	r/w	隐藏计数寄存器
vci_temp_1	0x13 to 0x1f		r/w	保留

表4.控制和状态的VCI间接寄存器定义

名称	地址	宽度	目录	描述
vci_cntl	0x20	2	r/w	vci状态控制寄存器 [31:30]=00 :运行 [31:30]=10: 暂停 [31:30]=11: 复位
vci_pc	0x21	8	r/w	vci程序计数器
vci_rom	0x22	16	r/w	vci_rom 输出端口
vci_dmvfifo_adr	0x23	3	r/w	dmb_fifo 读/写地址
vci_dmvfifo	0x24	2	r/w	dmv_fifo 数据端口
vci_pointer	0x25	7	r/w	旋转器指针
vci_dec_lpred	0x26	12	r/w	Luma DC预测值. 对DC_Lpred的写复位到由DEC_模式寄存器中的DC精度位所指示的恒定值
vci_dec_cpred	0x27		r/w	色度DC预测值，对DEC_cpred的写复位到由DEC_模式寄存器中的DC精度位所指示的恒定值
		[23:12]		V
		[11:10]		U
vci_temp_2	0x28 to 0x2f			保留

表5.预测器&状态机的VCI间接寄存器定义

名称	地址	宽度	目录	描述
vci_mv_predfh0	0x30	13	r/w	mvfs,运动矢量预测器 正向水平－第一
vci_mv_predfv0	0x31	13	r/w	mvfs,运动矢量预测器 正向垂直－第一
vci_mv_predbh0	0x32	13	r/w	mvfs,运动矢量预测器 反向垂直－第一
vci_mv_predfh1	0x34	13	r/w	mvfs,运动矢量预测器 正向水平－第二
vci_mv_predfv1	0x35	13	r/w	mvfs,运动矢量预测器 正向垂直－第二
vci_mv_predbh1	0x36	13	r/w	mvfs,运动矢量预测器 反向水平－第二
vci_mv_predbv1	0x37	13	r/w	mvfs,运动矢量预测器 反向垂直－第二
vci_err_bits	0x38	10	r	误差条件 [9] motype_err [8] coef_err [7] rl_error [6] cbp_err [5] mv_err [4] mbi_err [3] mtype_err [2] get_ac_err [1] dctdcsz_chroma_err [0] dctdcsz_luma_err

表5. 预测器&状态机VCI间接寄存器定义（续表）

名称	地址	宽度	目录	描述
vci_stm	0x39	4	r	状态机 [14:12] vstate : motion vec STM [11:9] dc_cstate : dc STM [8:7] sc_state : start-code STM [6:3] ms_cstate : MB STM [2:0] lc_cstate : block STM
vci_tmp_3	0x3a to 0x3f			

表6.定义带有RBUS接口的VLD控制寄存器

144:

表6.带有rbus接口的VLD寄存器

名称	目录	位字段	描述
dec_value1	r/w	[15:0]	为除DECODE BLOCK命令之外的所有命令保存译码值。 DECODE BLOCK命令返回这个字段中的译码DC系数
dec_value2	r/w		该值保存当前运行和电平
		[17:12]	译码运行长度
		[11:0]	译码电平
mvfifo_adr	r/w	[4:0]	mv_fifo 读/写地址 位4-0:读，1: 写
mvfifo_data	r/w	[12:0]	mv_fifo 数据端口

表6.带有rbus接口的VLD寄存器（续表）

名称	目录	位字段	描述
vld_cntl	r/w	[31:30]	00:运行，10：暂停，11：复位
vld_pic_hdr	r/w	[31:0]]	照片级参数: [31:28] f_code[0][0] (F,H) [27:24] f_code[0][1] (F,V) [23:20] f_code[1][0] (B,H) [19:16] f_code[1][1] (B,V) [15:14] intra_dc_prec [13:12] pic_structure [11] topfld_first [10] frame_prediction_frame_dct [9] concealment_motion_vectors [8] q_scale_type [7] intra_vlc_format [6] alternate_scan [5] repeat_first_field [4] chroma_420 [3] progressive_frame [2:0] pic_type
rlfifo_adr	r/w	[6:0]	rl_fifo 读/写地址 位6－0：读，1：写
rlfifo_data	r/w	[17:0]	rl_fifo 数据端口
vld_status	r/w	[10:0]	状态位 [10] vld_busy [9] vld_mv_fifo_empty [8] vld_cfifo_empty [7:4] chfifo_wr_addr [3:0] chfifo_rd_addr
vld_cmd	w	[7:0]	要从CPU执行的vld命令
dec_status	r/w	[15]	出错．如果检测到位流误差则设置粘性位.

表6.带有rbus接口的VLD寄存器（续表）

名称	目录	位字段	描述
		[14]	chan_fifo_empty. 信道FIFO空
		[13:10]	chan_fifo_wrprt. 信道FIFO写指针
		[9:6]	chan_fifo_rdptr. 信道FIFO读指针
		[5:0]	bitcnt译码器位流读指针，包含下一次被译码器读取的位数
vld_cmd	r/w	[7:0]	来自CPU的vld命令－如果从rbus接口写该地址，则执行vld命令

表7定义带有GBUS接口的VLD控制寄存器

152:

表7.带有gbus接口的VLD寄存器

名称	目录	位字段	描述
word_fifo	r/w	[63:0]	译码器FIFO数据端口

现在参看图 6A-6G 的微定序器指令 214a-214g，TBIT（测试位）指令 214a 测试 VCI 寄存器 220 中的一个位（由“bitnum” 218 给出），如果等于 4 位正向相对地址 222 的值则转移。COMP（比较）指令 214c 将 vci_vld_out 寄存器的 8 msb 或 8 lsb 数据内容（由“msb” 224 的状态确定）与立即数 226 进行比较，并将结果存储在标志寄存器 228 中。如果“st_code” 230=1，则将 vci_vld_out 寄存器的内容与起始代码模式进行比较，如果相匹配则将标志寄存器 228 设置为表明起始代码已经找到。BRANCH 指令 214d 执行到 8 位转移目标地址 232 的绝对转移。“err” 位 234 表明 vci 误差，“halt” 位表明将 vci 控制寄存器改变为暂停状态。SETF（设置标志）指令 214e 用依据立即数 238 的值来设置标志寄存器的 8 lsb。INCM（增量宏块）指令 214f 将 VCI

寄存器 240 的较低字节与立即数 242 进行比较，如果不相等就停止。流程控制指令的最后一个指令、HALT 指令 214g 暂停视频译码器 104，并发出（deassert）vld_busy 信号。

ICMD 和 CMDI（发出命令）指令 214b 向视频译码器 104 发出 22 个命令中的一个，并将输出存储在 VCI 寄存器 244 中。通过 ICMD 指令，微定序器 134 向视频译码器 104 发出一个命令并等待。CMDI 指令使得微定序器 134 向视频译码器 104 发出一个命令，并保持运行微代码。

在表 8 中提供了向视频译码器 104 发出的各种命令:

表8.VLD命令

OP 代码	符号	描述
0000 0001	<vld_escape>	宏块换码符
0000 0010	<vld_peek>	取数－不执行，只更新CC
0000 0011	<vld_startcode>	找到起始代码
0000 0100	<vld_mbi>	宏块地址递增
0000 0101	<vld_cbp>	编码块模式
0000 0110	<vld_intra_luma>	运行内亮度块
0000 0111	<vld_intra_chromau>	运行内色度U块
0000 1000	<vld_intra>chromav>	运行内色度V块
0000 1001	<vld_non_intra>	运行非内块
0000 1010*	<vld_mbs>	产生宏块起始信号
0000 1100*	<vld_dpcm>	复位亮度和色度预测器
0000 1101*	<vld_mv_pred>	复位运动矢量预测器
0001 0000	<vld_dmv>	获得双重主要运动矢量
0001 0100	<vld_non_coded>	运行非编码块
0001 0101	<vld_field_motype>	译码场运动类型
0001 0110	<vld_frame_motype>	译码帧运动类型
0001 1000	<vld_mtypei>	为I照片获得宏块类型
0001 1001	<vld_mtypep>	为P照片获得宏块类型
0001 1010	<vld_mtypeb>	为B照片获得宏块类型
0001 1011	<vld_mtyped>	为D照片获得宏块类型
0010 xxxx	<vld_get<bitxx>	获得下面的1至16位 (0=>16)
0011 0xyz	<vld_mvxyz>	获得运动矢量：x-正向，y－水平，z-第一

*: 需要从 cmdi 指令调度

表 8 的 VLD 命令是对于许多类型的编码都通用的自主操作. VLD 指令一般可以分类为用于从 MPEG VLC 表获得值的可变长度表译码命令、用于从编码视频数据接收参数化的位数的指令或块操作。

<vld_get_bitxx>命令例如从旋转器/桶形移位器 158 提取特定的位数，并用所取出的位数来增量指针寄存器 160. <vld_dmv>命令采用来自视频数据位流的三个参数从一个 MPEG VLC 表取出运动矢量值。具体地，将每个运动矢量存储在前一运动矢量和预测值的组合. 预测值本身以商（可变长度编码的）和余数（固定长度代码）的形式编码. <vld_dmv>命令用于通过 MPEG VLC 表对商进行译码，以确定余数的位置和商的值. 余数是采用<vld_get_bitxx>命令获得的. 最后，将商和余数进行合并，以生成运动矢量分量. <vld_peek>命令允许在不以所取出的位数递增指针寄存器 160 的情况下的来自旋转器/桶形移位器 158 的特定位数的可见性。

虽然已经以对各个实施例的描述说明了本发明并且已经相当详细地说明了这些实施例，但另外的优点和修改对于本领域普通技术人员来说将变得显而易见. 更广义方面的发明因此并不限于特定细节、示意性装置和方法以及所示和所述的示例性实例. 因此，在不背离本申请的总的发明概念的精神和范围的情况下，可对这些细节作出修改。

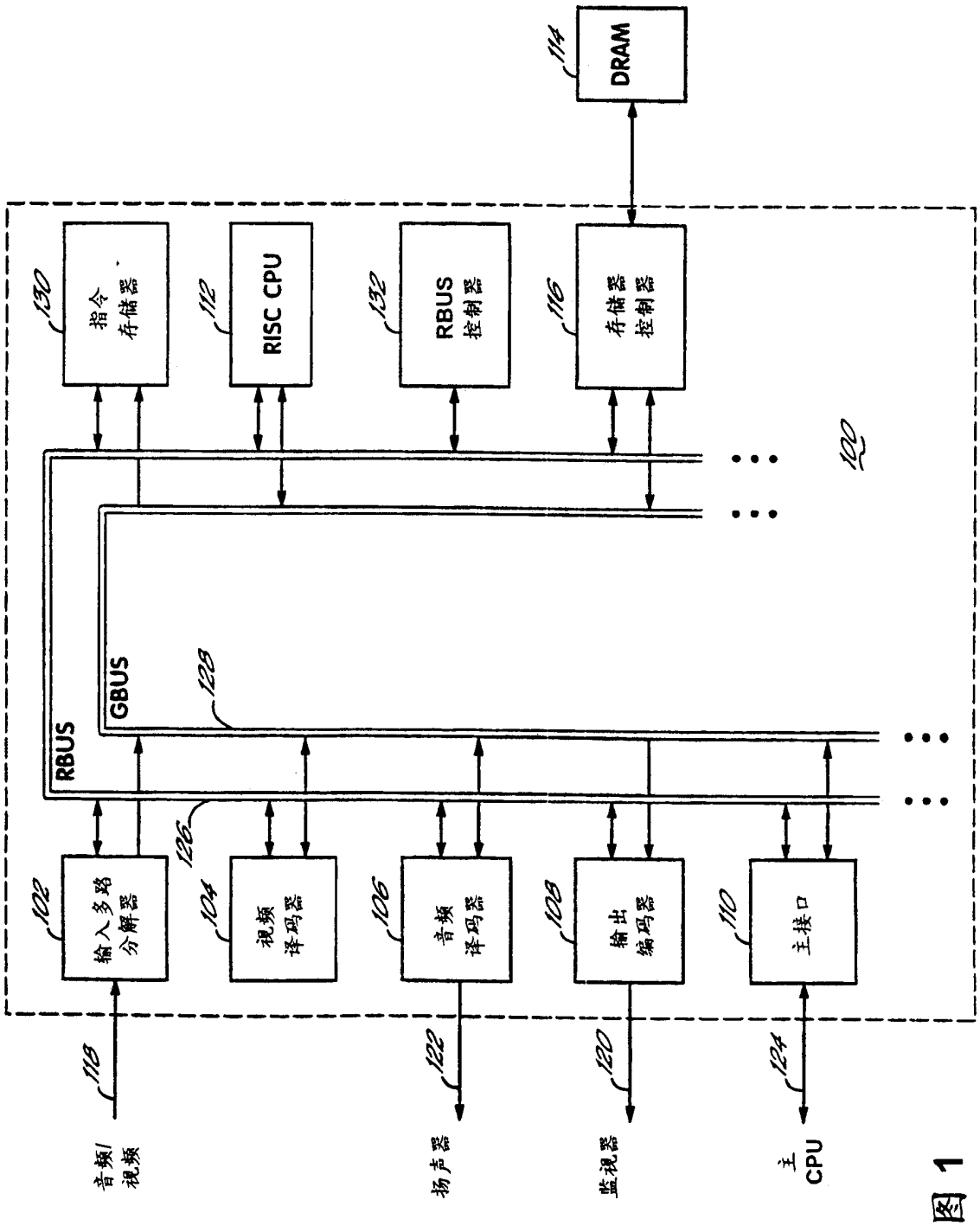

DRAM
114
指令存储器
130
RISC CPU
112
RBUS控制器
132
存储器控制器
116
100
GBUS
128
RBUS
126
输入多路分解器
102
视频译码器
104
音频译码器
106
输出编码器
108
主接口
110
118
122
120
124
音频/视频
扬声器
监视器
主CPU

图 1

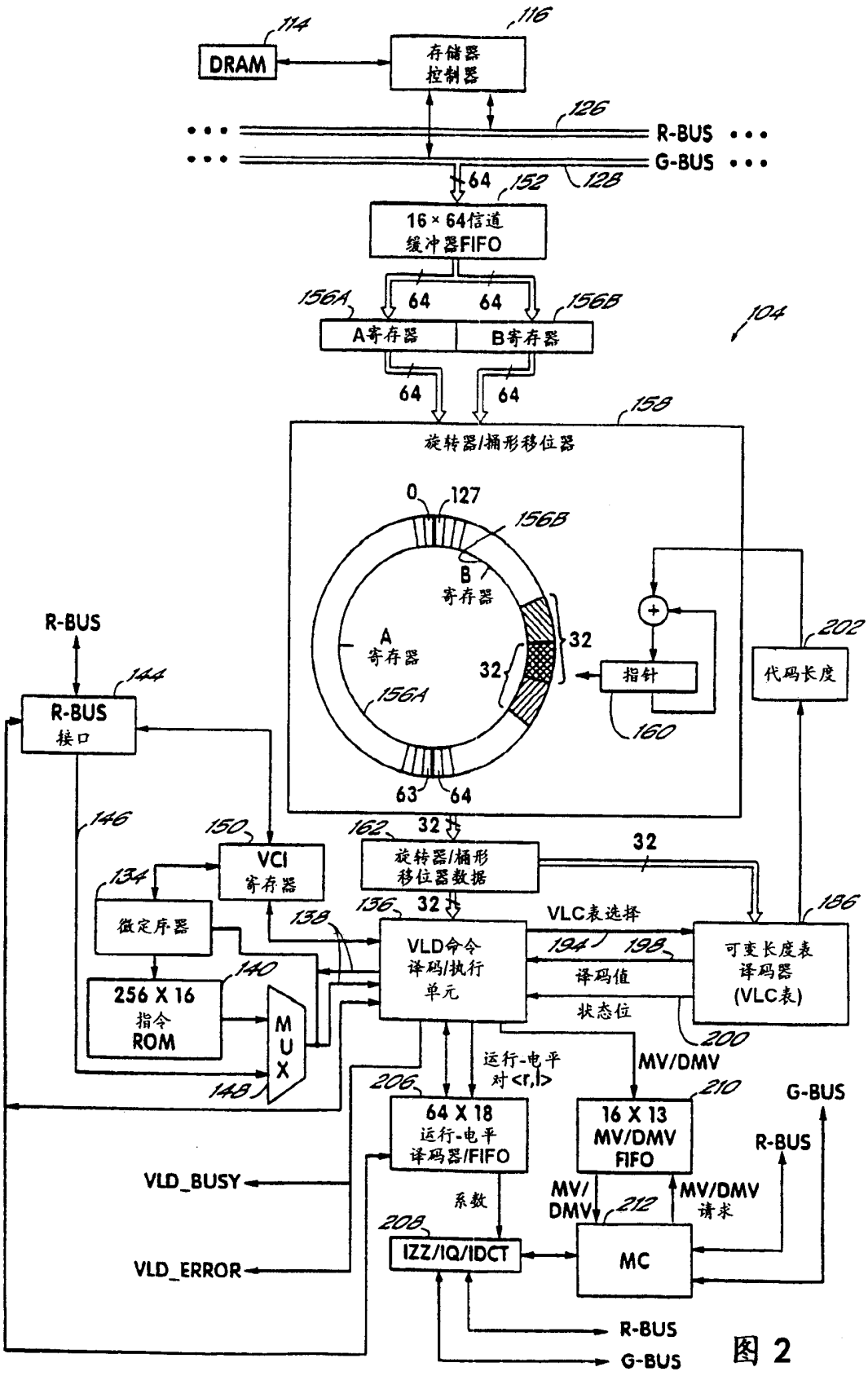

DRAM
存储器控制器
R-BUS
G-BUS
16×64信道缓冲器FIFO
A寄存器
B寄存器
旋转器/桶形移位器
指针
代码长度
R-BUS接口
VCI寄存器
旋转器/桶形移位器数据
微定序器
256 X 16指令ROM
MUX
VLD命令译码/执行单元
VLC表选择
可变长度表译码器(VLC表)
译码值
状态位
运行-电平对<r,l>
MV/DMV
64 X 18运行-电平译码器/FIFO
16 X 13 MV/DMV FIFO
系数
MV/DMV请求
IZZ/IQ/IDCT
MC
VLD_BUSY
VLD_ERROR
R-BUS
G-BUS

图 2

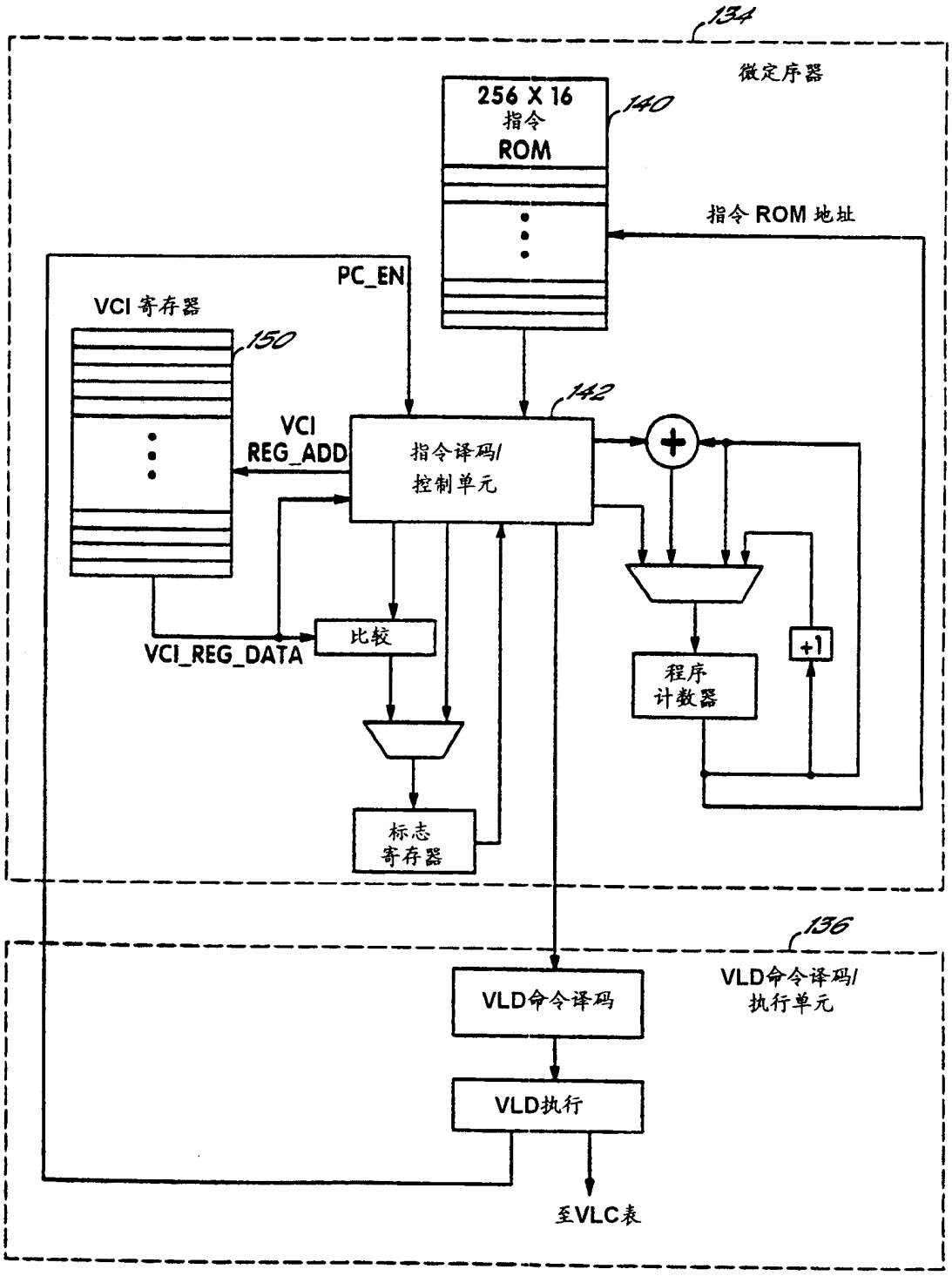

134
微定序器
256 X 16
指令
ROM
140
指令 ROM 地址
PC_EN
VCI 寄存器
150
142
VCI
REG_ADD
指令译码/
控制单元
比较
VCI_REG_DATA
程序
计数器
+1
标志
寄存器
136
VLD命令译码/
执行单元
VLD命令译码
VLD执行
至VLC表

图 2A

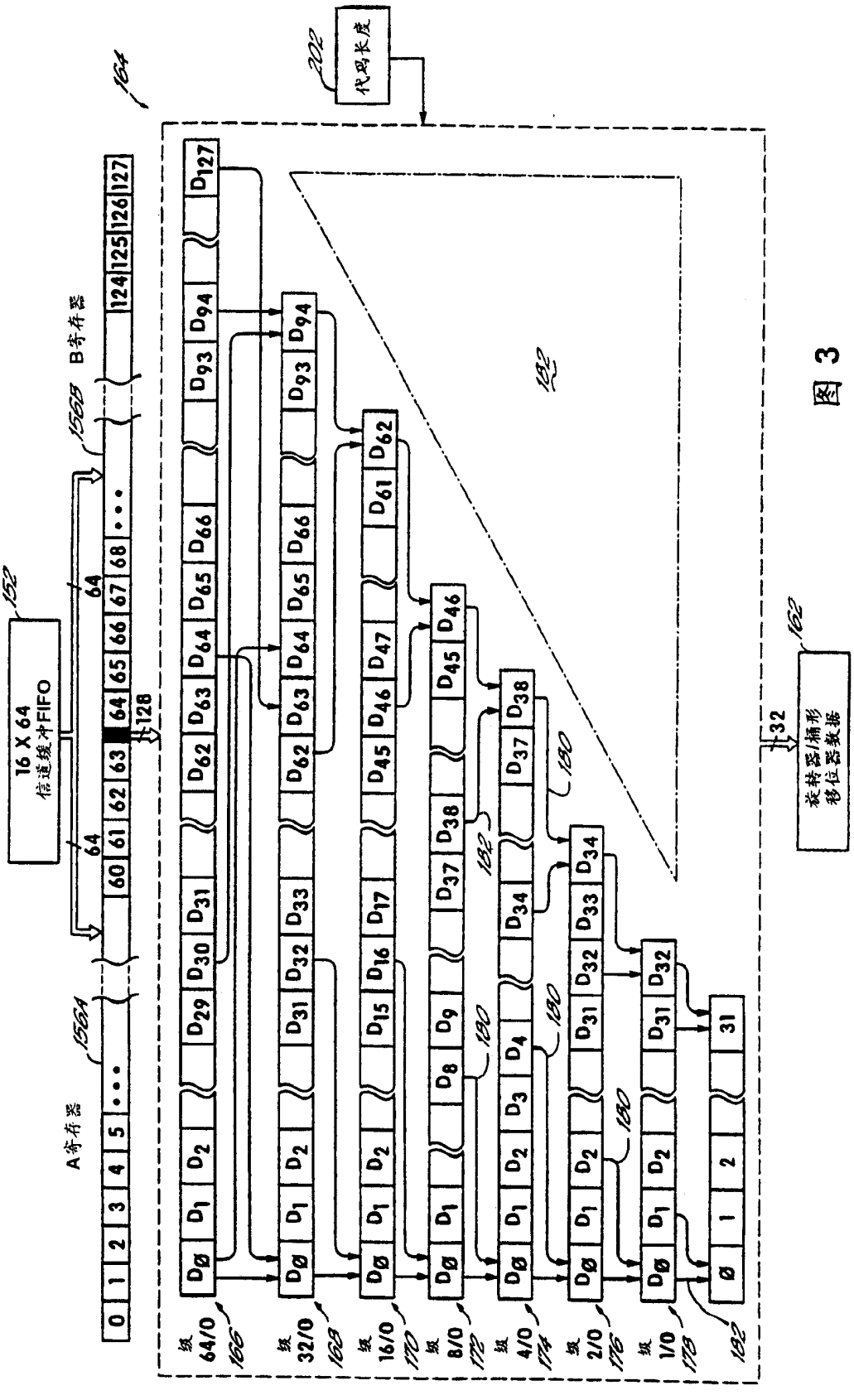

16 X 64
信道缓冲FIFO
152
A寄存器
B寄存器
154
158
164
128
64
32
202
代码长度
162
旋转器/桶形
移位器数据
182
180
190
级
64/0
166
级
32/0
168
级
16/0
170
级
8/0
172
级
4/0
174
级
2/0
176
级
1/0
178
192

图 3

模式匹配
用于"MACROBLOCK_ADDRESS_INCREMENT"的VLC表

可变长度代码	增量值	模式匹配	代码长度
1	1	$K_{Ø}$	1
011	2	K_1	3
010	3	K_1	3
0011	4	K_2	4
0010	5	K_2	4
00011	6	K_3	5
00010	7	K_3	5
0000111	8	K_{4_1}	7
0000110	9	K_{4_1}	7
0000101	10	$K_{4\_Ø}$	8
0000100	11	$K4\_Ø$	8
0000011	12	$K4\_Ø$	8
0000010	13	$K4\_Ø$	8
0000001	14	K_{5_1}	8
0000000	15	K_{5_1}	8
00000101	16	$K_{5\_Ø1}$	10
00000100	17	$K_{5\_Ø1}$	10
00000011	18	$K_{5\_Ø1}$	10
00000010	19	$K_{5\_Ø1}$	10
00000011	20	$K_{5\_ØØ1}$	10
00000010	21	$K_{5\_ØØ1}$	10
00000011	22	$K_{5\_ØØØ}$	11
00000010	23	$K_{5\_ØØØ}$	11
00000001	24	$K_{5\_ØØØ}$	11
00000000	25	$K_{5\_ØØØ}$	11
00000011	26	K_6	11
00000010	27	K_6	11
00000001	28	K_6	11
00000000	29	K_6	11
00000011	30	K_6	11
00000010	31	K_6	11
00000001	32	K_6	11
00000000	33	K_6	
00000001000	MACROBLOCK_ESCAPE		

图 4

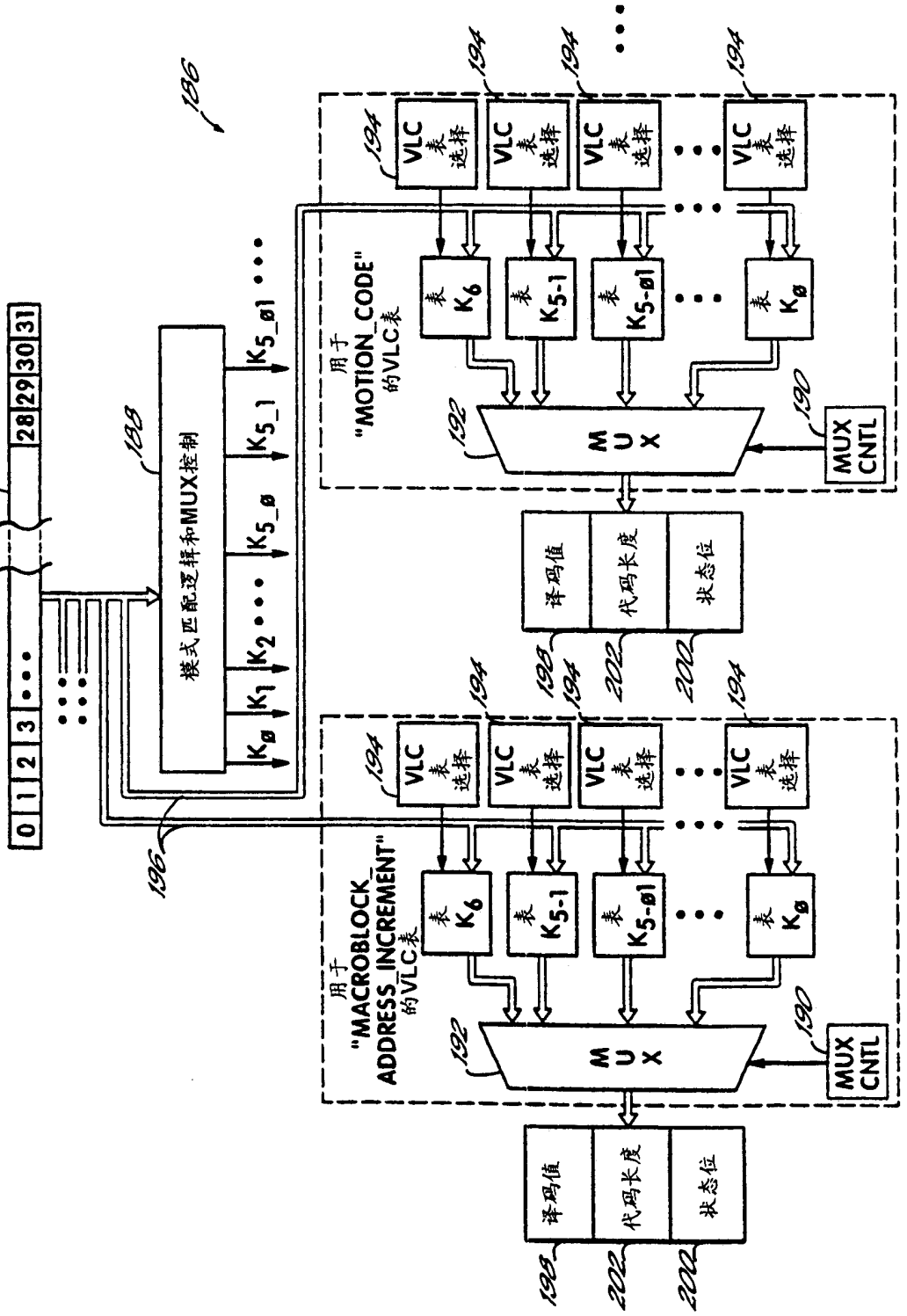

0 1 2 3 28 29 30 31
162
模式匹配逻辑和MUX控制
188
K0 K1 K2 K5_0 K5_1 K5_01
196
用于 "MOTION_CODE" 的VLC表
用于 "MACROBLOCK_ADDRESS_INCREMENT" 的VLC表
VLC表选择
194
表 K6
表 K5-1
表 K5-01
表 K0
MUX
192
MUX CNTL
190
译码值
代码长度
状态位
198
202

图 5

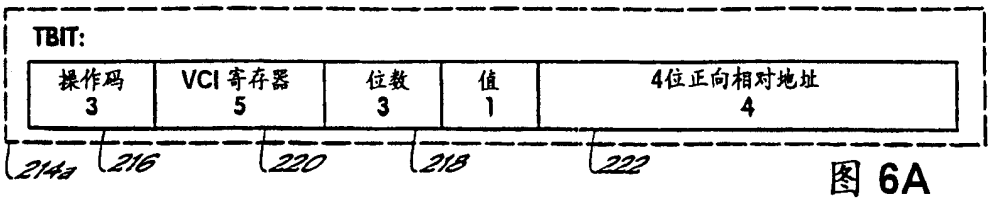


图 6A

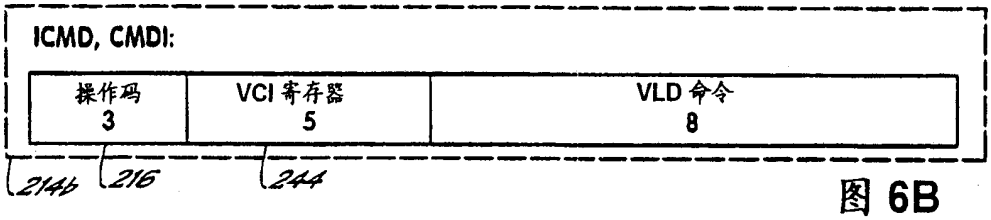


图 6B

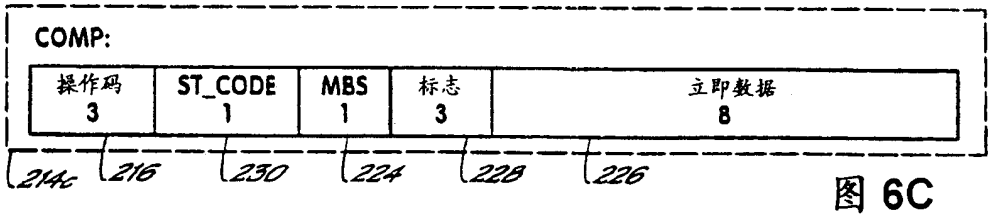


图 6C

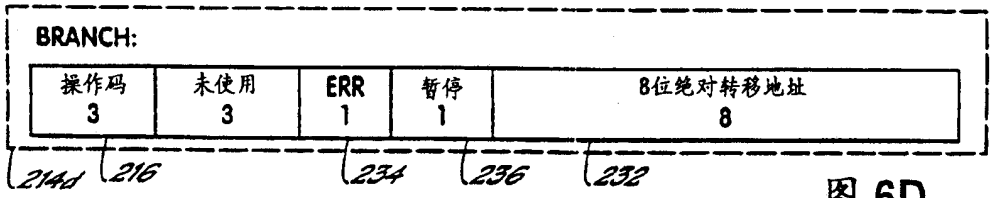


图 6D

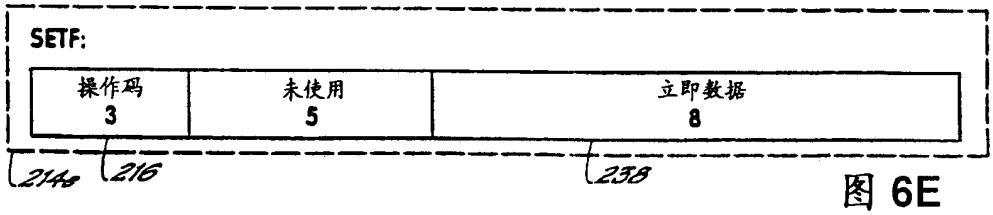


图 6E

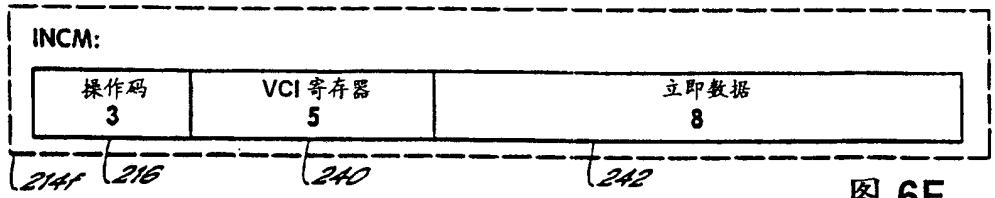


图 6F

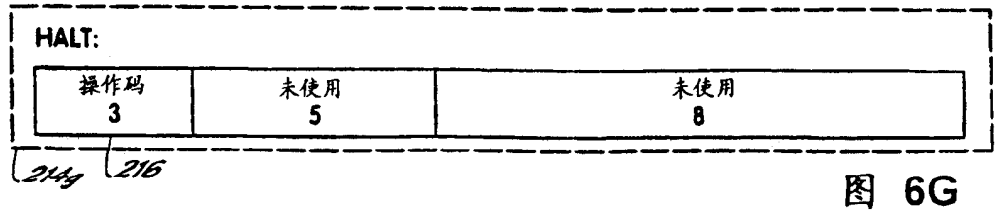


图 6G