[19]中华人民共和国国家知识产权局

[51]Int. Cl[7]
G06F 9/00

[12]发明专利说明书

[21] ZL 专利号 98117663.1

[45]授权公告日 2002 年 12 月 11 日

[11]授权公告号 CN 1096024C

[22]申请日 1998.9.1 [21]申请号 98117663.1

[30]优先权

[32]1997.9.1 [33]JP [31]235625/97

[73]专利权人 松下电器产业株式会社

地址 日本国大阪府

[72]发明人 今西浩 荒木敏之

[56]参考文献

US4084224 1978. 4.11 G06F918

US4807142 1989. 2.21 G06F3153

US4979118 1990.12.18 H04M706

US5353331 1990.10. 4 H04M1100

US5425086 1995. 6.13 H04M1500

US5465335 1995.10. 7 G06F700

US5640563 1997. 6.17 G06F900

审查员 蔡 萍

[74]专利代理机构 中科专利商标代理有限责任公司

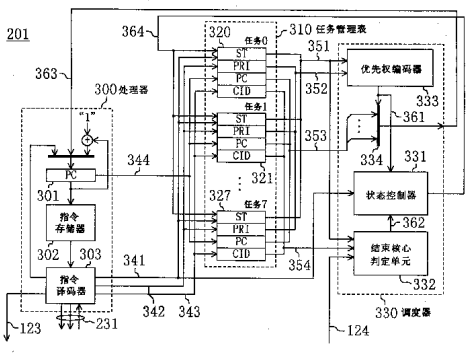
代理人 汪惠民

权利要求书 3 页 说明书 12 页 附图 7 页

[54]发明名称 微控制器、数据处理系统及任务切换控制方法

[57]摘要

一种微控制器，设置有处理器、任务管理表和调度器。处理器顺序执行用来控制被分配的各硬件驱动器（核心）的多个任务。任务管理表存储任务管理信息，包括表示多个任务的各执行状态的状态信息、表示该多个任务的各执行优先顺序的优先权信息以及表示该多个任务分别分配到哪个核心的核心 ID 信息。在特定的指令被译码或者任一核心的执行结束时，调度器根据任务管理信息使处理器进行任务切换。



ISSN1008-4274

1. 一种微控制器，其特征在于包括:

用来根据编程的指令顺序执行多个任务的处理器，上述处理器与多个硬件驱动器一起操作；

用于存储包括（i）表示每个上述任务的执行情况的状态信息、（ii）表示每个上述任务的执行优先权级别的优先权信息、以及（iii）表示向上述多个硬件驱动器分配上述多个任务的分配信息的任务管理信息的任务管理表；和

根据上述任务管理信息使上述处理器在任务之间进行切换的调度器，其中上述多个硬件驱动器中的每一个在上述处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行，如果检测到上述硬件驱动器中的任何一个的结束执行，上述调度器使上述处理器在任务之间进行切换。

2. 根据权利要求 1 所述的微控制器，其特征在于:

每个上述任务可以处在表示执行等待状态的第 1 状态、表示正在运行状态的第 2 状态和表示等待被分配的硬件驱动器结束执行的等待状态的第 3 状态中的一种状态。

3. 根据权利要求 2 所述的微控制器，其特征在于:

上述处理器具有以下功能，即在执行中的任务当中将被分配到该任务的硬件驱动器启动之后对特定的指令进行解码时，更新上述状态信息以使该任务的状态从上述第 2 状态变更为上述第 3 状态。

4. 根据权利要求 2 所述的微控制器，其特征在于:

上述调度器包括判定单元和状态控制器，上述判定单元在任一个硬件驱动器结束执行时，根据上述任务管理信息判分配到该硬件驱动器的任务；上述状态控制器在被上述判定单元启动时，具有更新上述状态信息以使上述被判定的任务的状态从上述第 3 状态变更为上述第 1 状态的功能。

5. 根据权利要求 4 所述的微控制器，其特征在于:

上状态控制器还具有以下功能：在被上述判定单元启动时，更新上述状态信息以使正在执行的任务的状态从上述第 2 状态变更为上述第 1 状态。

6. 根据权利要求 4 所述的微控制器，其特征在于:

上述调度器还包括优先权编码器，该优先权编码器能根据上述任务管理信息，选择上述处于第 1 状态的任务当中执行优先权级别最高的任务作下一个应执行的任务。

7. 根据权利要求 6 所述的微控制器，其特征在于:

上述状态控制器还具有以下功能：更新上述状态信息以使由上述优先权编码器选出的任务的状态从上述第 1 状态变更为上述第 2 状态。

8. 根据权利要求 1 所述的微控制器，其特征在于:

上述任务管理表包括用来保存与在上述任务切换时刻之前所执行的任务有关的上述处理器的资源的区域。

9. 根据权利要求 1 所述的微控制器，其特征在于还包括:

上述多个硬件驱动器分别作为互相独立的工作区而使用的多个寄存器文件。

10. 根据权利要求 1 所述的微控制器，其特征在于还包括:

用来存储上述多个硬件驱动器当中至少 2 个硬件驱动器所共用的设定的参数的寄存器文件。

11. 一种数据处理系统，包括：

用来执行相应的数据处理的多个硬件驱动器；和

用来控制上述多个硬件驱动器的微控制器，其特征在于:

上述微控制器包括根据编程的指令顺序执行多个任务的处理器；

用于存储包括（i）表示每个上述任务的执行情况的状态信息、（ii）表示每个上述任务的执行优先权级别的优先权信息、以及（iii）表示向上述多个硬件驱动器分配上述多个任务的分配信息的任务管理信息的任务管理表；和

根据上述任务管理信息使上述处理器在任务之间进行切换的调度器，其中上述多个硬件驱动器中的每一个在上述处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行，如

果检测到上述硬件驱动器中的任何一个的结束执行，上述调度器使上述处理器在任务之间进行切换。

12. 根据权利要求 11 所述的数据处理系统，其特征在于:

上述多个任务分别具有表示等待执行的第 1 状态、表示正在执行的第 2 状态和表示等待被分配的硬件驱动器结束执行的第 3 状态。

13. 根据权利要求 11 所述的数据处理系统，其特征在于:

上述多个硬件驱动器分别是用来编码 MPEG 图像数据的部分处理核心。

14. 一种任务切换控制方法，包括向对应的硬件驱动器分配一个或多个任务，由调度器根据有关该任务/硬件驱动器分配的信息控制任务切换，其特征在于:

上述多个硬件驱动器中的每一个在处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行；

每个上述任务可以处在表示执行等待状态的第 1 状态、表示正在运行状态的第 2 状态、和表示等待被分配的硬件驱动器结束执行的等待状态的第 3 状态中的一种状态；和

在任何一个上述硬件驱动器的执行结束时，上述调度器将分配到上述结束执行的硬件驱动器的任务的状态从上述第 3 状态变更为上述第 1 状态，以便允许上述处理器在任务之间切换。

15. 根据权利要求 14 所述的任务切换控制方法，其特征在于:

在硬件驱动器结束执行时，将执行中的任务的状态从上述第 2 状态变更为上述第 1 状态。

微控制器、数据处理系统及任务切换控制方法

技术领域

本发明涉及具有多任务功能的微控制器、由控制多个硬件驱动器的该微控制器而构成的数据处理系统以及任务切换控制方法。

背景技术

具有多任务功能的微控制器已被我们所知晓。藏在这种微控制器内的单一处理器顺序执行多个任务。为此，任务定时器定期发出要求任务切换的定时器中断信号。每次处理器接收该定时器中断信号，操作系统(OS)中的中断处理程序被启动，从而该中断处理程序进行任务调度和资源的保存及恢复。

上述以往的微控制器由于在中断处理程序中进行任务调度，所以存在着任务切换时的开销(overhead)大，微控制器实质上的工作效率下降的问题。此问题在图像数据编码等重视实时性的应用中特别严重。

发明内容

本发明的目的是实现微控制器的高速任务切换。

本发明的另一目的是在由控制多个硬件驱动器的微控制器构成的数据处理系统中，实现该微控制器的高速任务切换。

本发明的其他目的在于提供为实现高速任务切换的任务切换控制方法。

根据本发明，提供一种微控制器，包括用来根据编程的指令顺序执行多个任务的处理器，上述处理器与多个硬件驱动器一起操作；用于存储包括（i）表示每个上述任务的执行情况的状态信息、（ii）表示每个上述任务的执行优先权级别的优先权信息、以及（iii）表示向上述多个硬件驱动器分配上述多个任务的分配信息的任务管理信息的任务管理表；

和根据上述任务管理信息使上述处理器在任务之间进行切换的调度器，其中上述多个硬件驱动器中的每一个在上述处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行，如果检测到上述硬件驱动器中的任何一个的结束执行，上述调度器使上述处理器在任务之间进行切换。

根据本发明，提供一种数据处理系统，包括用来执行相应的数据处理的多个硬件驱动器；和用来控制上述多个硬件驱动器的微控制器，上述微控制器包括根据编程的指令顺序执行多个任务的处理器；用于存储包括（i）表示每个上述任务的执行情况的状态信息、（ii）表示每个上述任务的执行优先权级别的优先权信息、以及（iii）表示向上述多个硬件驱动器分配上述多个任务的分配信息的任务管理信息的任务管理表；和根据上述任务管理信息使上述处理器在任务之间进行切换的调度器，其中上述多个硬件驱动器中的每一个在上述处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行，如果检测到上述硬件驱动器中的任何一个的结束执行，上述调度器使上述处理器在任务之间进行切换。

根据本发明，提供一种任务切换控制方法，包括向对应的硬件驱动器分配一个或多个任务，由调度器根据有关该任务/硬件驱动器分配的信息控制任务切换，上述多个硬件驱动器中的每一个在处理器启动时开始执行数据处理，如果上述数据处理结束，向上述调度器通知该结束执行；每个上述任务可以处在表示执行等待状态的第 1 状态、表示正在运行状态的第 2 状态、和表示等待被分配的硬件驱动器结束执行的等待状态的第 3 状态中的一种状态；和在任何一个上述硬件驱动器的执行结束时，上述调度器将分配到上述结束执行的硬件驱动器的任务的状态从上述第 3 状态变更为上述第 1 状态，以便允许上述处理器在任务之间切换。

为达到上述目的，本发明的微控制器采用的做法是不在中断处理程序中控制任务切换而在多个任务被分配到与之相对应的硬件驱动器的环境下，根据表示该分配的信息用硬件调度程度(hardware scheduler，以下称调度器")控制任务切换。在多个硬件驱动器当中，有的执行时间上关键的处理，又有的不执行那种处理。若按照本发明，这种多个硬件驱动

器彼此之间的关系反映到多个任务各自的执行优先顺序，因此不须在任务切换时重新判定哪个硬件驱动器执行时间上关键的处理，就能在短时间里选择下一个应该执行的任务。就是说，任务切换时的开销变小，高速的任务切换得以实现。

另外，鉴于在响应从任务定时器定期发出的中断信号来进行任务切换的分时操作方式中会发生时间损耗，本发明的微控制器采用立即响应各硬件驱动器的执行结束这事件发生而进行任务切换的事件驱动方式。多个任务分别包括表示等待执行的第 1 状态(READY state)、表示正在执行的第 2 状态(ACTIVE state)以及表示等待所分配的硬件驱动器结束执行的第 3 状态(SLEEP state)。ACTIVE 状态是任务正在使用微控制器的状态，其中对被分配到该任务的硬件驱动器进行控制。READY 状态虽是任务能使用微控制器的状态，但是该任务还没被选择而正在等待被选择的状态。SLEEP 状态是等待被分配的硬件驱动器结束执行，即不能使用微控制器的状态。启动了所分配的硬件驱动器的任务响应特定的指令(tash-sleep 指令)从 ACTIVE 状态转移到 SLEEP 状态。在某一硬件驱动器执行结束时，人 该硬件驱动器的任务从 SLEEP 状态转移到 READY 状态，正在执行中的任务从 ACTIVE 状态转移到 READY 状态。之后，处于 RDADY 状态的任务当中执行优先权高最的任务被选择为下一个应执行的任务，从而该被选择的任务从 READY 状态转移到 ACTIVE 状态。

如果在微控制器中准备好多个寄存器文件以作为多个硬件驱动器分别互相独立的工作区，那么在任务切换时只要保存程序计数器等处理器资源即可，因此任务切换时的开销能更小。在微控制器中准备好存储多个硬件驱动器共用的设定参数的寄存器文件也可。

附图说明

图 1 是本发明所涉及的 MPEG 图像编码器的结构示例方框图。

图 2 是表示图 1 中的微控制器的详细结构的方框图。

图 3 是表示图 2 中的任务控制器的详细结构的方框图。

图 4 是表示图 1 中的编码器的核心和任务的对应关系的概念图。

图 5 是表示图 1 中的编码器的任务状态转移情况的概念图。

图 6 是表示由图 1 中的 5 个核心进行的宏块流水线处理的时序图。

图 7 是在图 6 的部分期间内 3 个任务的各状态转移的具体示例时序图。

具体实施方式

下面，参照附图详细说明本发明。

图 1 示出本发明所涉及的数据处理系统之一的 MPEG (Moving Picture Experts Group)图像编码器的结构例。图 1 的编码器由一个微控制器 101、构成宏块流水线的 5 个硬件驱动器(以下称“核心”)111~115 和 3 个缓冲存储器 116~118 构成。5 个核心则是运动检测器(Motion Detector：MD)111、运动补偿器(Motion Compensator：MC)112、离散余弦变换器(Discrete Cosine Transformer：DCT)113、量化器（Quantizer：Q）114 以及可变长度编码器（Variable Length Coder：VLC）115，分别由具有多任务功能的微控制器 101 控制。121 是要被编码的图像数据，122 是表示编码结果的符号化数据。微控制器 101 将启动信号 123 分别提供给 5 个核心 111~115，并从 5 个核心 111~115 中分别接收结束信号 124。另外，微控制器 101 的构成为能通过信号线 131~135 分别与 5 个核心 111~115 传送信号，又能通过信号线 136 将共用参数提供给 5 个核心 111~115。

图 2 示出微控制器 101 的详细结构。微控制器 101 包括用来实现多任务功能的任务控制器 201、分别作为上述 5 个核心 111~115 的互相独立的工作区的 5 个核心寄存器文件 211~215、一个用来存储上述 5 个核心 111~115 当中至少有 2 个核心所共用的设定参数的共用寄存器文件 216、一个作为任务控制器 201 工作区而被使用的通用寄存器文件 217、乘法器 221、移位器 222、算术逻辑运算单元（Arithmetic and Logic Unit：ALU）223 以及数据存储器 224。241 是 A 总线，242 是 B 总线。243C 是总线，231 是将这些总线和任务控制器 201 连接的信号线。任务控制器 201 提供上述启动信号 123 并接收上述结束信号 124。寄存器文件 211~216 分别介于 C 总线 243 和上述与之相对应的信号线 131~136 之间，并且其各两个输出分别与 A 总线 241 和 B 总线 242 相连。通用寄存器文件 217 和数据存储器 224 的各输入与 C 总线 243 相连，其各两个输出与

A 总线 241 和 B 总线 242 相连。乘法器 221、移位器 222 和 ALU223 的各两个输入与 A 总线 241 和 B 总线 242 相连，其各输出与 C 总线 243 相连。另外，不设 5 个核心寄存器文件 211~215 和共用寄存器文件 216 而从 C 总线 243 直接引出上述信号线 131~136 也可。

若按照图 1 的 MPEG 图像编码器，以 16×16 像素构成的宏块为单位进行图像数据处理。首先，在 MD 核心 111 中对所输入的图像数据 121 求出备选的运动向量。在 MC 核心 112 中使用这些运动向量求出图像的差分数据，并选出最佳运动向量。在 DCT 核心 113 中对被选出的运动向量的差分数据进行离散余弦变换，经过 Q 核心 114 被量化，并在 VLC 核心 115 中和求得的运动向量等附属信息一起被编成可变长度代码之后，作为符号化数据 122 被输出。

下面，参照图 2 详细说明。首先，任务控制器 201 通过信号线 231、ALU223 和 C 总线 243 将工作参数设定在 MD 核心寄存器文件 211 中，由启动信号 123 启动 MD 核心 111。MD 核心 111 从 MD 核心寄存器文件 211 中通过信号线 131 读入工作参数，同时输入图像数据 121。MD 核心 111 的执行结束之后，求得的备选运动向量通过信号线 131 被写入 MD 核心寄存器文件 211，从 MD 核心 111 输出结束信号 124。任务控制器 201 接收该结束信号 124 之后，从 MD 核心寄存器文件 211 读出备选运动向量，并根据它使用乘法器 221、移位器 222、ALU223 和通用寄存器文件 217 求出 MC 核心 112 所用的工作参数。该工作参数被设定于 MC 核心寄存器文件 212，从而由启动信号 123 启动 MC 核心 112。MC 核心 112 从 MC 核心寄存器文件 212 中通过信号线 132 读入工作参数之后，求出图像的差分数据。MC 核心 112 的执行结束后，差分数据的总和通过信号线 132 被写入 MC 核心寄存器文件 212，图像的差分数据被写入缓冲存储器 116，并从 MC 核心 112 输出结束信号 124。任务控制器 201 接收该结束信号 124 后，从 MC 核心寄存器文件 212 读出差分数据的总和，并根据它使用乘法器 221、移位器 222、ALU223 和通用寄存器文件 217 从上述备选向量中选出最佳运动向量。与求得的运动向量对应的差分数据的地址被设定在 DCT 核心寄存器文件 213 中，由启动信号 123 启动 DCT 核心 113。DCT 核心 113 根据设定于 DCT 核心寄存器文件 213 的地址从

缓冲存储器 116 读出差分数据，并对此进行离散余弦变换。DCT 核心 113 的执行结束后，离散余弦变换的结果被写入缓冲存储器 117，从 DCT 核心 113 输出结束信号 124。然后，在 Q 核心 114 中进行量化处理，其结果被写入缓冲存储器 118 中，并在 VLC 核心 115 中进行可变长度编码处理，其结果作为符号化数据 122 被输出。另外，在 1 个宏块的处理期间，上述 5 个核心 111～115 中有几个核心与微控制器 101 之间多次进行启动信号 123 和结束信号 124 的传输。共用寄存器文件 216 被用于预先向 5 个核心 111～115 提供切换 MPEG1 和 MPEG2 时所需的共用参数，并预先向 MD 核心 111 和 MC 核心 112 提供用来指定运动评价模式的共用参数。

图 3 示出任务控制器 201 的详细结构。任务控制器 201 由处理器 300、任务管理表 310 和调度器 330 构成。处理器 300 是最多能顺序执行 8 个任务的 RISC（Reduced Instruction Set Computer）型处理器，具有用来生成指令地址的程序计数器（PC）301、用来存储由一连串指令构成的程序的指令存储器 302 和用来译码指令的指令译码器 303。要提供给各核心的启动信号 123 由指令译码器 303 提供。指令译码器 303 通过信号线 231 与指令执行用资源即上述乘法器 221、移位器 222、ALU223 等相连。任务管理表 310 是用来存储任务管理信息的电路块，具有分别对应于从任务 0 到任务 7 的 8 个任务的 8 个任务管理寄存器文件 320～327。在此，任务管理信息包括表示多个任务的各执行情况的状态信息（ST 信息）、表示该多个任务的各执行优先权的优先权信息（PRI 信息）和表示该多个任务分别被分配到 5 个核心 111～115 中的哪一核心的核心 ID 信息（CID 信息）。此外，在任务管理表 310 中每一个任务都包括为保存处理器 300 的资源即 PC301 的内容的区域。在此保存区域中也能保存有关 ALU223（参照图 2）的运算结果的标记（flag）等。调度器 330 是根据任务管理表 310 所存储的任务管理信息使处理器 300 进行任务切换所用的电路块，具有状态控制器 331、结束核心判定单元 332、优先权编码器 333 和选择器 334。结束核心判定单元 332 是在从 5 个核心 111～115 中的任一核心接收结束信号 124 时，为判定被分配到该执行结束了的核心中的任务的单元。该判定参照任务管理表 310 而进行，表示判定结果的任务号码 362

被通知给状态控制器 331。优先权编码器 333 是用来选择其次应该执行的任务的电路块。该选择参照任务管理表 310 而进行，表示选择结果的任务号码 361 被通知给状态控制器 331 和选择器 334。状态控制器 331 是用来更新任务管理表 310 中的 ST 信息的电路块。选择器 334 控制向处理器 300 的资源恢复。

图 4 示出图 1 的 MPEG 图像编码器中的核心和任务的对应关系。在此，微控制器 101 执行 6 个任务 400～405。任务 400 是用来控制低位阶层的 5 个任务 401～405 且管理整个编码处理的主任务（任务 0）。不存在应被分配到该主任务 400 的核心。任务 401 是对被分配的 MD 核心 111 进行控制的运动检测任务（任务 1）。任务 402 是对被分配的 MC 核心 112 进行控制的运动补偿任务（任务 2）。任务 403 是对被分配的 DCT 核心 113 进行控制的离散余弦变换任务（任务 3）。任务 404 是对被分配的 Q 核心 114 进行控制的量化任务（任务 4）。任务 405 是对被分配的 VLC 核心 115 进行控制的可变长度编码任务（任务 5）。

在此，假设有关图 4 所示的至少 6 个任务 400～405 的任务管理信息被存储在图 3 中的任务管理表 310 中，如图 3 所示，PRI 信息响应优先权设定信号 342 而被设定，CID 信息响应核心设定信号 343 而被设定。在指令译码器 303 译码优先权设定指令时，优先权设定信号 342 从指令译码器 303 被提供给任务管理表 310。在指令译码器 303 译码核心设定指令时，核心设定信号 343 从指令译码器 303 被提供给任务管理表 310。

图 5 是表示各任务的状态转移情况的概念图。任务具有表示停止的 STOP 状态、表示等待执行的 READY 状态、表示执行中的 ACTIVE 状态以及表示等待所被分配的核心结束执行的 SLEEP 状态。但，在任务 0 中不存在 SLEEP 状态。刚复位了的任务处于 STOP 状态。处于 STOP 状态的任务由 task_ready 指令被转移到 READY 状态（转移 501）。如果在要求任务切换的事件发生时由调度器 330 选择处于 READY 状态的任务，该任务被转移到 ACTIVE 状态（转移 511）。此时，到此刻为止处于 ACTIVE 状态的任务由调度器 330 被转移到 READY 状态（转移 522）。处于 ACTIVE 状态的任务由处理器 300 执行，该任务能由 task_sleep 指令被转移到 SLEEP 状态（转移 521），又能由 task_stop 指令被转移到 STOP 状态（转

移 523）。处于 SLEEP 状态的任务在所分配的核心结束执行时，被转移到 READY 状态（转移 531）。

下面，详细说明图 3 的任务控制器 201 的动作。指令译码器 303 译码 task_ready 指令、task_sleep 指令或者 task_stop 指令时，就发生任务切换。比方说，在正在执行的任务中完成了被分配到该任务的核心的工作参数的设定并完成了该核心的启动的任务，由 task_sleep 指令从 ACTIVE 状态被转移到 SLEEP 状态。另外，在 5 个核心 111～115 中的任一核心结束执行时也发生任务切换。任务切换时的任务控制器 201 的动作顺序为：（1）启动调度器、（2）保存正在执行的任务资源、（3）选择其次应执行的任务、（4）恢复所保存的资源。

首先，说明根据指令的任务切换顺序。

（A-1）启动调度器

task_ready 指令、task_sleep 指令或者 task_stop 指令被译码时，指令译码器 303 输出状态变更信号 341。状态变更信号 341 被输入状态控制器 331。结果，调度器 330 被启动。

（A-2）保存正在执行的任务资源

状态变更信号 341 又被输入到任务管理表 310 中，ST 信息被更新。同时，到此时为止被执行了的任务的 PC301 值通过信号线 344 被保存在任务管理表 310 中。

（A-3）选择其次应执行的任务

优先权编码器 333 从任务管理表 310 通过信号线 351 接收 ST 信息，并通过信号线 352 接收 PR1 信息，从而将处于 READY 状态的任务当中执行优先权级别最高的任务作为其次应执行的任务选择。表示该选择结果的任务号码 361 被通知给状态控制器 331 和选择器 334。

（A-4）恢复所保存的资源

状态控制器 331 将对应于任务号码 361 的状态变更信号 364 提供给任务管理表 310。结果，由优先权编码器 333 选出的任务的 ST 信息被从 READY 状态更新为 ACTIVE 状态。选择器 334 从任务管理表 310 通过信号线 353 读出由任务号码 361 指定的任务的 PC，然后将该 PC 提供给信号线 363，结果，其次应执行的任务的 PC 值被设定在处理器 300 中，

从而开始执行该任务。

其次，说明核心结束执行时而发生的任务切换的顺序。

（B-1）启动调度器

任一核心结束执行时，结束信号 124 被输入到结束核心判定单元 332。结束核心判定单元 332 根据结束信号 124 判定哪个核心结束执行。此外，结束核心判定单元 332 通过信号线 354 读出任务管理表 310 中的 CID 信息，并判定哪个任务被分配到执行结束了的核心。在从 ST 信息能确认出该任务处于 SLEEP 状态时，表示该判定结果的任务号码 362 被通知给状态控制器 331。结果，调度器 330 被启动。状态控制器 331 将对应于任务号码 362 的状态变更信号 364 提供给任务管理表 310。结果，执行结束了的任务的 ST 信息被从 SLEEP 状态更新为 READY 状态。另外，没有任务分配给结束了执行的核心时，调度器 330 不被启动。

（B-2）保存执行中的任务资源

并且，状态控制器 331 将状态变更信号 364 提供给任务管理表 310 以便在这以前执行了的任务的 ST 信息能被从 ACTIVE 状态更新为 READY 状态。同时，在这以前执行了的任务的 PC301 值被保存在任务管理表 310 中。

（B-3）选择其次应执行的任务

优先权编码器 333 从任务管理表 310 接收 ST 信息和 PRI 信息，并将处于 READY 状态的任务当中执行优先权级别最高的任务作为其次应执行的任务选择。表示该选择结果的任务号码 361 被通知给状态控制器 331 和选择器 334。

（B-4）恢复所保存的资源

状态控制器 331 将对应于任务号码 361 的状态变更信号 364 提供给任务管理表 310。结果，由优先权编码器 333 选择了的任务的 ST 信息被从 READY 状态更新为 ACTIVE 状态。选择器 334 从任务管理表 310 中读出由任务号码 361 指定的任务的 PC，并将该 PC 提供给处理器 300。结果，其次应执行的任务的 PC 值被设定在处理器 300 中，从而开始执行该任务。

图 6 示出由图 1 中的 5 个核心 111～115 进行的宏块流水线处理的情

况。流水线间距被设定为在各核心中 1 个宏块的处理所需的时间的最大值。于是，其特性为在各流水线间距期间中都存在着比其他核心早结束执行的核心。在此，产生闲置时间。并且，还有闲置时间依据图像数据而发生变化的特性。在图 1 的示例中，通过采用事件驱动方式的任务切换，适合这些特性的 MPEG 图像编码器得以实现。另外，各流水线间距期间的核心的启动次数依在该核心中进行的处理的内容和数据而异。比方说，DCT 核心 113 在 1 个流水线间距期间被启动 1 次。另一方面，由于 MC 核心 112 将 1 个宏块数据分成亮度成分和色差成分并对每个成分执行细分化处理，因此其在 1 个流水线间距期间中响应数据而被启动多次。

图 7 示出在图 6 中由虚线所特定的部分期间的 3 个任务各自的状态转移具体例。任务 0 是为管理整个编码处理的主任务，任务 1 是分配到 MD 核心 111 的任务，任务 2 是分配到 MC 核心 112 的任务（参照图 4）。假设这 3 个任务当中任务 1 的执行优先权级别最高，任务 2 的执行优先权级别仅次于任务 1，任务 0 的执行优先权级别最低。另外，在时刻 t0，假设任务 1 处于 ACTIVE 状态，任务 0 和任务 2 分别处于 READY 状态。

如图 7 所示，在时刻 t1～t7 分别发生任务切换。该图中的△t 表示在一次任务切换中的开销。下面，按顺序进行说明。到时刻 t1 之前，任务 1 启动 MD 核心 111。之后，在时刻 t1 收到 task_sleep 指令，任务 1 从 ACTIVE 状态转移到 SLEEP 状态。此时，由于任务 0 和任务 2 都处于 READY 状态，而且任务 2 的执行优先权级别比任务 0 高，所以任务 2 从 READY 状态转移到 ACTIVE 状态。任务 2 启动 MC 核心 112。然后，在时刻 t2 收到 task_sleep 指令，任务 2 从 ACTIVE 状态转移到 SLEEP 状态。此时，只有任务 0 处于 READY 状态，因此任务 0 从 READY 状态转移到 ACTIVE 状态。在时刻 t3，由于 MC 核心 112 执行结束，任务 2 从 SLEEP 状态转移到 READY 状态。与此同时，在这以前处于 ACTIVE 状态的任务 0 转移到 READY 状态。此时，由于任务 0 和任务 2 都处于 READY 状态，而且任务 2 的执行优先权级别比任务 0 高，所以任务 2 从 READY 状态转移到 ACTIVE 状态。任务 2 重新启动 MC 核心 112。在时刻 t4，收到 task_sleep 指令的任务 2 从 ACTIVE 状态转移到 SLEEP 状态。此时，只

有任务 0 处于 READY 状态，因此任务 0 从 READY 状态转移到 ACTIVE 状态。在时刻 t5，由于 MD 核心 111 执行结束，任务 1 从 SLEEP 状态转移到 READY 状态。与此同时，在这以前处于 ACTIVE 状态的任务 0 转移到 READY 状态。此时，由于任务 0 和任务 1 处于 READY 状态，而且任务 1 的执行优先权级别比任务 0 高，所以任务 1 从 READY 状态转移到 ACTIVE 状态。在时刻 t6，由于 MC 核心 112 执行结束，任务 2 从 SLEEP 状态转移到 READY 状态。与此同时，在这以前处于 ACTIVE 状态的任务 1 转移到 READY 状态。此时，由于任务 0、任务 1 和任务 2 都处于 READY 状态，而且任务 1 的执行优先权级别比任务 0 和任务 2 的各执行优先权级别高，所以任务 1 从 READY 状态返回到 ACTIVE 状态。任务 1 重新启动 MD 核心 111。然后，在时刻 t7 收到 task_sleep 指令，任务 1 从 ACTIVE 状态转移到 SLEEP 状态。此时，由于任务 0 和任务 2 处于 READY 状态，而且任务 2 的执行优先权级别比任务 0 高，所以任务 2 从 READY 状态转移到 ACTIVE 状态。

在采用以往的使用中断处理程序的分时操作方式进行任务切换时，在一次任务切换中的开销要十几个机器周期，但采用本发明所涉及的事件驱动方式进行任务切换时，能将图 7 中所示的开销△t 抑制为几个机器周期。若考虑到在图 6 中的各宏块流水线间距期间中最多发生 20 几次任务切换，上述两种方式之间的开销差会更大。由本发明而实现的开销缩小也能使流水线间距缩小。即能达成图像数据的高速编码处理。

综上所述，图 1 的 MPEG 图像编码器能实现高速的任务切换。另外，通过将高执行优先权设定在将要执行时间上关键处理的任务中，能保证正常的图像编码处理。并且，由于所采用的构成为：在任一核心结束执行时，将被分配到该结束了执行的核心的任务的状态和在此刻以前处于执行中的任务的状态都变更为 READY 状态，然后，将处于 READY 状态的所有的任务当中执行优先权最高的任务作为其次该执行的任务选择，因此能简化优先权编码器 333 的内部结构。并且，对每个任务都能独立地描述程序，所以能提高编程效率，又有利于调试。

另外，本发明也能适用于图像译码器等其他的数据处理系统。在上述示例中，对所有的硬件驱动器（核心）各分配了一个任务。可是，有

分配不到任务的核心也可，将多个任务分配给 1 个核心也可。但，没有同时将 1 个任务分配给多个核心的情况。

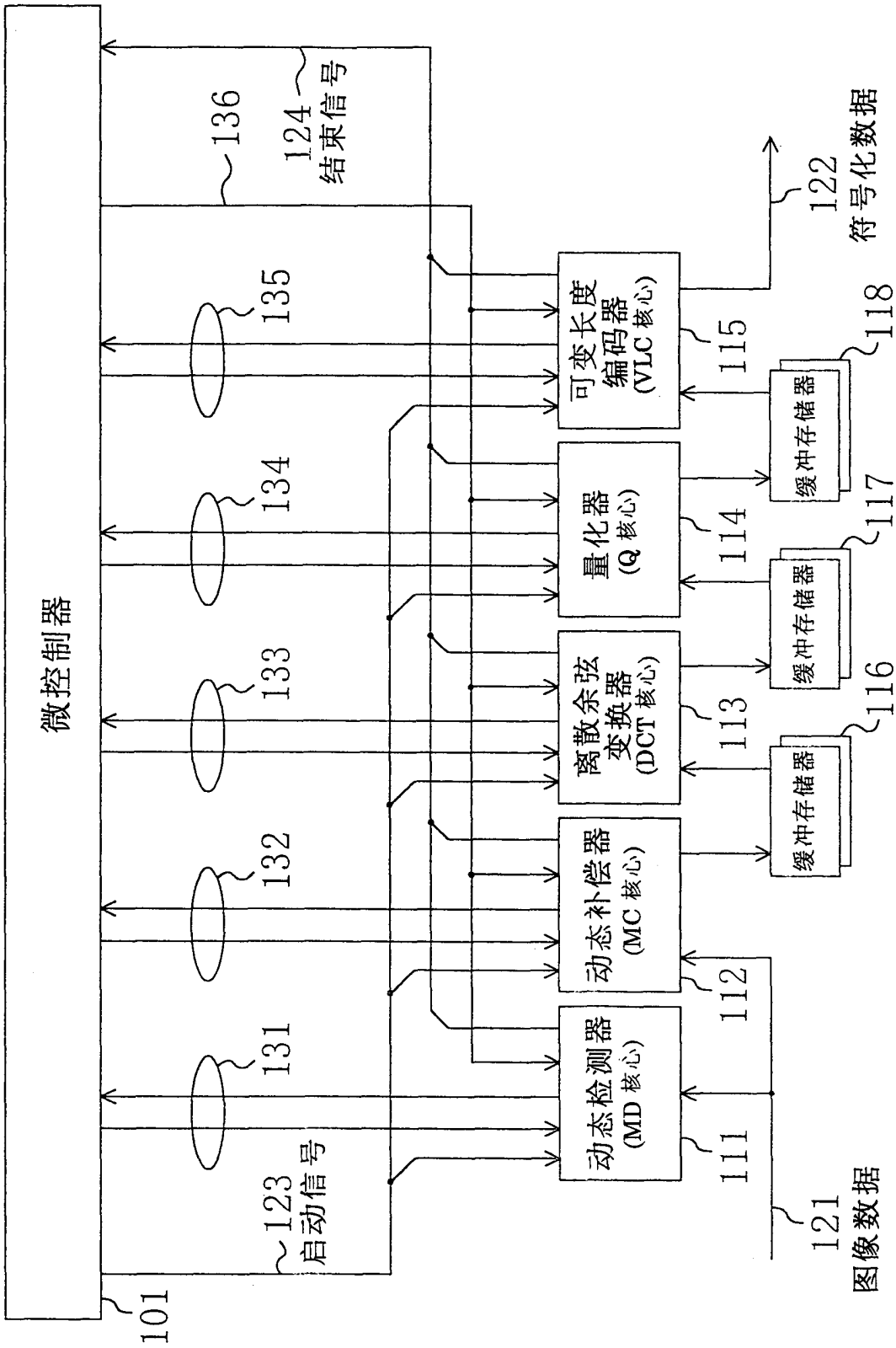

微控制器
101
123
启动信号
131
132
133
134
135
136
124
结束信号
动态检测器
(MD 核心)
111
动态补偿器
(MC 核心)
112
离散余弦
变换器
(DCT 核心)
113
量化器
(Q 核心)
114
可变长度
编码器
(VLC 核心)
115
缓冲存储器
116
缓冲存储器
117
缓冲存储器
118
121
图像数据
122
符号化数据

图 1

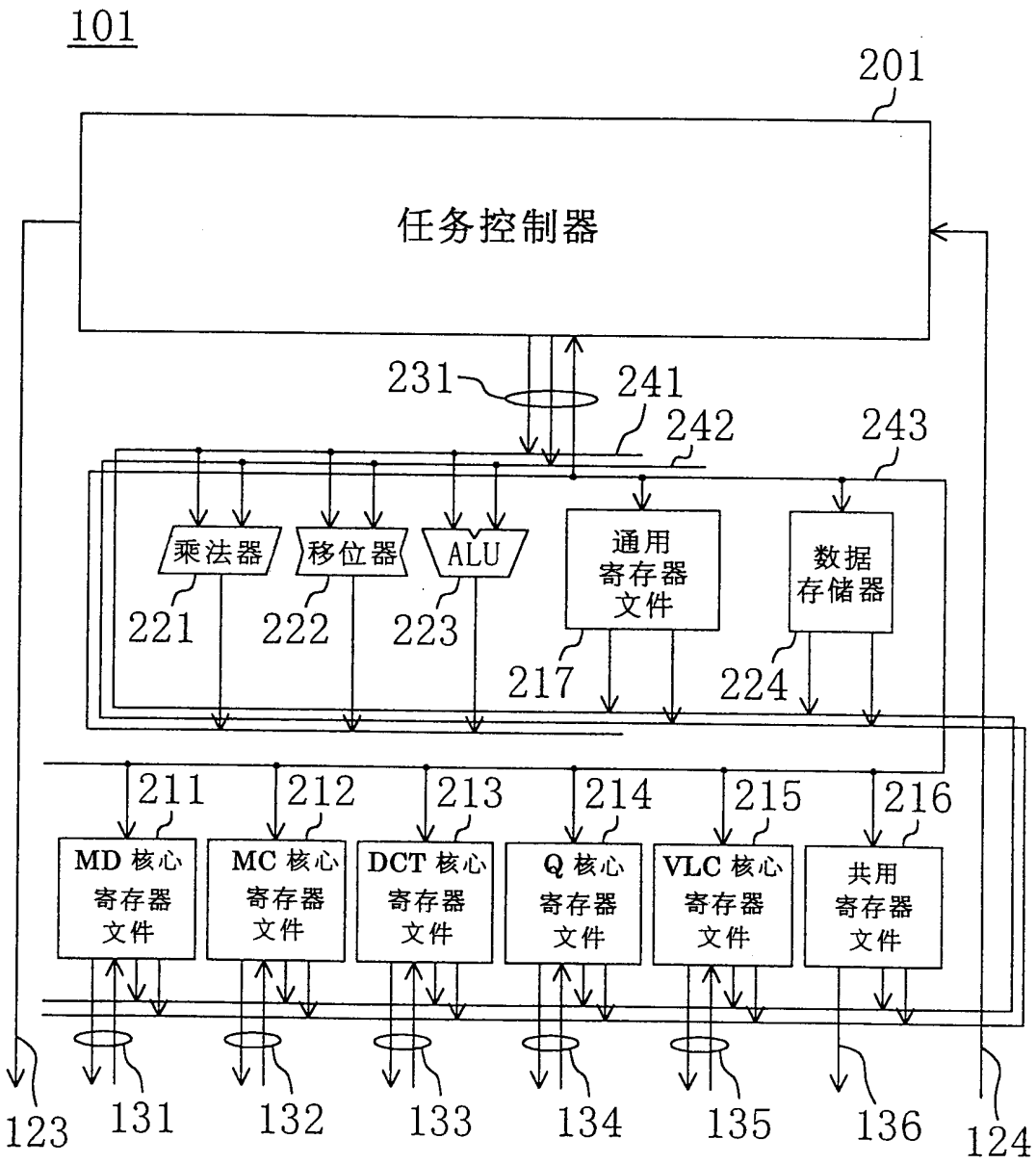

101
201
任务控制器
231
241
242
243
乘法器
移位器
ALU
通用
寄存器
文件
数据
存储器
221
222
223
217
224
211
212
213
214
215
216
MD 核心
寄存器
文件
MC 核心
寄存器
文件
DCT 核心
寄存器
文件
Q 核心
寄存器
文件
VLC 核心
寄存器
文件
共用
寄存器
文件
123
131
132
133
134
135
136
124

图 2

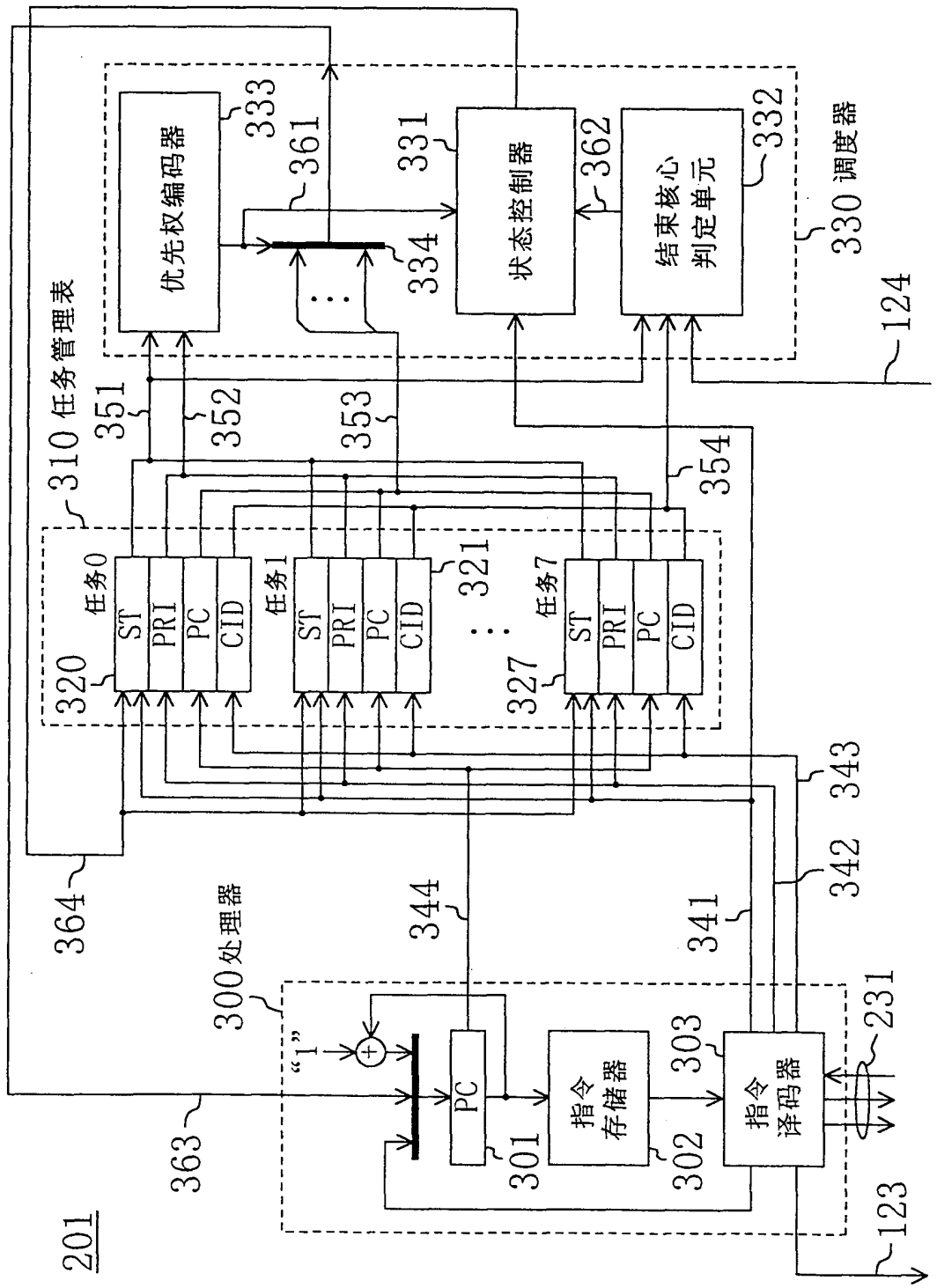

201
300处理器
310任务管理表
320
321
327
330调度器
331
332
333
334
341
342
343
344
351
352
353
354
361
362
363
364
123
124
231
301
302
303
任务0
任务1
任务7
ST
PRI
PC
CID
"1"
优先权编码器
状态控制器
结束核心
判定单元
指令
存储器
指令
译码器

图 3

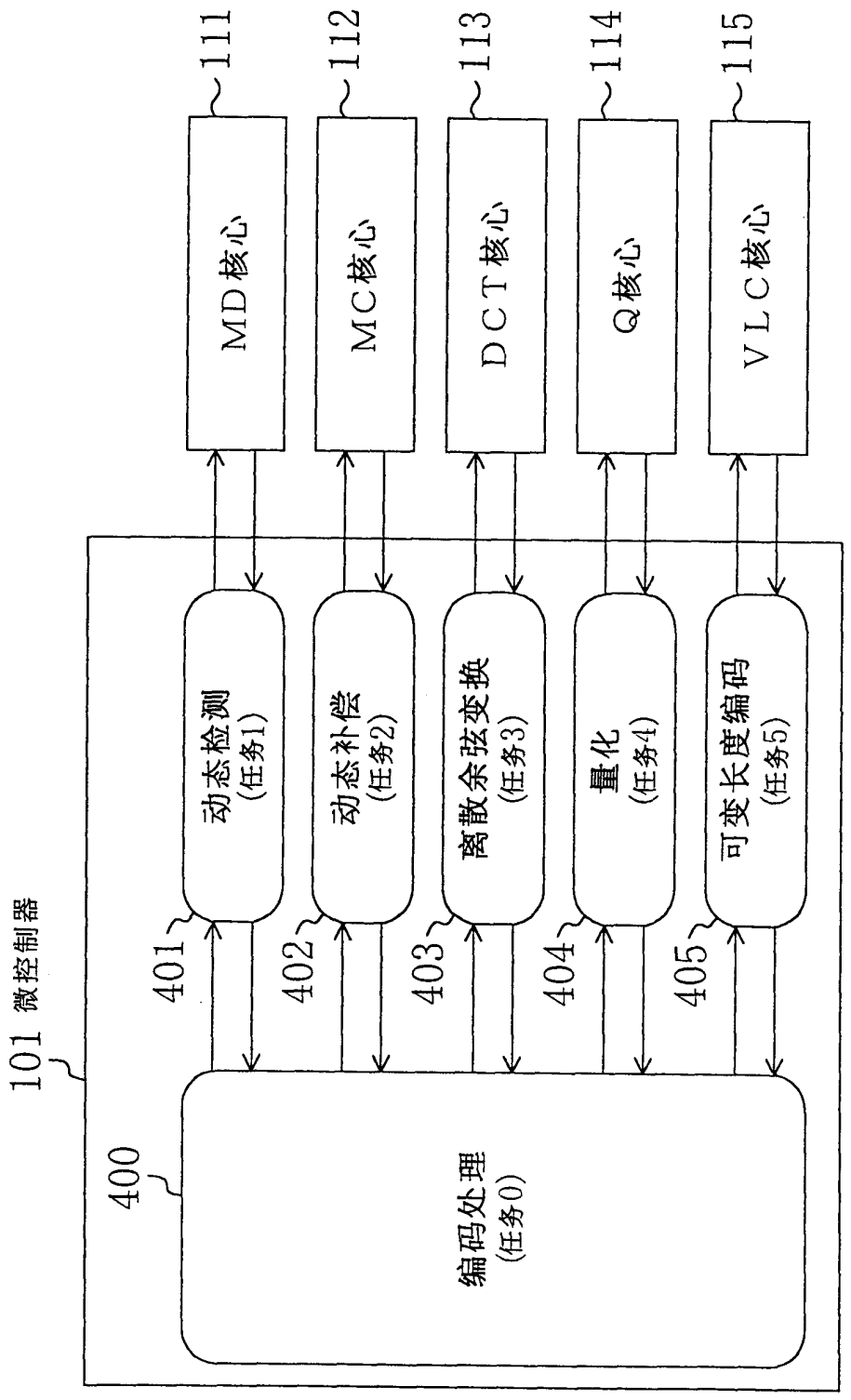

111
112
113
114
115
MD核心
MC核心
DCT核心
Q核心
VLC核心
101 微控制器
400
编码处理
(任务0)
401
动态检测
(任务1)
402
动态补偿
(任务2)
403
离散余弦变换
(任务3)
404
量化
(任务4)
405
可变长度编码
(任务5)

图 4

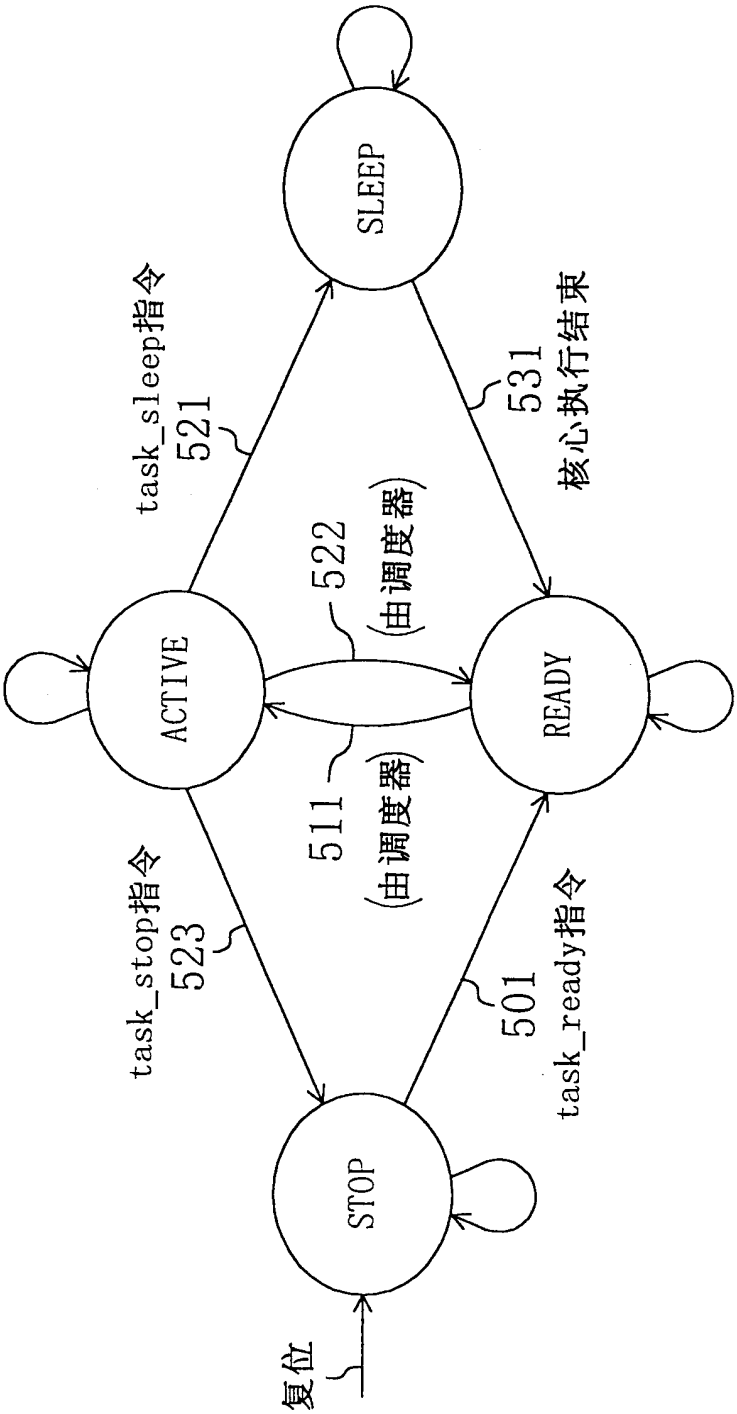

SLEEP
ACTIVE
READY
STOP
复位
task_sleep指令
521
task_stop指令
523
511
(由调度器)
522
(由调度器)
531
核心执行结束
501
task_ready指令

图 5

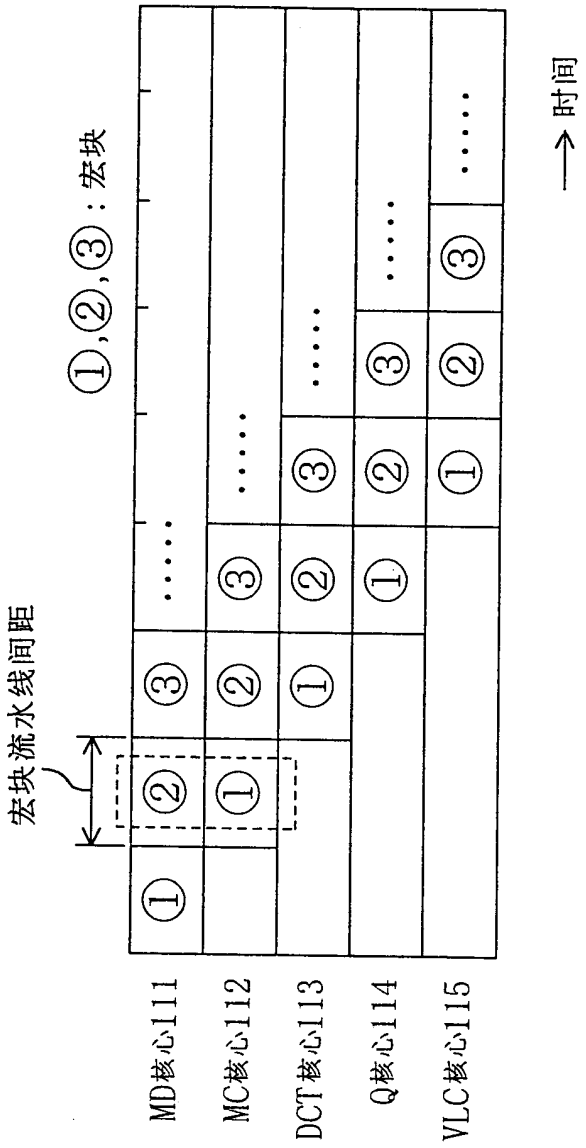

①,②,③：宏块
宏块流水线间距
MD核心111
MC核心112
DCT核心113
Q核心114
VLC核心115
时间

图 6

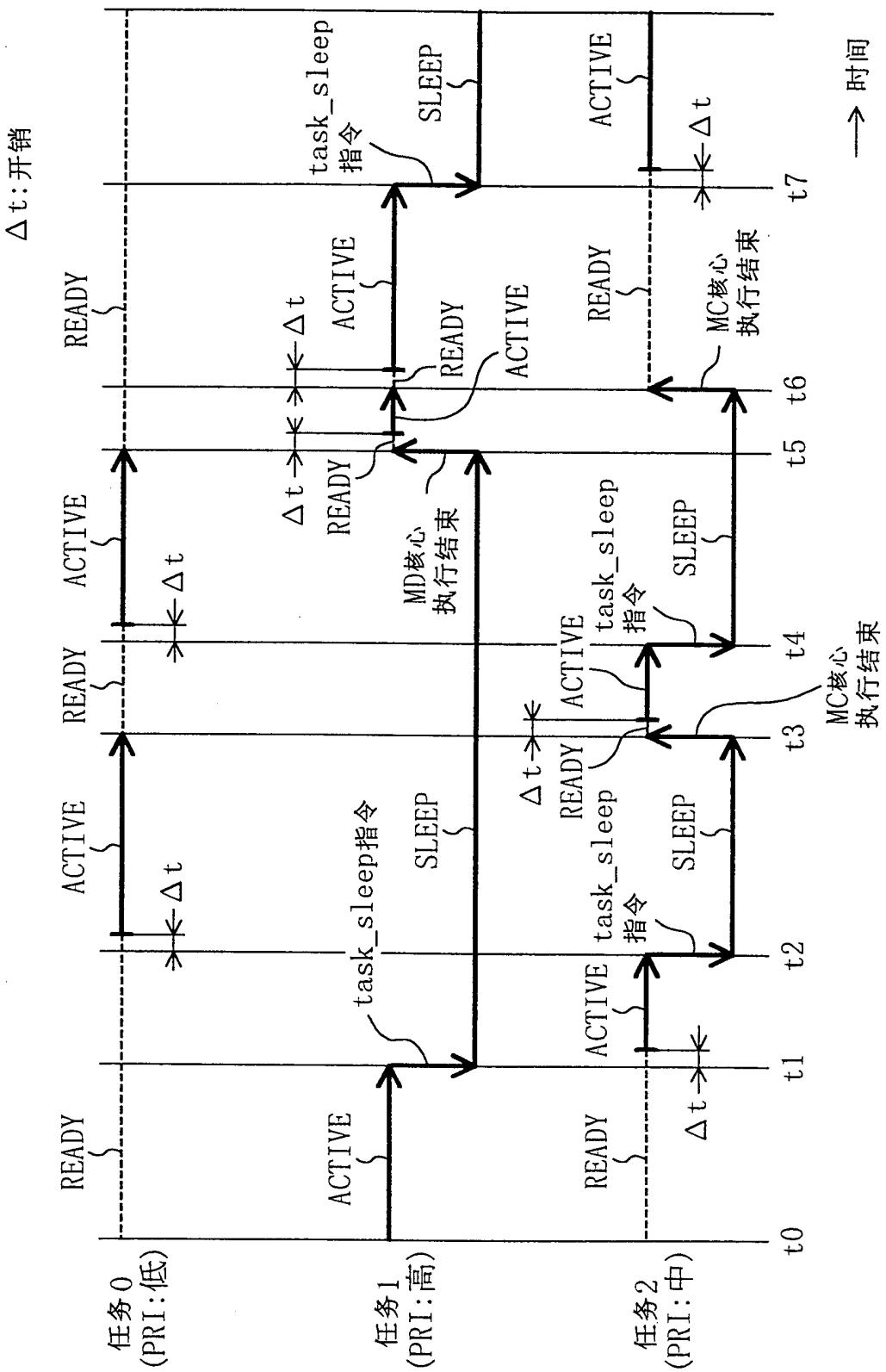

Δt:开销
任务0
(PRI:低)
任务1
(PRI:高)
任务2
(PRI:中)
READY
ACTIVE
SLEEP
Δt
task_sleep指令
MD核心
执行结束
MC核心
执行结束
t0
t1
t2
t3
t4
t5
t6
t7
时间

图 7