

[51] Int. Cl.
G06F 13/28 (2006.01)

1、一种处理器系统，包括：

能够并行进行算术处理的多个算术单元；

保存所述多个算术单元用于算术处理的数据的存储器；

与主处理器的处理并行地执行所述多个算术单元之间的数据传输，及所述多个算术单元和所述存储器之间的数据传输的多个 DMA 控制器；

与所述主处理器的处理并行地控制所述多个算术单元的启动和所述多个 DMA 控制器的启动的第一 DMA 控制电路；和

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

2、按照权利要求 1 所述的处理器系统，其中所述第一 DMA 控制电路控制所述多个 DMA 控制器的启动，以致所述多个 DMA 控制器并行进行数据传输。

3、按照权利要求 1 所述的处理器系统，其中所述第一 DMA 控制电路根据规定的事件，控制所述多个算术单元的启动和所述多个 DMA 控制器的启动，所述规定的事件包括从所述 DMA 控制器传送的数据传输完成的通知，从所述算术单元传送的算术处理完成的通知，和从所述主处理器传送的通知至少之一。

4、按照权利要求 3 所述的处理器系统，还包括保存与所述多个算术单元的操作状态相关的信息和与所述多个 DMA 控制器的操作状态相关的信息的状态存储器，

其中所述第一 DMA 控制电路根据保存在所述状态存储器中的内容，控制所述多个算术单元的启动和所述多个 DMA 控制器的启动。

5、按照权利要求 4 所述的处理器系统，还包括保存操作所述多个 DMA 控制器所需的信息的 DMA 信息存储器，

其中所述第一 DMA 控制电路根据保存在所述状态存储器中的内容，更新保存在所述 DMA 信息存储器中的内容。

6、按照权利要求 1 所述的处理器系统，其中所述第一 DMA 控制电路执行所述多个算术单元和所述多个 DMA 控制器的初始设置。

7、按照权利要求 1 所述的处理器系统，其中所述第一 DMA 控制电路包括根据固有算法执行算术处理的专用算术处理电路，和

所述控制处理器包括根据通用程序代码执行算术处理的通用处理器。

8、按照权利要求 7 所述的处理器系统，其中所述通用处理器根据其中描述所述多个算术单元和所述多个 DMA 控制器的启动序列的程序代码，命令所述专用算术处理电路。

9、按照权利要求 1 所述的处理器系统，还包括：

保存其中描述所述多个算术单元和所述多个 DMA 控制器的启动序列的程序代码的启动序列存储器；

其中所述第一 DMA 控制电路包括根据保存在所述启动序列存储器中的程序代码，控制所述多个算术单元的启动和所述多个 DMA 控制器的启动的启动控制器。

10、按照权利要求 1 所述的处理器系统，还包括：

一个主处理器；和

与所述主处理器连接，执行图形处理的图形处理处理器，

其中所述多个算术单元、所述存储器、所述多个 DMA 控制器和所述第一 DMA 控制电路和控制处理器都设置在所述图形处理处理器之内。

11、按照权利要求 10 所述的处理器系统，其中所述图形处理处理器包括：

把多边形的顶点信息转换成象素数据的象素转换器；和

所述象素转换器、所述第一 DMA 控制电路和控制处理器和所述存储器共同连接的总线，

其中所述多个算术单元通过利用所述象素转换器的输出和保存在所述存储器中的数据，进行算术运算。

12、按照权利要求 1 所述的处理器系统，其中所述存储器是独立

于主存储器单独设置的嵌入式 DRAM。

13、一种 DMA 控制电路，包括：

分别保存与从主处理器指示的多个指令相关的信息的多个指令存储器；

与所述主处理器的处理并行地执行均能够并行进行算术处理的多个算术单元之间的数据传输，及在保存所述多个算术单元用于算术处理的数据的存储器和所述多个算术单元之间的数据传输，并根据保存在所述多个指令存储器中的信息执行所述多个算术单元的启动控制的第一 DMA 控制电路；和

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

14、一种控制 DMA 控制器的方法，包括：

把能够并行进行算术处理的算术单元用于算术处理的数据保存到存储器中；

在所述多个算术单元之间，以及在所述多个算术单元和所述存储器之间，并行于主处理器的处理，利用多个 DMA 控制器进行数据传输；

通过第一 DMA 控制电路监视所述多个 DMA 控制器和所述多个算术处理单元，并向控制处理器传送表示数据传输完成的数据传输完成信息；

根据数据传输完成信息，通过第一 DMA 控制电路确定其它数据传输是否可能；和

当确定数据传输可能时，通过第一 DMA 控制电路命令所述多个 DMA 控制器中的至少一个传输数据。

15、一种图形处理方法，包括：

把顶点信息转换成象素信息；

根据象素信息，由多个算术单元产生图像；

把能够并行进行算术处理的所述算术单元用于算术处理的数据保存到存储器中；

在所述多个算术单元之间,以及在所述多个算术单元和所述存储器之间,利用多个 DMA 控制器进行数据传输;

通过第一 DMA 控制电路监视所述多个 DMA 控制器和所述多个算术单元;

根据表示数据传输完成的数据传输完成信息,确定其它数据传输是否可能;

当确定其它数据传输可能时,通过第一 DMA 控制电路命令所述多个 DMA 控制器中的至少一个传输数据; 和

通过控制处理器根据所述主处理器的指令控制所述第一 DMA 控制电路。

16、一种图形处理电路,包括:

把顶点信息转换成象素信息的象素信息转换器;

能够根据象素信息并行进行算术处理的多个算术单元;

与主处理器的处理并行地,在所述多个算术单元之间,以及在保存所述多个算术单元使用的数据的存储器和所述算术单元之间,进行数据传输的多个 DMA 控制器;

保存与 DMA 控制器进行的多个 DMA 传输相关的指令信息的多个指令信息存储器;

根据存储在多个指令信息存储器中的信息监视多个 DMA 控制器或多个算术单元,根据向控制处理器表示数据传输完成的数据传输完成信息确定其他数据传输是否可能,以及在确定其他数据传输可能时命令多个 DMA 控制器中的至少一个来传输数据的第一 DMA 控制电路; 和

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

17、按照权利要求 16 所述的图形处理电路,其中所述第一 DMA 控制电路通过监视所述多个 DMA 控制器,确定 DMA 传输是否完成。

18、按照权利要求 17 所述的图形处理电路,其中所述指令信息是表示所述多个 DMA 控制器和所述多个算术单元之间的 DMA 传输

是否完成的信息。

19、按照权利要求 16 所述的图形处理电路，其中所述第一 DMA 控制电路控制所述多个 DMA 控制器的启动，从而所述多个 DMA 控制器并行进行数据传输。

20、按照权利要求 16 所述的图形处理电路，其中所述第一 DMA 控制电路根据规定的事件，控制所述多个算术单元的启动和所述多个 DMA 控制器的启动，所述规定的事件包括从所述 DMA 控制器传送的数据传输完成的通知，从所述算术单元传送的算术处理完成的通知，和从所述主处理器传送的通知至少之一。

21、按照权利要求 16 所述的图形处理电路，其中所述第一 DMA 控制电路包括根据固有算法执行算术处理的专用算术处理电路，

所述控制处理器包括根据通用程序代码执行算术处理的通用处理器；和

其中所述专用算术处理电路根据所述通用处理器的指令，控制所述多个算术单元的启动和所述多个 DMA 控制器的启动。

22、按照权利要求 16 所述的图形处理电路，其中设置 30 个以上的所述 DMA 控制器。

23、按照权利要求 16 所述的图形处理电路，其中所述指令信息存储器是嵌入式 DRAM。

处理器系统，DMA 控制电路，DMA 控制方法， DMA 控制器用控制方法，图形处理方法和图形处理电路

技术领域

本发明涉及能够借助 DMA（直接存储器存取）实现数据传输过程的处理器系统，DMA 控制电路，DMA 控制方法，DMA 控制器用控制方法，图形处理方法和图形处理电路。

背景技术

在传统的 DMA 传输中，与主处理器的操作并行地进行数据传输，以便减小主处理器上的处理负载。在进行 DMA 传输的时候，主处理器能够进行另一过程。为此，DMA 传输适合于传送大量数据，例如图像数据的情况。

一般在 DMA 控制器的控制下进行 DMA 传输。主处理器事先在 DMA 控制器中设置代表进行 DMA 传输的时间，从其开始进行 DMA 传输的位置，和对其进行 DMA 传输的位置的传输控制信息。根据设置信息，DMA 控制器进行 DMA 传输。

另一方面，作为提高处理器系统中的处理速度的方法，已知的有相互并行地排列和操纵多个算术单元的方法。这种情况下，最好相互并行地传送多个算术单元的计算处理结果，并且可安排多个 DMA 控制器。

但是，当安排多个 DMA 控制器，使得能够相互并行地进行数据传输过程时，主处理器必须为 DMA 控制器设置多条传输控制信息，主处理器上的处理负载增大。处理负载与 DMA 控制器的数目成比例地增大。

发明内容

根据本发明一个实施例的处理器系统包括：

能够并行进行算术处理的多个算术单元；

保存所述多个算术单元用于算术处理的数据的存储器；

与主处理器的处理并行地执行所述多个算术单元之间的数据传输，及所述多个算术单元和所述存储器之间的数据传输的多个 DMA

控制器;

与所述主处理器的处理并行地控制所述多个算术单元的启动和所述多个 DMA 控制器的启动的第一 DMA 控制电路; 和

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

根据本发明一个实施例的 DMA 控制电路包括:

分别保存与从主处理器指示的多个指令相关的信息的多个指令存储器; 和

与所述主处理器的处理并行地执行均能够并行进行算术处理的多个算术单元之间的数据传输, 及在保存所述多个算术单元用于算术处理的数据的存储器和所述多个算术单元之间的数据传输, 并根据保存在所述多个指令存储器中的信息, 启动执行所述多个算术单元的启动控制的调度器第一 DMA 控制电路;

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

根据本发明一个实施例的控制 DMA 控制器的方法包括:

把能够并行进行算术处理的算术单元用于算术处理的数据保存到存储器中;

在所述多个算术单元之间, 以及在所述多个算术单元和所述存储器之间, 并行于主处理器的处理, 利用多个 DMA 控制器进行数据传输;

通过第一 DMA 控制电路监视所述多个 DMA 控制器和所述多个算术处理单元, 并向控制处理器传送表示数据传输完成的数据传输完成信息;

根据数据传输完成信息, 通过第一 DMA 控制电路确定其它数据传输是否可能; 和

当确定数据传输可能时, 通过第一 DMA 控制电路命令所述多个 DMA 控制器中的至少一个传输数据。

根据本发明一个实施例的图形处理方法包括:

把顶点信息转换成象素信息;

根据象素信息, 由多个算术单元产生图像;

把能够并行进行算术处理的所述算术单元用于算术处理的数据保存到存储器中;

在所述多个算术单元之间, 以及在所述多个算术单元和所述存储

器之间，利用多个 DMA 控制器进行数据传输；

通过第一 DMA 控制电路监视所述多个 DMA 控制器和所述多个算术单元；

根据表示数据传输完成的数据传输完成信息，确定其它数据传输是否可能；

当确定其它数据传输可能时，通过第一 DMA 控制电路命令所述多个 DMA 控制器中的至少一个传输数据；和

通过控制处理器根据所述主处理器的指令控制所述第一 DMA 控制电路。

根据本发明一个实施例的图形处理电路包括：

把顶点信息转换成象素信息的象素信息转换器；

能够根据象素信息并行进行算术处理的多个算术单元；

与主处理器的处理并行地，在所述多个算术单元之间，以及在保存所述多个算术单元使用的数据的存储器和所述算术单元之间，进行数据传输的多个 DMA 控制器；

保存与 DMA 控制器进行的多个 DMA 传输相关的指令信息的多个指令信息存储器；

根据存储在多个指令信息存储器中的信息监视多个 DMA 控制器或多个算术单元，根据向控制处理器表示数据传输完成的数据传输完成信息确定其他数据传输是否可能，以及在确定其他数据传输可能时命令多个 DMA 控制器中的至少一个来传输数据的第一 DMA 控制电路；和

根据所述主处理器的指令控制所述第一 DMA 控制电路的控制处理器。

附图说明

图 1 是表示根据本发明第一实施例的处理器系统的示意结构的方框图。

图 2 是表示图 1 中所示的图形处理处理器 2 的内部结构的细节的方框图。

图 3 是表示控制器 21 的内部结构的例子的方框图。

图 4 是表示图 3 中的专用电路 32 的内部结构的例子的方框图。

图 5 是表示图 1 的控制器 21 执行的程序的例子的流程图。

图 6A 和 6B 表示专用电路 32 执行的任务的例子。

图 7A 表示传统 DMAC 31 的第一操作,图 7B 表示传统 DMAC 31 的第二操作,图 7C 表示根据本实施例的 DMAC 的操作。

图 8 是对应于图 7A 的计时图。

图 9 是对应于图 7B 的计时图。

图 10 是对应于图 7C 的计时图。

图 11 表示如何使用同步寄存器的一个例子。

图 12 表示如何使用同步寄存器的另一例子。

图 13 表示如何使用同步寄存器的另一例子。

图 14 表示处理两个指令串的方法。

图 15 是表示控制器 21 的程序的例子的数据流程图。

图 16 表示主处理器 1 的程序的一个例子。

图 17 是表示根据第二实施例的控制器 21 的内部结构的方框图。

图 18 是表示图 17 中的专用电路 32a 的内部结构的例子的方框图。

图 19 是表示图 17 中的控制器 21 执行的程序的例子的流程图。

图 20 是表示根据本申请的处理器系统装入数字电视机中的情况的方框图。

图 21 是表示根据本实施例的处理器系统装入视频记录器/播放器中的例子的方框图。

具体实施方式

下面参考附图,说明根据本发明一个实施例的处理器系统,DMA 控制电路,DMA 控制方法,DMA 控制器用控制方法,图形处理方法和图形处理电路。

(第一实施例)

图 1 是表示根据本发明第一实施例的处理器系统的示意结构的方框图。图 1 中所示的处理器系统包括主处理器 1,图形处理处理器 2,主存储器 3 和 I/O 处理器 4。

主处理器 1 包括主处理器 11,多个数字信号处理器(DSP)和控制与外部电路的输入/输出操作的 I/O 单元 13、14 和 15。I/O 单元 13

控制与主存储器 3 的输入/输出操作, I/O 单元 14 控制与图形处理处理器 2 的输入/输出操作, I/O 单元 15 控制与 I/O 处理器 4 的输入/输出操作。

图形处理处理器 2 包括充当本实施例的特征部分的控制器 21, 执行与主处理器 1 的数据交换的 I/O 单元 22, 各种通用总线, 例如 PCI 总线, 控制视频数据、音频数据等的输入/输出操作的 I/O 单元 23, 和执行图形处理计算的图形处理单元 24。

图形处理单元 24 包括把多边形的顶点信息转换成象素数据的象素转换器 26, 和处理象素数据的多个算术单元 (DSP) 27。

I/O 处理器 4 控制与通用总线, 诸如 HDD 和 DVD 之类的外围设备, 以及网络的连接。

图 2 是表示图 1 中所示的图形处理处理器 2 的内部结构的细节的方框图。多个算术单元 27 中的每一个都包括由多个处理器构成的一个处理器群集 28, 和保存处理器群集 28 的处理结构的一个存储器 29。处理器群集 28 中的多个处理器能够彼此并行地执行独立过程, 并且能够执行一个过程, 以致多个处理器群集 28 共享该过程。存储器 29 保存处理器群集 28 的执行结果。图 2 中所示的控制器 21, 象素转换器 26, I/O 单元 22 及 23, 和存储器 29 与公共总线 30 连接。

图 3 是表示控制器 21 的内部结构的例子的方框图。图 3 中所示的控制器 21 包括多个 DMA 控制器 (DMAC) 31, 一个专用电路 32, 由多个通用处理器构成的一个控制处理器 33, 一个计时器 34, 一个中断单元 35 和一个存储器 36。

DMA 控制器 31 执行多个算术单元 27 之间的数据传输, 以及多个算术单元 27 和存储器 36 之间的数据传输。专用电路 32 是专用于该系统的电路, 执行关于 DMA 控制器 31 和算术单元 27 的启动控制。控制处理器 33 根据保存在存储器 36 中的程序代码, 或者来自主处理器 1 的指令, 控制专用电路 32。计时器 34 执行时间管理, 并根据需要指令中断单元 35 执行中断。中断单元 35 接收来自计时器 34 的信号, 或者来自 DMA 控制器 31 或算术单元 27 的完成信号, 对控制处理器

33 执行中断。

图 3 中所示的专用电路 32, 计时器 34, 中断单元 35 和存储器 36 与本地网络 37 连接。主处理器 1 也通过 I/O 单元 22 与本地网络 37 连接。存储器 36 由例如嵌入式 DRAM 构成。提供例如 30 个以上的 DMA 控制器。

图 4 是表示图 3 中的专用电路 32 的内部结构的例子的方框图。该方框图表示其中以控制处理器 33 的协处理器的形式, 连接专用电路 32 的结构。图 4 中的专用电路 32 包括控制处理器 I/O 单元 41, 多个 DMA 寄存器 42, 一个 DMA 发布单元 43, 多个同步寄存器 44, 和一个同步寄存器控制单元 45。

控制处理器 I/O 单元 41 与控制处理器 33 交换数据。DMA 寄存器保存 DMAC 31 的操作所需的各种信息。DMA 发布单元 43 执行把 DMA 寄存器 42 中的各种信息传送给 DMAC31 的过程。从其传送信息的特定 DMA 寄存器和向其传送信息的特定 DMAC 31 由控制处理器 33 确定。通过控制处理器 I/O 单元 41, 把所述特定 DMA 寄存器和特定 DMAC 31 通知 DMA 发布单元 43。同步寄存器 44 保存 DMAC 31 和算术单元 27 的操作状态。同步寄存器控制单元 45 控制同步寄存器 44 的更新。

图 5 是表示图 1 的控制器 21 执行的程序的例子的流程图。该流程图表示当在主处理器 1 中准备的数据被 DMA 传送给图形处理处理器 2 中的存储器 36 时执行的程序。

当主处理器 1 准备将由图形处理处理器 2 处理的数据时, 主处理器 1 把准备完成信号传送给图 3 中的本地网络 37。该信号由图 4 中的同步寄存器控制单元 45 接收 (步骤 S1)。

同步寄存器控制单元 45 更新同步寄存器 44 的值 (步骤 S2)。更具体地说, 同步寄存器 44 被设置成代表主处理器 1 的数据传输的准备已完成的值。

控制处理器 33 根据专用指令, 通过控制处理器 I/O 单元 41 读取同步寄存器 44 的值, 以确认主处理器 1 的准备已完成 (步骤 S3)。

这里，所述专用指令表示一项任务。包括由控制处理器 33 和专用电路 32 执行的专用指令（任务）的指令串表示一个任务串。

通过控制处理器 I/O 单元 41，控制处理器 33 把 DMAC 31 的设置信息传送给 DMA 寄存器 42（步骤 S4）。

控制处理器 33 根据专用指令，通过控制处理器 I/O 单元 41 启动 DMA 发布单元 43，并指令 DMA 发布单元 43 把保存在 DMA 寄存器 42 中的 DMAC 31 的设置信息设置到每个 DMAC 31 中（步骤 S5）。

之后，DMAC 31 执行 DMA 传输。当 DMA 传输完成时，DMAC 31 向同步寄存器控制单元 45 通知 DMA 传输已完成（步骤 S6）。同步寄存器控制单元 45 更新同步寄存器 44（步骤 S7）。

控制处理器 33 根据专用指令，通过控制处理器 I/O 单元 41，读取同步寄存器的值，并确认 DMA 传输完成（步骤 S8）。之后，控制处理器 33 通过控制处理器 I/O 单元 41 启动算术单元 27，处理从主处理器 1 传送给图形处理处理器 2 中的存储器的数据（步骤 S9）。

如上所述，图 4 中的同步寄存器控制单元 45 监视同步寄存器 44 的值。定期监视通常被称为“轮询”。根据本实施例的同步寄存器控制单元 45 能够通过轮询，监视 DMAC 31，算术单元 27 和主处理器 1 的操作。

代替用同步寄存器控制单元 45 监视同步寄存器 44，控制处理器 33 可通过控制处理器 I/O 单元 41 监视同步寄存器 44。

这样，控制 DMAC 31 的命令被称为指令，每个命令由例如 256 位数据构成。专用电路 32 中的同步寄存器控制单元 45 和同步寄存器 44 用作任务调度器。

更具体地说，任务包括与 DMAC 31 的数据传输控制相关的命令，与算术单元 27 的启动控制以及算术单元 27 的初始设置相关的命令，和与关于主处理器 1 的中断通知相关的命令。

根据本实施例的专用电路 32 持续执行任务，直到称为阻挡任务（block task）的特殊任务被执行为止。阻挡任务是等待在所述阻挡任务之前发出的任务（DMA 执行或者处理器群集的过程）的执行完成

的任务。当阻挡任务被执行时，专用电路 32 等待，直到完成设定的任务的执行为止。

图 6A 表示专用电路 32 执行的任务的例子，表示了其中执行任务 A、B、C、D、E、F 和 G 的例子。图 6A 中的任务被写成如图 6B 中所示的数据流程图。专用电路 32 执行任务 A，等待作为阻挡任务的 A 的执行完成。当完成任务 A 的执行时，专用电路 32 并行执行任务 F 和 B。专用电路 32 等待作为阻挡任务的 F 和 B 的执行完成。当完成任务 F 和 B 的执行时，专用电路 32 并行执行任务 C 和 D。专用电路 32 等待作为阻挡任务的 C 和 D 的执行完成。当完成任务 C 和 D 的执行时，专用电路 32 执行任务 E。专用电路 32 等待作为阻挡任务的 E 的执行完成。当完成任务 E 的执行时，专用电路 32 执行任务 G。

按照这种方式，图 6B 中的阻挡任务能够使多个 DMA 传输同步。

本实施例的专用电路 32 能够借助除了 DMA 传输完成的通知之外的事件启动接下来的 DMA 传输。这里提及的事件是例如算术单元 27 的计算过程的完成，或者来自主处理器 1 的通知。

图 7A 表示传统 DMAC 31 的第一操作，图 7B 表示传统 DMAC 31 的第二操作，图 7C 表示根据本实施例的 DMAC 的操作。图 8 是对应于图 7A 的计时图，图 9 是对应于图 7B 的计时图，图 10 是对应于图 7C 的计时图。

图 7A 和 8 表示最普及的 DMA 传输。在完成某一 DMA 传输之后，执行下一 DMA 传输。这种情况下，如图 8 中所示，主处理器 1（CPU）执行 DMA 的指定（t1），算术单元 27 的寄存器设置（t2），和算术单元 27 的执行的指定（t3）。DMAC 31 执行由主处理器 1 指定的 DMA 命令。

在图 7A 和 8 中所示的例子中，与 DMA 传输相关的各种设置和指定，以及算术单元 27 的执行的指定由主处理器 1 完成。因此，主处理器 1 上的负载过大。于是，主处理器 1 执行另一过程的时段（t4）变短，从而主处理器 1 的性能降低。

在图 7B 和 9 中所示的例子中，能够相互并行地执行多个 DMA 传输。但是，如图 9 中所示，主处理器 1 执行寄存器设置和算术单元 27 的起动的指定（时段 t5）。根据所述指定，DMAC 31 传输寄存器设置和算术单元 27 的起动的指定（时段 t6）。该例子中，由于能够并行地对多个 DMAC 31 进行寄存器设置，因此当要进行寄存器设置的寄存器的数目较大时，能够减小主处理器 1 上的处理负载。但是，当只重新设置一些寄存器时，和图 7A 和 8 中的情况一样，主处理器 1 上的处理负载增大。

另一方面，在图 7C 和 10 中所示的实施例中，不仅只通过利用 DMA 传输的结束作为触发信号，进行下一 DMA 传输，而且还利用来自算术单元 27 或主处理器 1 的通知作为触发信号，执行下一 DMA 传输。DMAC 31 能够执行算术单元 27 的寄存器设置，和算术单元 27 的执行的指定。更具体地说，如图 10 中所示，当主处理器 1 指定 DMAC 31 执行 DMA 传输（时段 t7）时，响应该指定，DMAC 31 执行算术单元 27 的寄存器设置（时段 t8），DMA 传输（时段 t9），和算术单元 27 的执行的指定（时段 t10）。当完成计算过程时，算术单元 27 把计算过程的结束通知 DMAC 31。

如图 10 中所示，由于 DMAC 31 控制 DMA 传输和控制算术单元 27 的执行，主处理器 1 能够向其它过程分配较长的时间。于是，能够提高主处理器 1 的性能。

下面更详细地说明主处理器 1 的操作，以及图形处理处理器 2 中的控制器 21 的操作。主处理器 1 读取保存在主存储器 3 中的任务串，以便把任务串传送给图形处理处理器 2 中的存储器。该传输过程可由主处理器 1 的存储任务直接写入该存储器中，或者 DMA 传输可实现执行成任务之一。

控制器 21 中的同步寄存器控制单元 45 把任务串的指针信息等设置到 DMAC 31 的 DMA 寄存器 42 中。根据 DMA 寄存器 42 的内容，DMA 发布装置对每个 DMAC 31 执行各种设置。

控制器 21 不仅能够执行 DMAC 31 的启动控制，而且能够执行

算术单元 27 的启动控制。作为当控制器 21 控制算术单元 27 时使用的任务，已知的有两种任务，即设置任务和突跳任务（kick task）。设置任务是对算术单元 27 执行各种设置的任务。更具体地说，进行各种设置，以便显示诸如纹理或顶点之类的三维图像。突跳任务是指定算术单元 27 的开始执行的任务。

如上所述，在同步寄存器 44 中，保存 DMAC 31，算术单元 27 等的操作状态。主处理器 1 能够通过同步寄存器控制单元 45，读取同步寄存器 44 的值。几种方法可被用作使用同步寄存器 44 的方法。图 11、12 和 13 中表示了这些方法中的一个典型方法。

在图 11 中，某一过程由控制器 21 执行（步骤 S11）。处理结果由写任务写入同步寄存器 44（步骤 S12）。控制器 21 中断该任务的执行，直到从主处理器 1 收到阻挡取消通知为止（步骤 S13）。当主处理器 1 定期进行同步寄存器 44 的轮询（步骤 S13），以获得在写任务中，写入同步寄存器 44 中的值时，主处理器 1 把阻挡取消通知控制器 21（步骤 S15）。

在图 12 中，控制器 21 借助突跳任务启动算术单元 27（步骤 S16），并中断该任务的执行，直到算术单元 27 中的过程结束为止（步骤 S17）。由突跳任务启动的算术单元 27 执行某一过程（步骤 S18）。当完成该过程时，算术单元 27 把完成通知传送给控制器 21，并把返回值写入同步寄存器 44（步骤 S19）。接收完成通知的控制器 21 参考通用寄存器的值完成转移（branch）（步骤 S20）。

在图 13 中，控制器 21 执行某一过程（步骤 S21）。当完成该过程时，中断该任务的执行，直到控制器 21 从主处理器 1 收到阻挡取消的通知为止（步骤 S22）。主处理器 1 动态设置控制器 21 重新开始执行该任务的时间（步骤 S23）。此时，主处理器 1 取消控制器 21 的阻挡，并把返回值写入同步寄存器 44 中（步骤 ST24）。控制器 21 参考同步寄存器 44 的值完成转移（步骤 ST25）。

如上所述，控制器 21 能够同时执行多个任务串。作为所述执行的一个例子，下面说明一个程序，执行所述程序，以致数据从主存储

器 3 传送到存储器中，数据的指针被设置到算术单元 27 中。这种情况下，控制器 21 同时执行两个任务串，并使任务串同步。所述同步由来自主处理器 1 的指定实现。

如图 14 中所示，假定存储器被分成四个区域（下面称为 FIFO 0-3）。在这两个任务串中的一个任务串（下面称为任务串 1）中，主存储器 3 把数据传送给 FIFO 0-3。在另一任务串（下面称为任务串 2）中，数据从 FIFO 0-3 传送给算术单元 27。

任务串 2 实际上借助设置任务，执行对算术单元 27 的初始设置。算术单元 27 从存储器读取数据。

图 15 是表示执行上述两个任务串 1 和 2 的控制器 21 的程序的例子的数据流程图。彼此并行地执行图 15 中的任务串 1 和任务串 2。在任务串 1 中，数据被顺序从主存储器 3 传送给 FIFO 0-3。之后，控制器 21 返回任务串 1 的顶部（步骤 S31-S39）。每次完成向 FIFO 0-3 的数据传输过程时，该过程被中断。当控制器 21 从主处理器 1 收到阻挡取消的通知时，控制器 21 执行下一数据传输过程。

另一方面，在任务串 2 中，FIFO 0-3 的地址被设置到算术单元 27 中，顺序重复指定算术单元 27 启动的过程，控制器 21 返回任务串 2 的顶部（步骤 S41-S54）。在 FIFO 0-3 的地址被设置到算术单元 27 中之后，中断该过程。当主处理器 1 取消该阻挡时，执行下一过程。

实现图 15 中的过程的主处理器 1 的程序的一个例子如图 16 中所示。

按照这种方式，在第一实施例中，独立于主处理器 1 安排具有实现多个 DMAC 31 和多个算术单元 27 的启动控制的控制器 21 的图形处理处理器 2，从而能够并行于在主处理器 1 中执行的过程，由控制器 21 完成 DMAC 31 的控制和算术单元 27 的启动指定。因此，能够减小主处理器上的处理负载。

DMAC 31 和算术单元 27 的启动指定也可由除 DMA 传输结束的通知之外的事件实现。因此，能够执行具有更高自由度的任务过程。

（第二实施例）

在上述第一实施例中，说明了其中 DMAC 31 和算术单元 27 的启动由控制器处理器 33 和专用电路 32 控制的例子。但是，DMAC 31 和算术单元 27 的启动控制可以只由专用电路 32 实现。

图 17 是表示根据第二实施例的控制器 21 的内部结构的方框图。和图 3 相同的附图标记表示图 17 中的相同部分。下面主要说明图 3 和图 17 的不同之处。在图 17 的控制器 21 中，图 3 中的控制处理器 33 和专用电路 32 被集成为一个专用电路 32a。

图 17 中的专用电路 32a 根据保存在存储器 36 中的程序代码，控制 DMAC 31 和算术单元 27。

图 18 是表示图 17 中的专用电路 32a 的内部结构的例子的方框图。图 18 中的专用电路 32a 包括一个任务取出解码器 51，一个同步管理单元 52，一个 DMA 发布单元 43，多个同步寄存器 44，和一个同步寄存器控制单元 45。

任务取出解码器 51 解释保存在存储器 36 中的程序代码。同步管理单元 52 执行由任务取出解码器 51 解释的任务，并读取同步寄存器 44 的值，以便控制算术单元 27 和 DMA 发布单元 43。

图 19 是表示图 17 中的控制器 21 执行的程序的例子的流程图。下面主要说明图 19 中的流程图和图 5 中的流程图之间的不同过程。在同步寄存器控制单元 45 更新同步寄存器 44 的值（步骤 S62）之后，同步管理单元 52 根据任务取出解码器 51 译解的任务，读取同步寄存器 44 的值（步骤 S63）。按照这种方式，确定完成主处理器 1 的准备中，关于数据传输的准备。

同步管理单元 52 把各条信息设置到 DMAC 31 中，以便把将设置在 DMAC 31 中的各条信息（这种情况下，在主处理器 1 中准备的数据）传送给图形处理处理器 2 中的存储器（步骤 S64）。

在 DMAC 31 被启动之后，当 DMAC 31 结束 DMA 传输时，完成信号被传送给同步寄存器控制单元 45（步骤 S65）。同步寄存器控制单元 45 更新同步寄存器 44（步骤 S66）。

根据任务取出解码器 51 译解的任务，同步管理单元 52 读取同步

寄存器 44 的值（步骤 S67），以确认 DMAC 31 的完成。

根据任务取出解码器 51 译解的任务，同步管理单元 52 启动算术单元 27，开始处理从主处理器 1 传送给图形处理处理器 2 中的存储器的数据（步骤 S68）。

如上所述，在第二实施例中，DMAC 31 和算术单元 27 的启动控制只由专用电路 32a 实现。因此，能够获得比利用通用处理器获得的电路规模更小的电路规模，并且还能够获得低能耗。

上面的每个实施例举例说明了控制器 21 被安置在图形处理处理器 2 中的情况。但是，控制器 21 可在安置在图形处理处理器 2 之外。

上面的每个实施例举例说明了以不同的芯片的形式，形成主处理器 1 和图形处理处理器 2 的情况。但是，主处理器 1 和图形处理处理器 2 也可作为宏核心（macro core）形成于同一芯片上。这种情况下，控制器 21 最好被安排在图形处理处理器 2 中。但是，控制器 21 也可安排在图形处理处理器 2 之外。

上面的每个实施例举例说明了控制器 21 专用于由图形处理处理器 2 执行的数据处理的情况。但是，控制器 21 也可控制另一 DMAC，即，主处理器 1 中的 DMAC。这种情况下，可与控制器 21 无关地在主处理器 1 中安排另一控制器。主处理器 1 中的 DMAC 和图形处理处理器 2 中的 DMAC 也可由共同的控制器控制。

如上所述的相同功能也可由代替控制器 21 的 OS（操作系统）处理。

根据本发明的处理器系统可被装入游戏机，家用服务器，电视机，便携式信息设备等中。

图 20 是表示根据本申请的处理器系统装入数字电视机中的情况的方框图。充当该结构的一个例子的数字电视机包括控制通信信息的数字电路板 55。数字电路板 55 包括实现和图 1 中一样的功能，控制图像信息的处理器系统 56。更具体地说，处理器系统 56 包括发射/接收视频和通信信息的收发器电路（DEMUX）57，解码器电路 58，处理器（CPU）59，图形处理电路（图形引擎）60，和数字格式转换器

61。

图 21 是表示根据本实施例的处理器系统装入视频记录器/播放器中的例子的方框图。作为该结构的一个例子，视频记录器/播放器包括实现和图 1 中一样的功能，并且控制图像信息的图像信息控制电路 62。更具体地说，图像信息控制电路 62 包括处理器（CPU）63，数字信号处理器（DSP）64，处理视频（图像）数据的处理器 65，和处理音频数据的处理器 66。

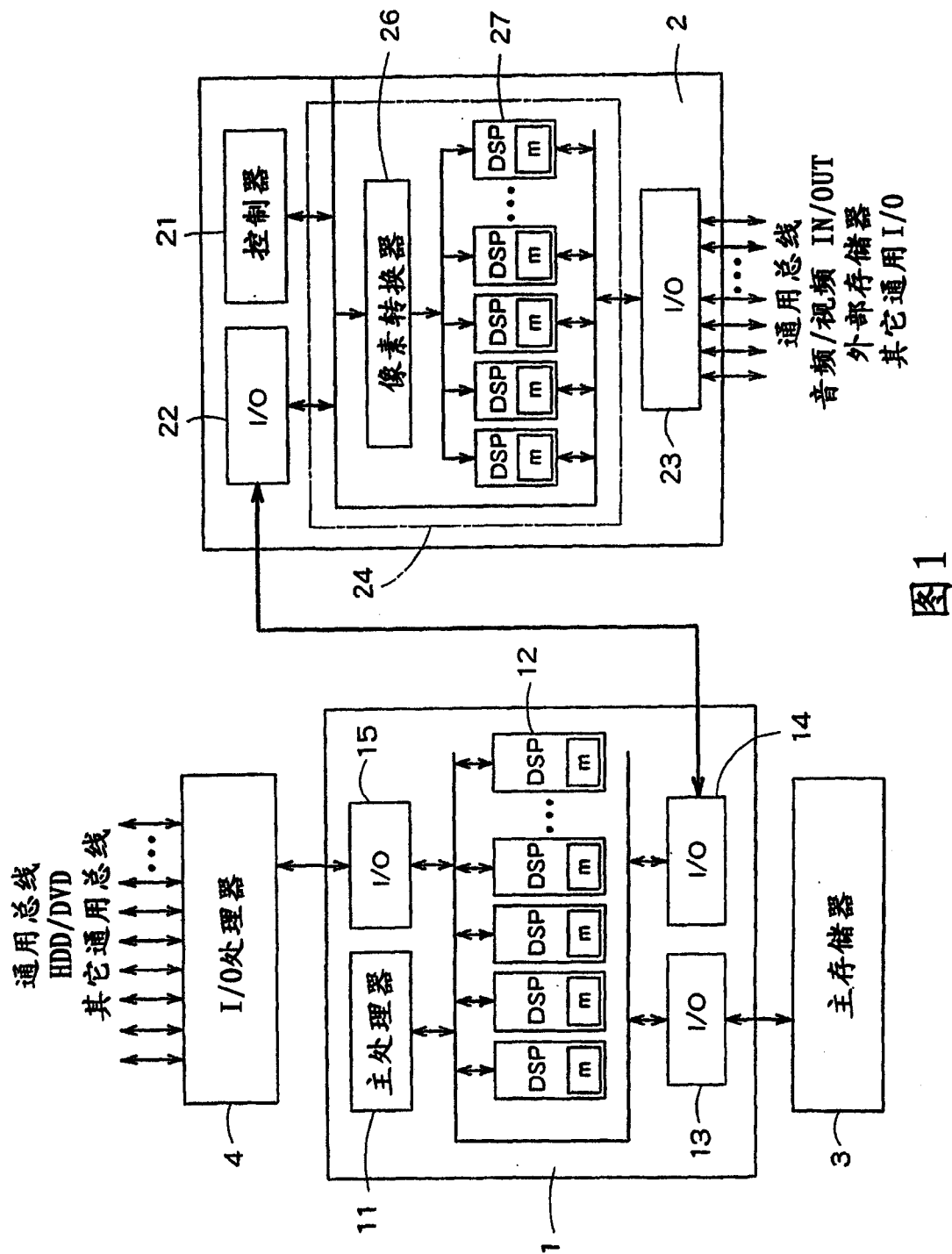


图1

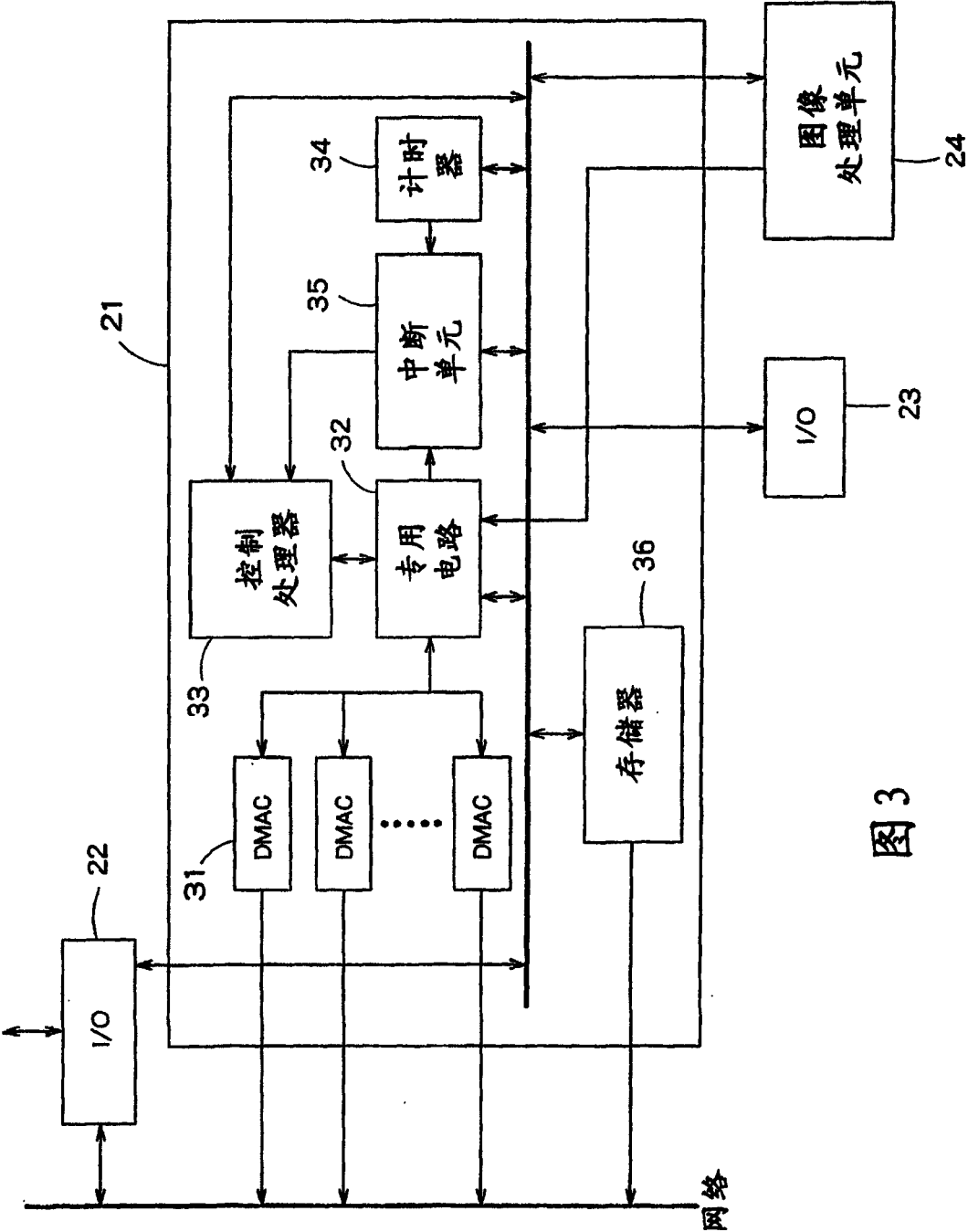


图 3

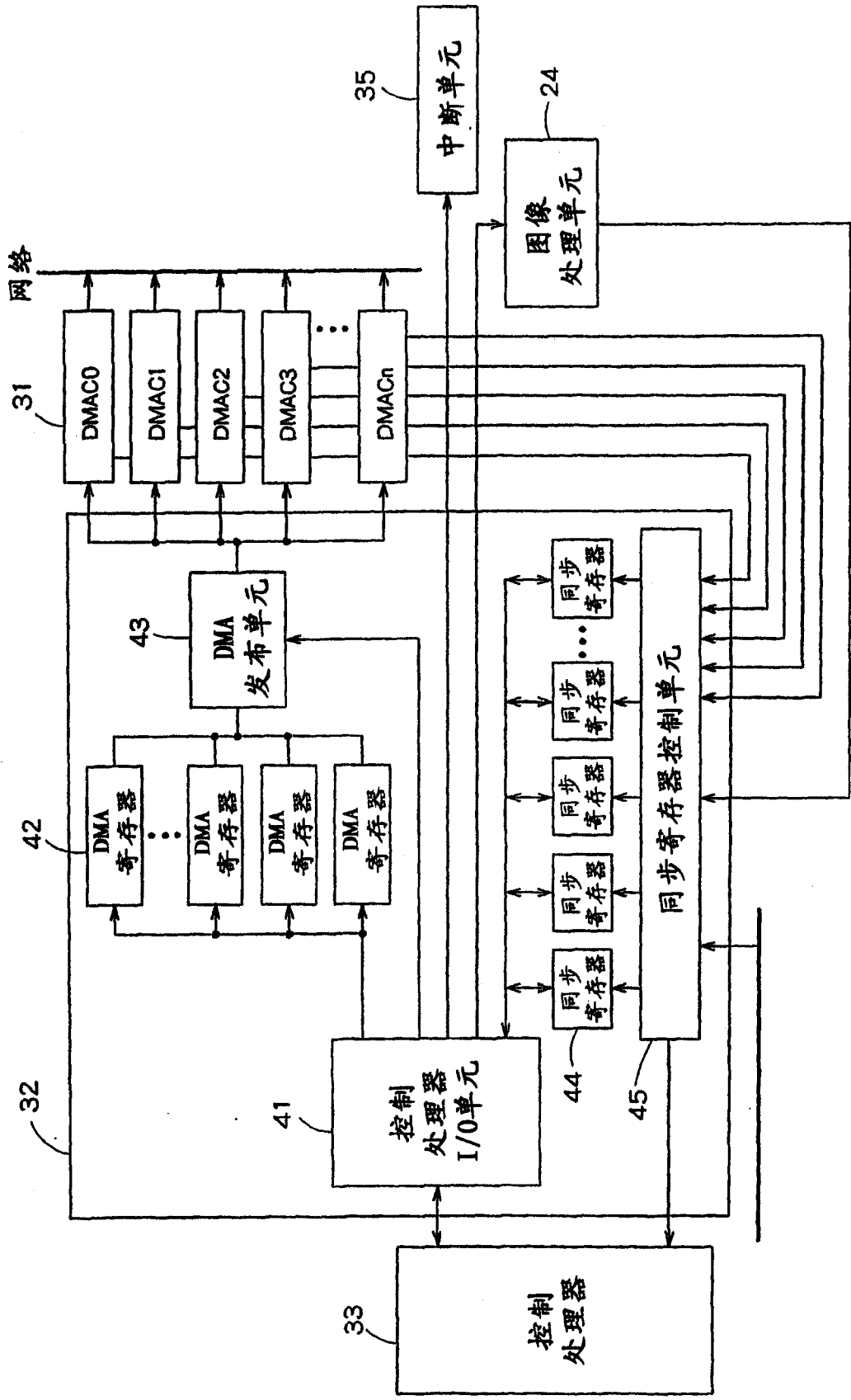


图 4

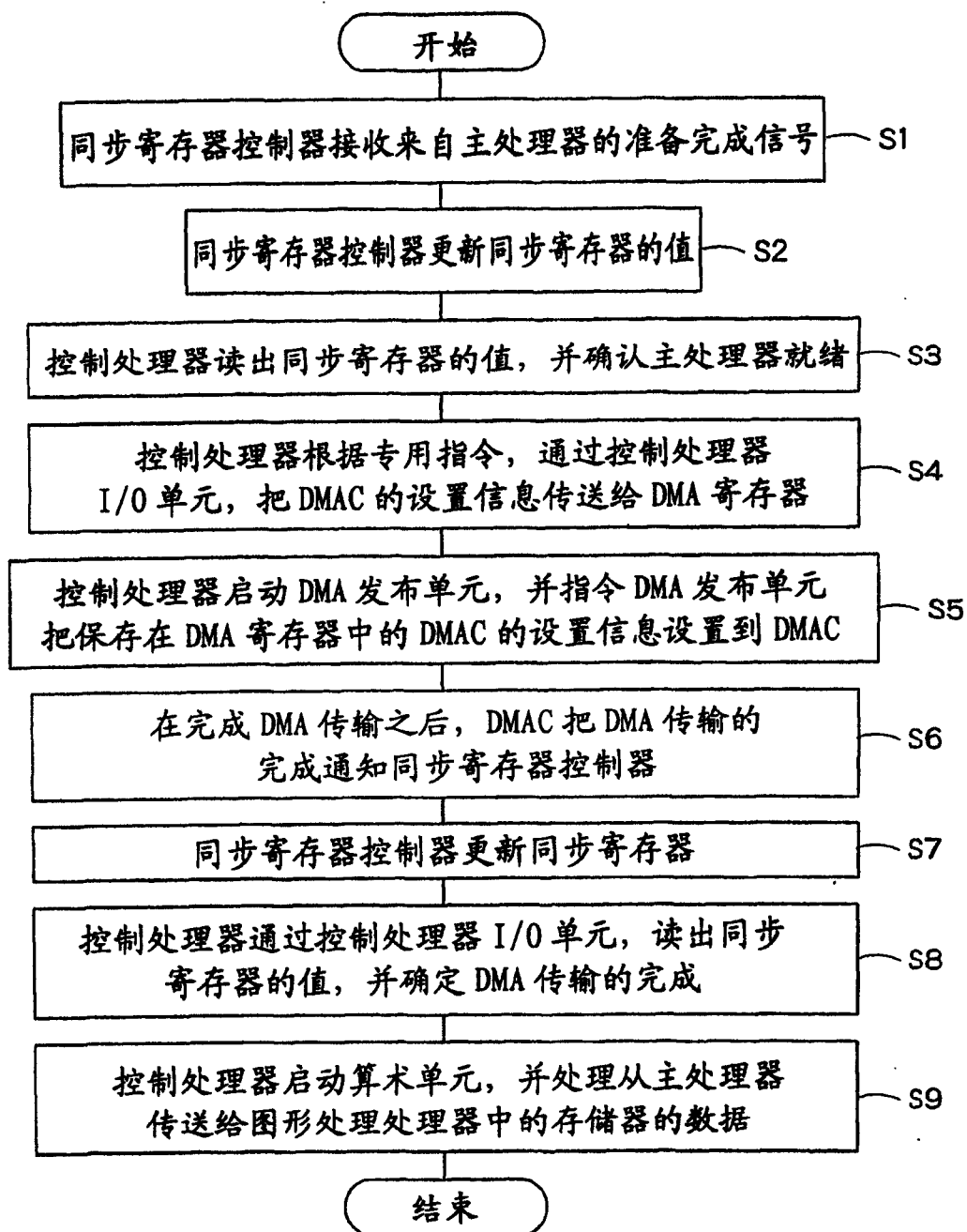


图 5

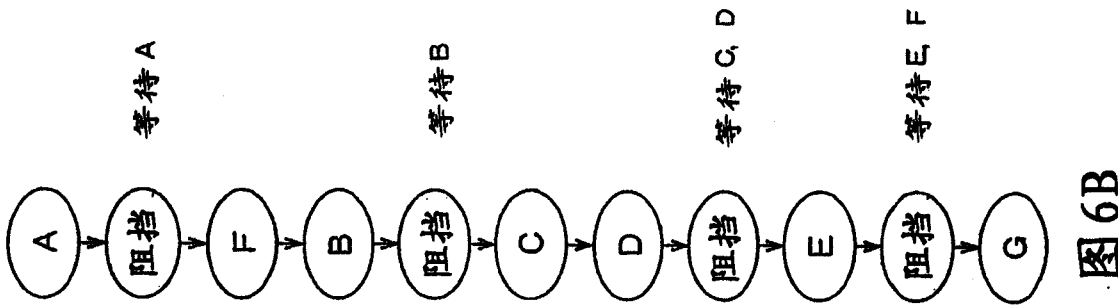


图 6B

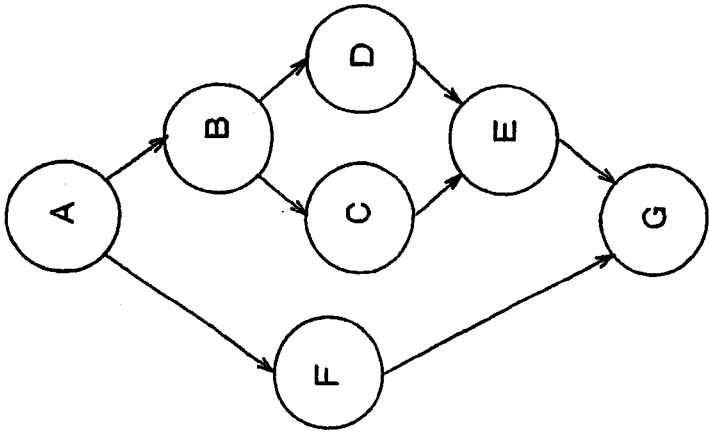
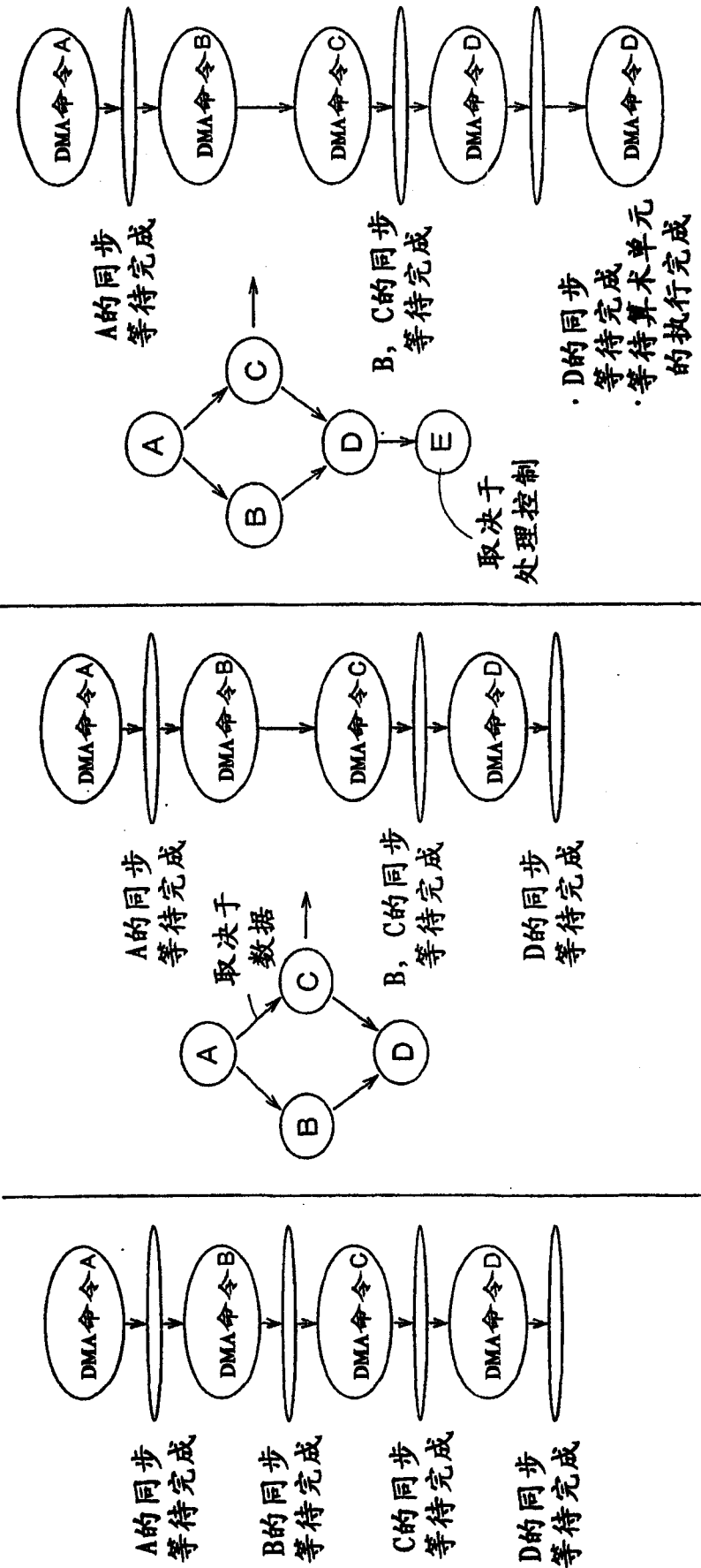


图 6A



常规结构1

常规结构2

提出的结构

图7A

图7B

图7C

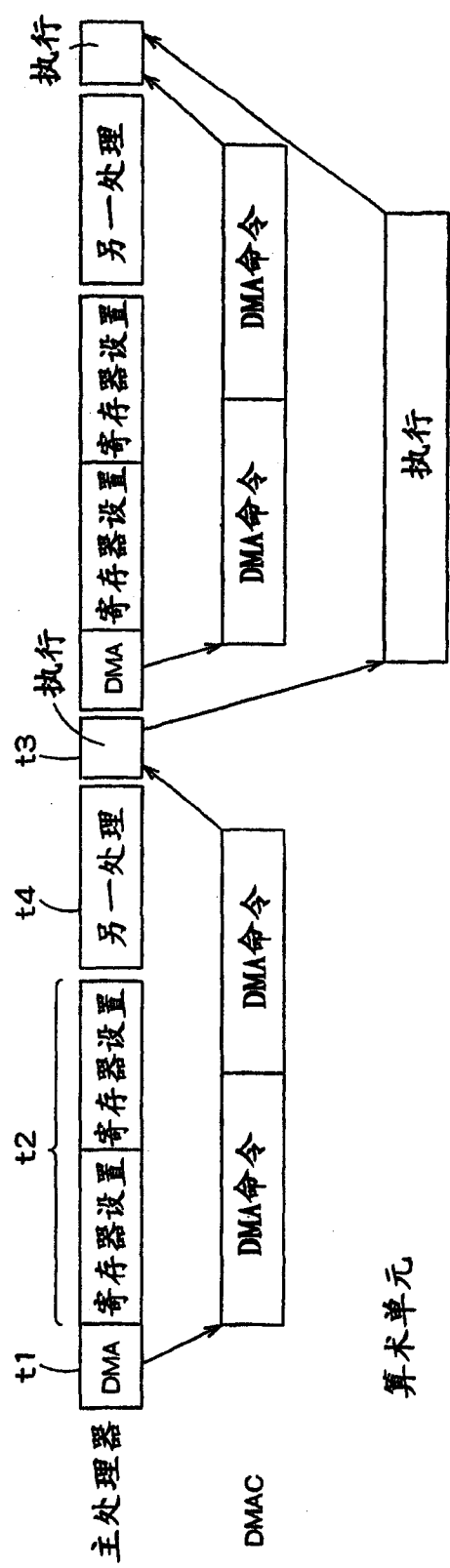


图8

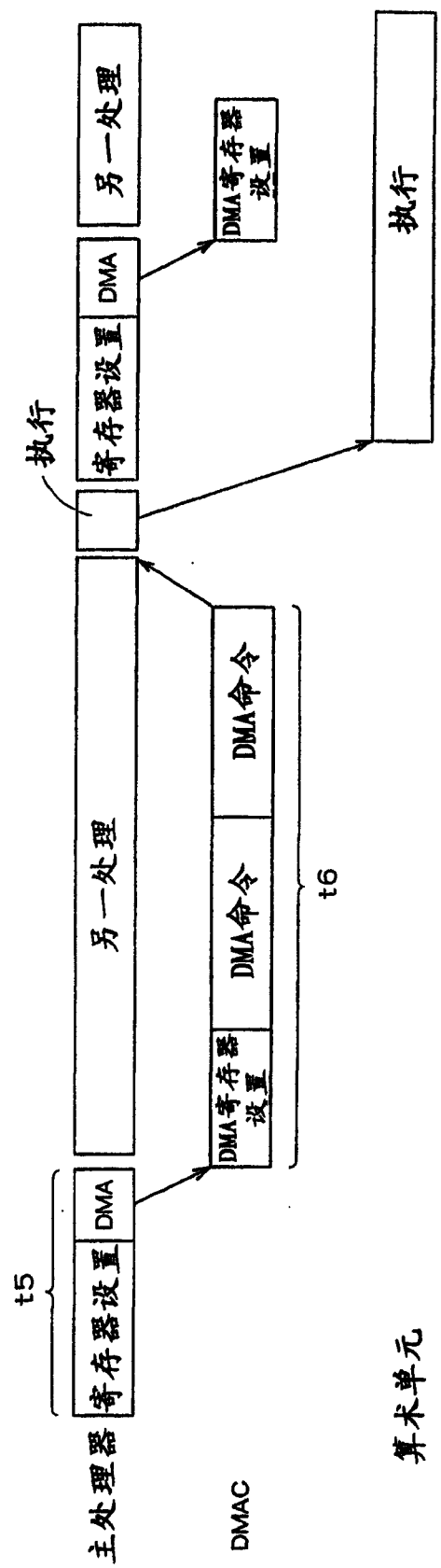
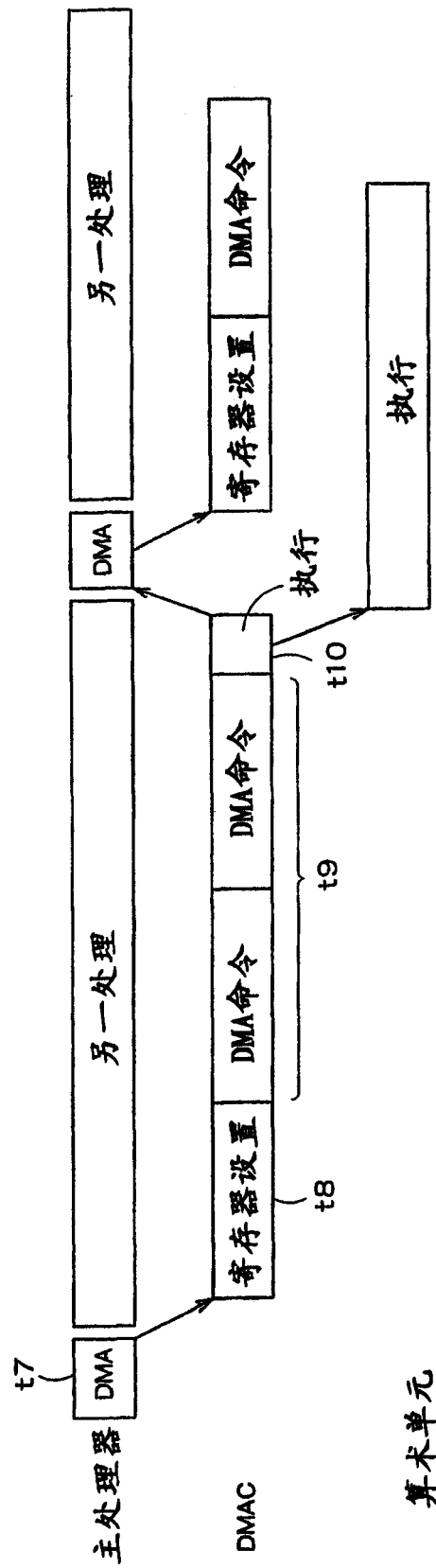


图9



10

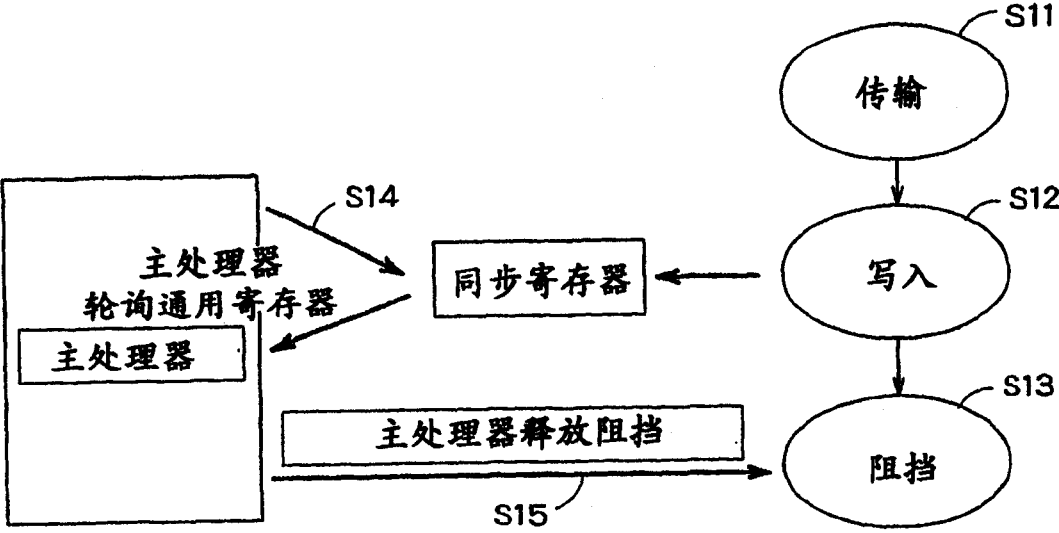


图 11

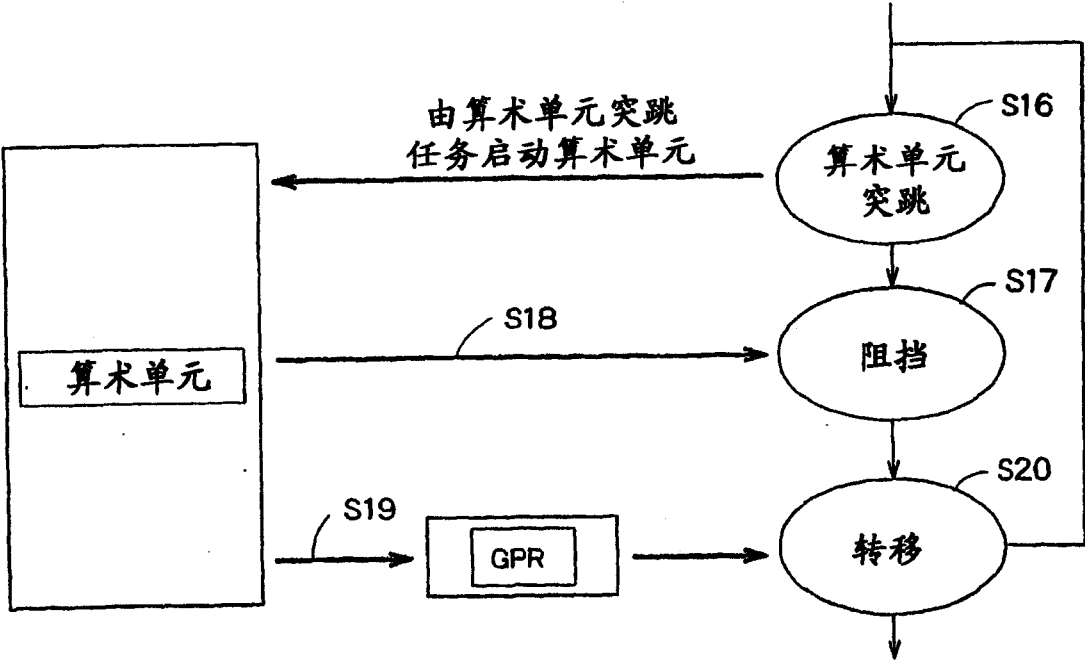


图 12

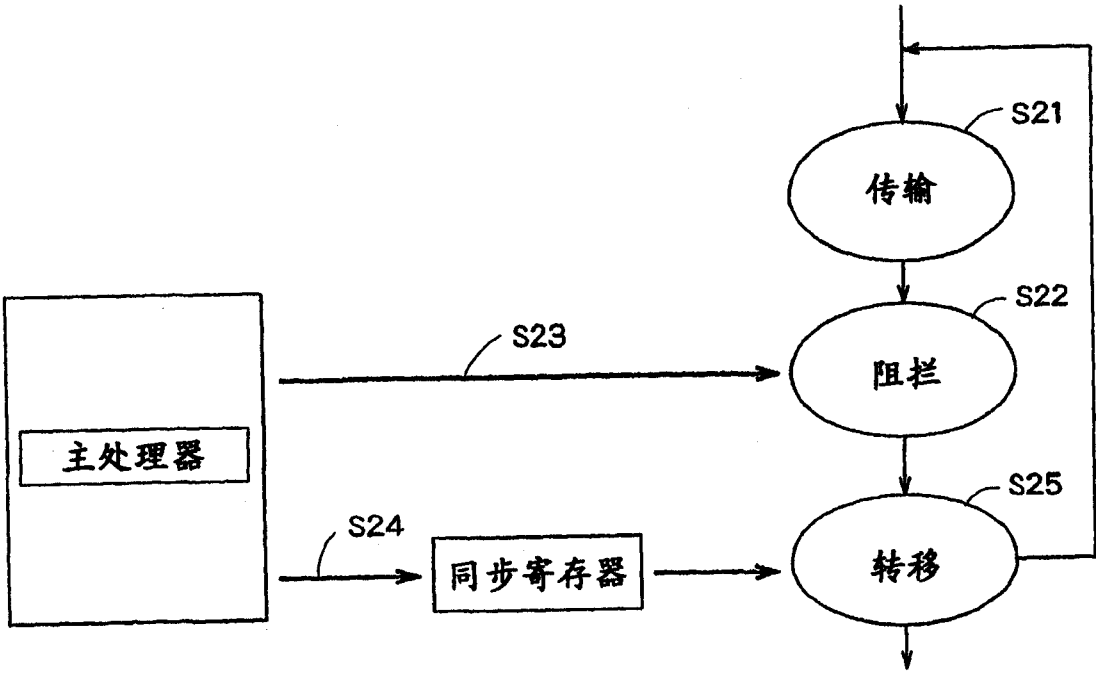


图13

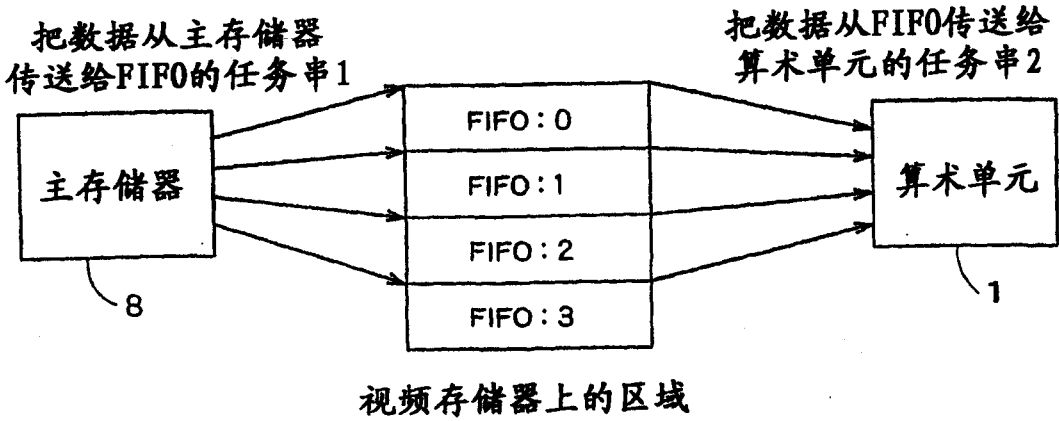


图14

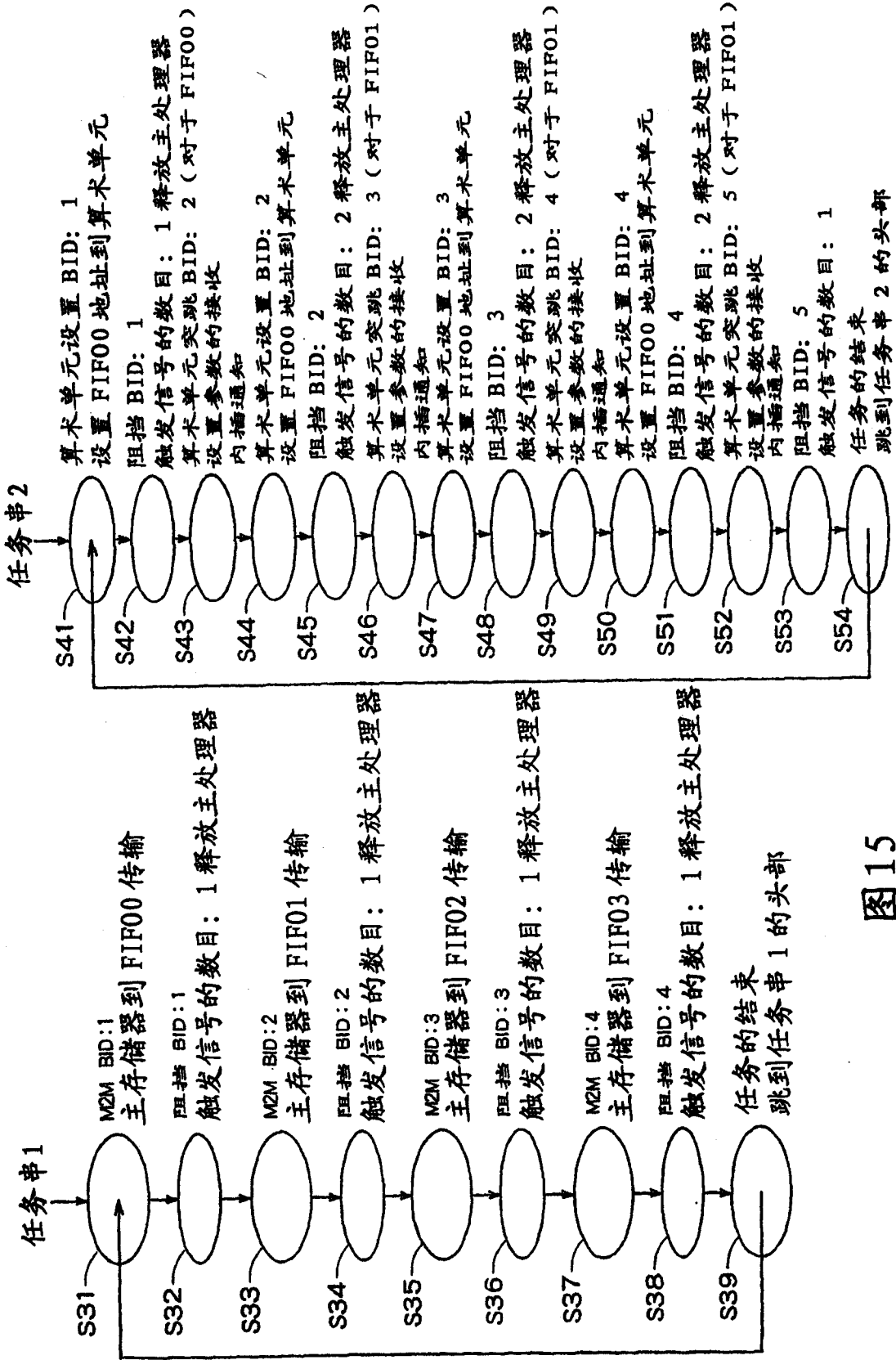


图15

```

/*****INITIALIZATION *****/
int depth=0;
//TRANSFER TaskString FROM MainMem TO DMA MEMORY
ts_transfer (TaskString1_MainMem_addr, TaskString1_LDPCUMEM_addr, TaskString1_size);
ts_transfer (TaskString2_MainMem_addr, TaskString2_LDPCUMEM_addr, TaskString2_size);
//REGISTER Task String TO TSS
ts_init (TSS0, TaskString1_LDPCUMEM_addr, contextID, TaskString1_tag);
ts_init (TSS1, TaskString2_LDPCUMEM_addr, contextID, TaskString2_tag);
START-UP TSS
ts_ctrl (TSS0, EXECUTE);
ts_ctrl (TSS1, EXECUTE);

/***** MAIN ROUTINE *****/
while (1) {
    if //IF TaskString1 WAITS RELEASE BY HOST PROCESSOR, AND FIFO IS NOT FULL
        if ((depth<3) && (ts_status(TSS0)==TSS_HOSTWAIT)) {
            depth++;
            ts_ctrl(TSS0, ACTIVATE);
        }
    //1.
    if //IF TaskString2 WAITS RELEASE BY HOST PROCESSOR, AND THERE IS DATA IN FIFO
        if ((depth>0) && (ts_status(TSS1)==TSS_HOSTWAIT)) {
            depth--;
            ts_ctrl(TSS1, ACTIVATE);
        }
}
}

```

图16

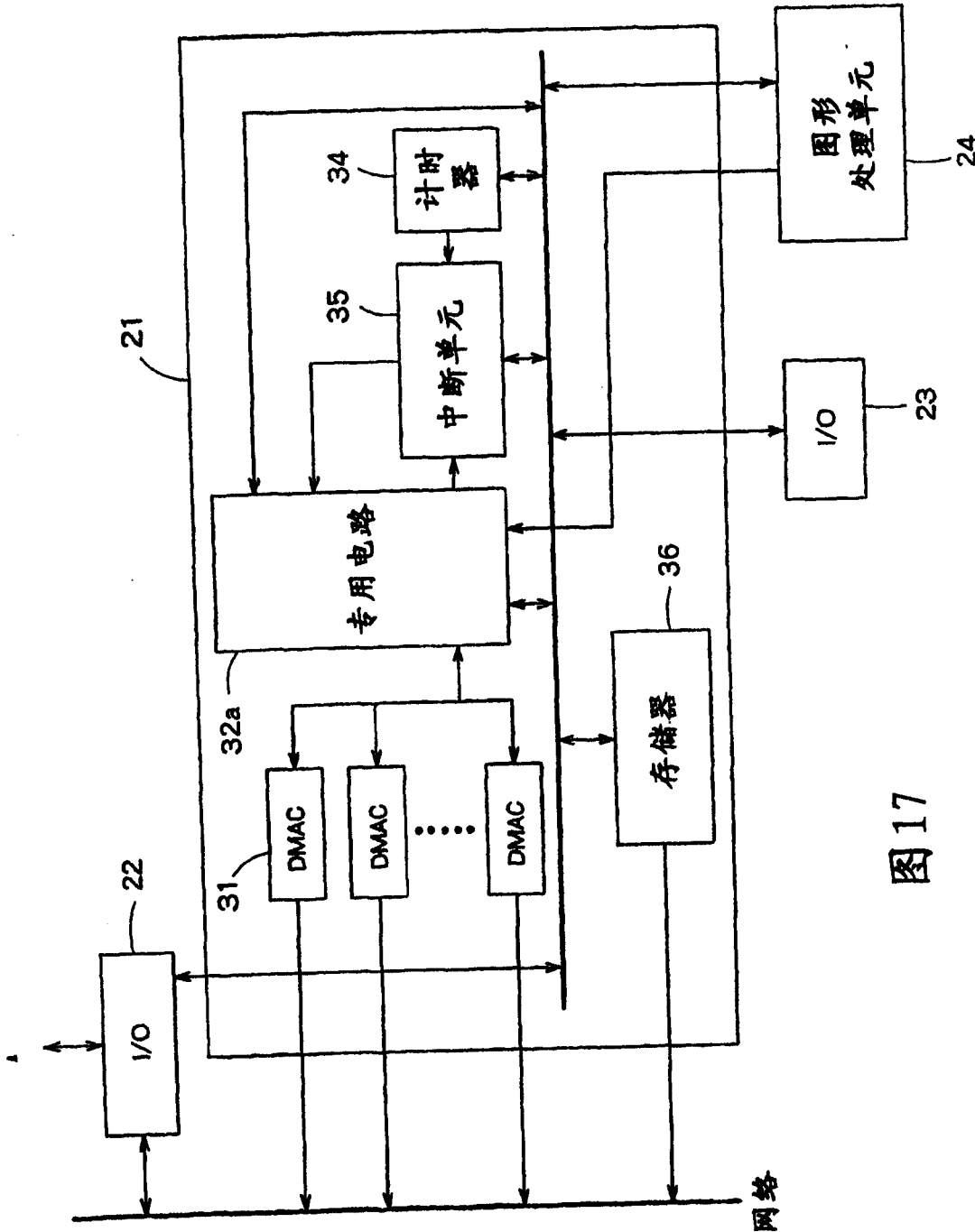


图17

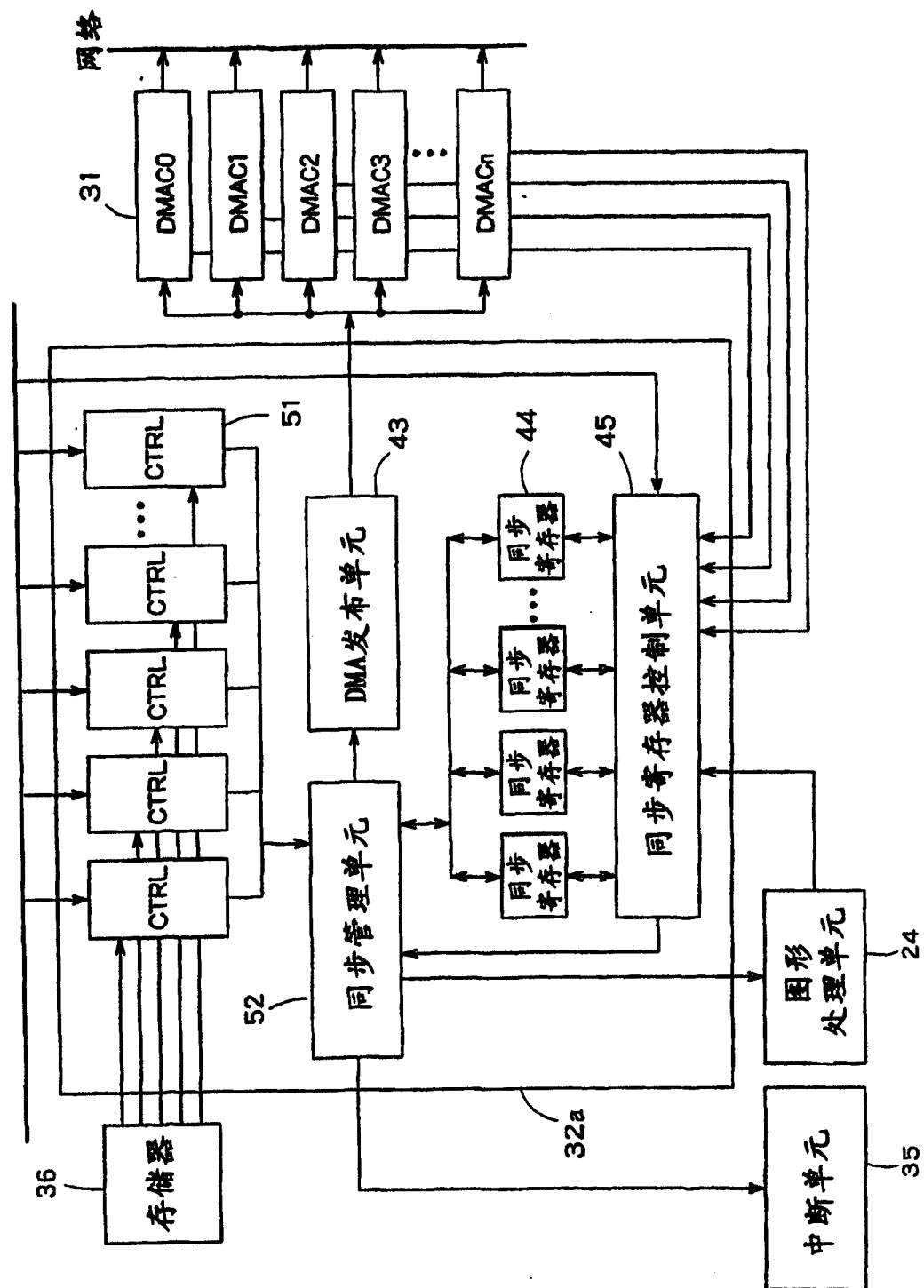


图18

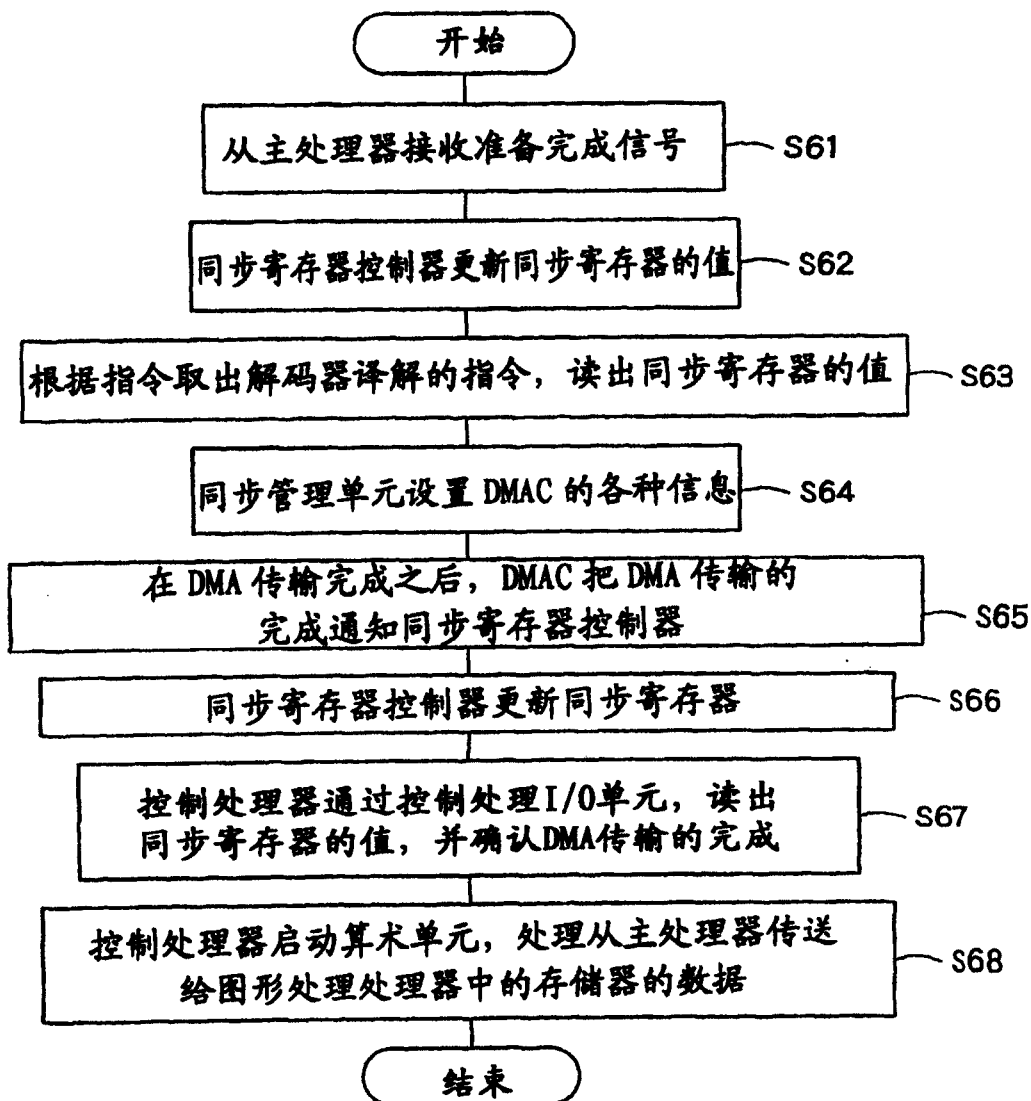


图 19

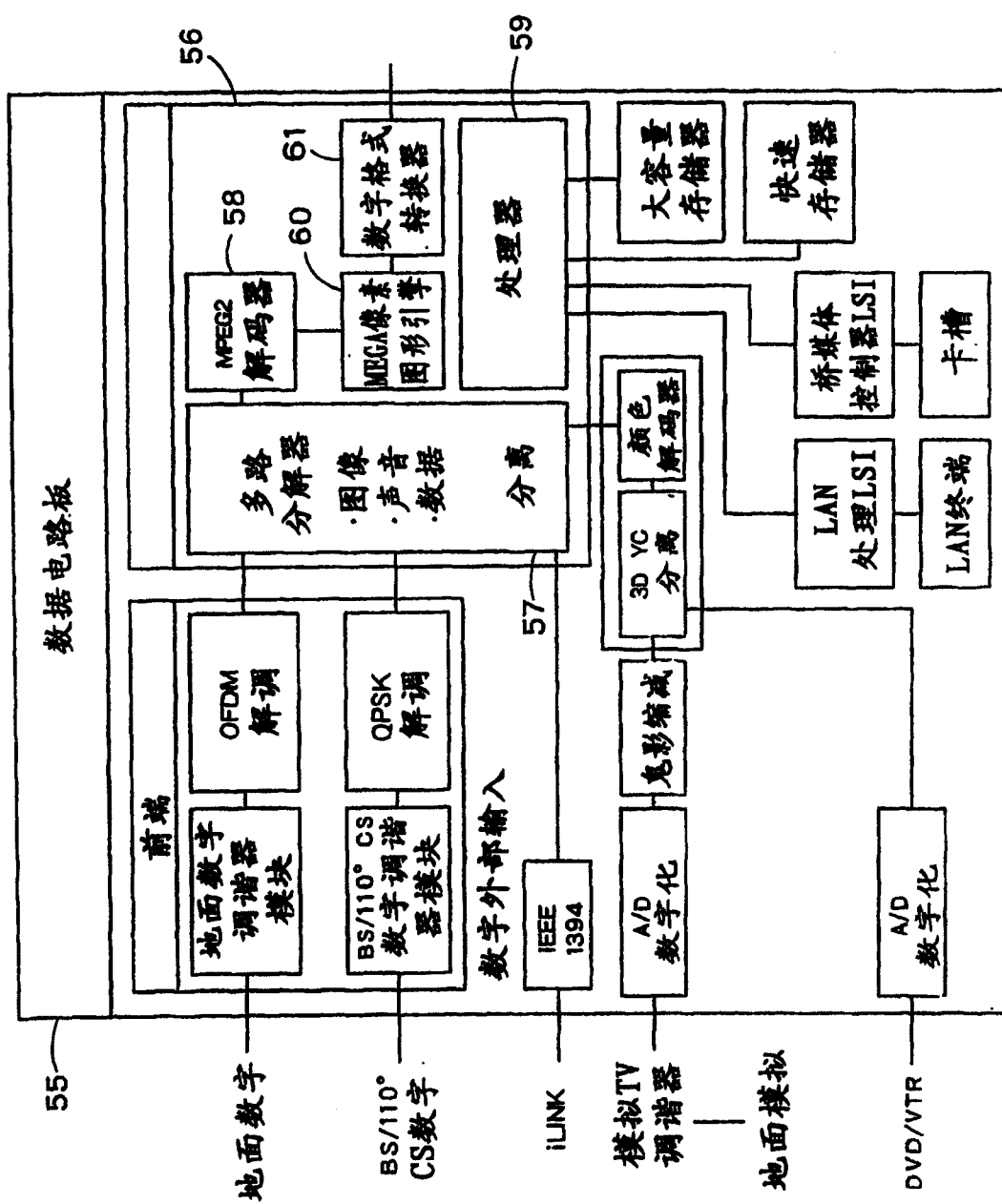


图 20

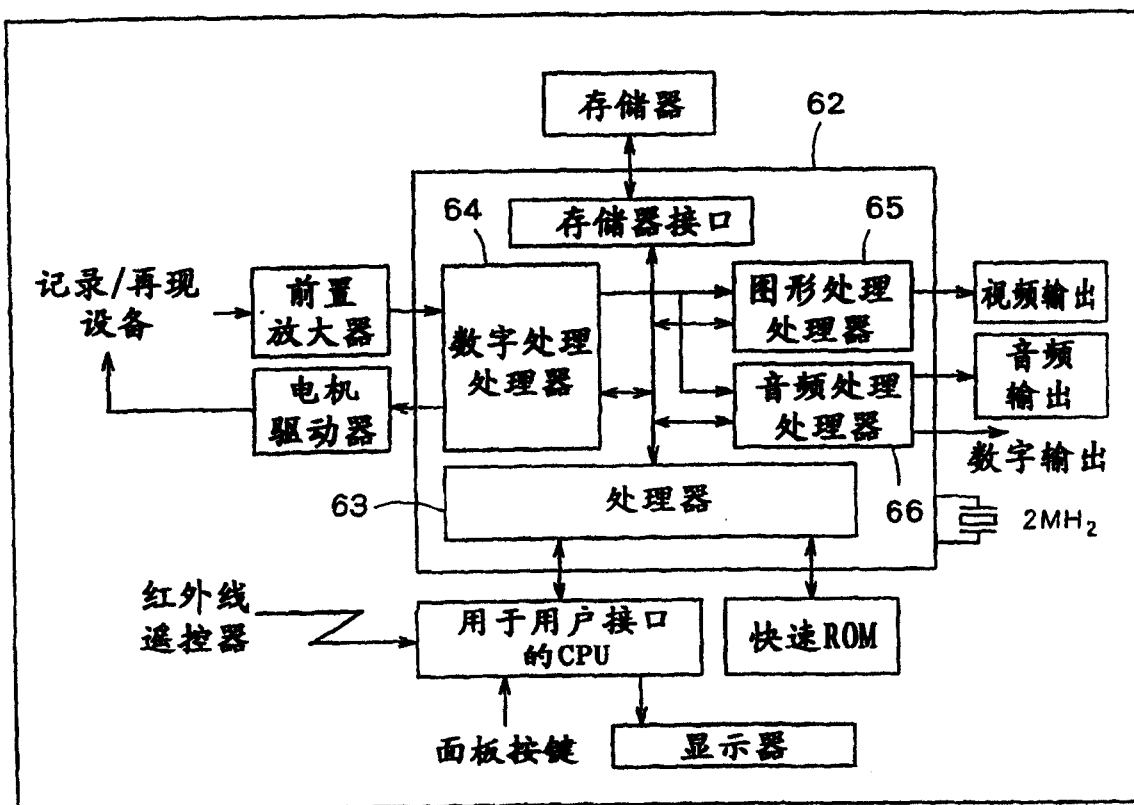


图 21