



(45)授权公告日 2019.04.12

A·恩格-哈斯特维德

申请公布号 CN 105659285 A

代理人 吕俊刚

(51) Int.Cl.

G06T 1/60(2006.01)

G06F 12/0815(2016.01)

G06F 12/0842(2016.01)

(56)对比文件

CN 102103482 A, 2011.06.22,

CN 102713837 A, 2012.10.03.

CN 101826056 A, 2010.09.08.

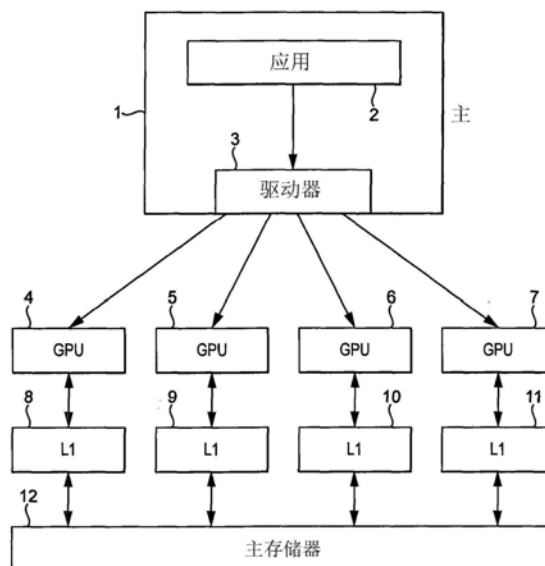
TW 201245976 A, 2012.11.16,

宙査员 原野

权利要求书7页 说明书26页 附图11页

数据处理系统、操作数据处理系统的方法和计算机可读介质

数据处理系统、操作数据处理系统的方法和计算机可读介质。一种数据处理系统包括可操作以启动执行线程的原子存储器请求的一个或多个处理器(4、5、6、7)及被用于在要执行原子存储器操作时存储执行原子存储器操作所需的数据的多个数据高速缓存(8、9、10、11)。当要针对数据高速缓存执行原子操作时,在来自存储器的“真实”高速缓存行抵达高速缓存前,要存取同一存储器位置的原子操作的结果被累积在所述数据高速缓存中的临时高速缓存行中。接着,在所述高速缓存行抵达所述高速缓存时,存储在所述临时高速缓存行中的所述原子操作的累积结果与来自存储器的所述高速缓存行组合。一旦所述高速缓存行抵达所述高速缓存,还可以重建单个原子值。



1. 一种操作数据处理系统的方法,该数据处理系统包括可操作以启动执行线程的原子存储器请求的一个或更多个启动器以及被用于在要执行原子存储器操作时存储执行原子存储器操作所需的数据的一个或更多个数据高速缓存,所述方法包括:

针对要一个接一个地针对同一存储器地址执行特定原子操作的一个或更多个执行线程的集合,所述原子操作具有关联的算术运算:

在执行所述集合中的第一线程的所述原子操作之前,确定要存储执行所述原子操作所需的数据的所述数据高速缓存是否已经存储有要被执行所述原子操作的所述存储器地址处的数据,并且如果所述数据高速缓存尚未存储该数据,则:

发送存储器请求,以将包含具有要被执行所述原子操作的所述存储器地址处的数据的数据项的高速缓存行装入所述数据高速缓存中;

在所述数据高速缓存中为所述原子操作分配临时高速缓存行;

在所述临时高速缓存行中为所述原子操作准备所述数据项;

使所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项中;

如果一个或更多个执行线程的所述集合包括超过一个线程,则针对所述数据高速缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项依次执行所述集合中的一个或更多个随后线程的所述原子操作,直到包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存为止;以及

当包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,利用所述临时高速缓存行中的用于所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于所述存储器地址的所述数据项作为另一运算对象,来执行所述原子操作的所述算术运算,

由此提供结果数据项,所述结果数据项包含针对要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果;以及

存储包含所述结果数据项的结果高速缓存行。

2. 根据权利要求1所述的方法,其中,执行线程的所述集合包括多个执行线程,并且所述方法包括:

在包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存之前,针对所述数据高速缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项依次执行多个线程的所述原子操作,以在包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存之前,在所述临时高速缓存行中累积多个执行线程的所述原子操作的结果。

3. 根据权利要求1所述的方法,所述方法包括:

利用同一临时高速缓存行来累积执行线程的两个或更多个不同集合的原子操作的结果,执行线程的每一个集合都针对各自不同的存储器地址来执行特定的原子操作,所述原子操作具有关联的算术运算;

并且其中,所述方法包括:

针对执行线程的每一个集合,发送存储器请求以将包含相应数据项的高速缓存行装入

所述数据高速缓存中,每一个相应数据项都包含当前的线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据;

使线程的每一个集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于当前的线程的所述集合的相应存储器地址的所述数据项中;

针对包括超过一个线程的一个或更多个执行线程的每一个集合,针对所述数据高速缓存中的所述临时高速缓存行中的、用于当前的线程的所述集合的所述存储器地址的所述数据项,依次执行线程的所述集合中的一个或更多个随后线程的所述原子操作,直到包含线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存为止;以及

在包含线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,针对线程的每一个相应集合,利用所述临时高速缓存行中的用于线程的所述集合的所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于线程的所述集合的所述存储器地址的所述数据项作为另一运算对象,来执行线程的所述集合的所述原子操作的所述算术运算,由此提供线程的所述集合的结果数据项,所述结果数据项包含针对要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果;以及

针对线程的每一个集合,存储包含所述结果数据项的所述结果高速缓存行。

4. 根据权利要求1所述的方法,其中,所述数据处理系统具有多个启动器,每一个启动器都具有相应的数据高速缓存。

5. 根据权利要求1所述的方法,其中,所述原子操作包括以下中的一项:整数加;逐位AND、逐位OR、逐位XOR;最小值;最大值;饱和无符号整数加;整数乘;浮点加;浮点乘;整数加,常数K取模;交换;以及比较并交换。

6. 根据权利要求1所述的方法,其中,所述临时高速缓存行中的每一个数据项是通过将该数据项初始化成用于当前的原子操作的所述算术运算的标识值来准备的;或者所述临时高速缓存行中的每一个数据项是通过将该数据项标记为未初始化来准备的。

7. 根据权利要求1所述的方法,所述方法还包括:

在等待所请求的所述高速缓存行抵达所述数据高速缓存的同时,通过从要使用同一存储器地址的排队的原子存储器操作的队列取得各个原子存储器请求并且针对所述临时高速缓存行依次执行各个原子存储器请求,将所述要使用同一存储器地址的排队的原子存储器操作累积到所述临时高速缓存行中。

8. 根据权利要求6所述的方法,其中:

所述临时高速缓存行包含多个数据项,并且所述临时高速缓存行中的每一个数据项都是通过将该数据项设置成用于所述算术运算的所述标识值来准备的;并且

使线程的所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于线程的所述集合的要被执行所述原子操作的相应存储器地址的所述数据项中的步骤包括:利用在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项中存储的所述标识值来执行所述第一线程的所述原子操作的所述算术运算,并将结果写入所述临时高速缓存行中的、用于要被执行所

述原子操作的所述存储器地址的所述数据项;

并且,所述方法还包括:当包含要被执行所述原子操作的所述存储器地址或多个存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,通过针对所述高速缓存行和所述临时高速缓存行中的各个相应的数据项对来执行所述原子操作的所述算术运算,将所述高速缓存行与所述临时高速缓存行组合起来以提供所述结果高速缓存行。

9. 根据权利要求1所述的方法,其中:

所述临时高速缓存行包含多个数据项,并且所述临时高速缓存行中的每一个数据项都是通过将该数据项标记为未初始化来准备的;并且

使线程的所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于线程的集合的要被执行所述原子操作的相应存储器地址的所述数据项中的步骤包括:将所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值写入所述临时高速缓存行中的、用于线程的所述集合的要被执行所述原子操作的所述存储器地址的所述数据项中;并且所述方法包括:接着将所述临时高速缓存行中的所述数据项标记为被初始化;

并且,所述方法还包括:当包含要被执行所述原子操作的所述存储器地址或多个存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时:

针对所述临时高速缓存行中的被标记为被初始化的任何数据项,利用所述临时高速缓存行中的用于所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于所述存储器地址的所述数据项作为另一运算对象,来执行所述原子操作的所述算术运算,由此提供所述存储器地址的所述结果数据项;以及

针对所述临时高速缓存行中的任何未初始化的数据项,将传入的高速缓存行中的所述数据项不加改变地复制到所述结果高速缓存行。

10. 根据权利要求1所述的方法,所述方法还包括:针对返回型原子操作,当针对所述临时高速缓存行执行该原子操作时,不接着将所述线程的所述原子存储器操作返回至所述启动器,而是将所述原子存储器操作放置在用于延迟的原子存储器请求的缓冲器中。

11. 根据权利要求10所述的方法,所述方法还包括:在将原子存储器请求放置在所述用于延迟的原子存储器请求的缓冲器中时,保持该原子存储器请求的原始变元值,或者在将原子存储器请求放置在所述用于延迟的原子存储器请求的缓冲器中时,利用从所述临时高速缓存行读取的数据值来替换所述原子存储器请求的所述原始变元值。

12. 根据权利要求10所述的方法,所述方法还包括:一旦组合了所述临时高速缓存行和所请求的高速缓存行来提供所述结果高速缓存行,就接着利用来自所请求的高速缓存行的所述存储器地址或多个存储器地址中的数据来执行与当前的所述存储器地址或多个存储器地址相关的任何延迟的存储器请求。

13. 根据权利要求10所述的方法,所述方法还包括:针对临时高速缓存行,跟踪哪些延迟的存储器请求正在等待所请求的高速缓存行。

14. 根据权利要求1所述的方法,其中,所述数据处理系统包括图形处理系统以执行图形处理操作,所述图形处理系统包括多个图形处理流水线,各个图形处理流水线包括执行图形着色器程序的一个或更多个可编程着色器级。

15. 一种数据处理系统,该数据处理系统包括:

一个或多个启动器,所述一个或多个启动器可操作以启动执行线程的原子存储器请求;以及

一个或多个数据高速缓存,所述一个或多个数据高速缓存用于在要执行原子存储器操作时存储执行所述原子存储器操作所需的数据,

其中,所述一个或多个数据高速缓存被设置成,针对要一个接一个地针对同一存储器地址执行特定原子操作的一个或多个执行线程的集合,所述原子操作具有关联的算术运算:

在执行所述集合中的第一线程的所述原子操作之前,确定要存储执行所述原子操作所需的数据的所述数据高速缓存是否已存储有要被执行所述原子操作的所述存储器地址处的数据,并且如果所述数据高速缓存尚未存储该数据,则:

发送存储器请求,以将包含具有用于要被执行所述原子操作的所述存储器地址处的数据的数据项的高速缓存行装入所述数据高速缓存中;

在所述数据高速缓存中为所述原子操作分配临时高速缓存行;

在所述临时高速缓存行中为所述原子操作准备所述数据项;

使所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项中;

如果一个或多个执行线程的所述集合包括超过一个线程,则针对所述数据高速缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项依次执行所述集合中的一个或多个随后线程的所述原子操作,直到包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存为止;以及

当包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,利用所述临时高速缓存行中的用于所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于所述存储器地址的所述数据项作为另一运算对象,来执行所述原子操作的所述算术运算,

由此提供结果数据项,所述结果数据项包含针对要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果;以及

存储包含所述结果数据项的结果高速缓存行。

16. 根据权利要求15所述的系统,其中,执行线程的所述集合包括多个执行线程,并且所述一个或多个数据高速缓存被配置成:

在包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存之前,针对所述数据高速缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项依次执行多个线程的所述原子操作,以在包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存之前,在所述临时高速缓存行中累积多个执行线程的所述原子操作的结果。

17. 根据权利要求15所述的系统,其中,所述一个或多个数据高速缓存被配置成:

利用同一临时高速缓存行来累积执行线程的两个或多个不同集合的原子操作的结果,执行线程的每一个集合都针对各自不同的存储器地址来执行特定的原子操作,所述原子操作具有关联的算术运算:

并且所述一个或更多个数据高速缓存被配置成：在利用同一临时高速缓存行来累积执行线程的两个或更多个不同集合的原子操作的结果时：

针对执行线程的每一个集合，发送存储器请求以将包含相应数据项的高速缓存行装入所述数据高速缓存中，每一个相应数据项都包含当前的线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据；

使线程的每一个集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于当前的线程的所述集合的相应存储器地址的所述数据项中；

针对包括超过一个线程的一个或更多个执行线程的每一个集合，针对所述数据高速缓存中的所述临时高速缓存行中的、用于当前的线程的所述集合的所述存储器地址的所述数据项，依次执行线程的所述集合中的一个或更多个随后线程的所述原子操作，直到包含线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存为止；以及

在包含线程的所述集合的、要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时，针对线程的每一个相应集合，利用所述临时高速缓存行中的用于线程的所述集合的所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于线程的所述集合的所述存储器地址的所述数据项作为另一运算对象，来执行线程的所述集合的所述原子操作的所述算术运算，由此提供线程的所述集合的结果数据项，所述结果数据项包含针对要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果；以及

针对线程的每一个集合，存储包含所述结果数据项的所述结果高速缓存行。

18. 根据权利要求15所述的系统，其中，所述数据处理系统具有多个启动器，每一个启动器都具有相应的数据高速缓存。

19. 根据权利要求15所述的系统，其中，所述原子操作包括以下中的一项：整数加；逐位AND、逐位OR、逐位XOR；最小值；最大值；饱和无符号整数加；整数乘；浮点加；浮点乘；整数加，常数K取模；交换；以及比较并交换。

20. 根据权利要求15所述的系统，其中，所述临时高速缓存行中的每一个数据项都是通过将该数据项初始化成用于当前的原子操作的所述算术运算的标识值来准备的；或者所述临时高速缓存行中的每一个数据项都是通过将该数据项标记为未初始化来准备的。

21. 根据权利要求15所述的系统，其中，所述一个或更多个数据高速缓存被配置成：在等待所请求的高速缓存行抵达所述数据高速缓存的同时，通过从要使用同一存储器地址的排队的原子存储器操作的队列取得各个原子存储器请求并且针对所述临时高速缓存行依次执行各个原子存储器请求，将所述要使用同一存储器地址的排队的原子存储器操作累积到所述临时高速缓存行中。

22. 根据权利要求20所述的系统，其中：

所述临时高速缓存行包含多个数据项，并且所述临时高速缓存行中的每一个数据项都是通过将该数据项设置成用于所述算术运算的所述标识值来准备的；并且

所述一个或更多个数据高速缓存被配置成：

通过利用在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址

的所述数据项中存储的所述标识值来执行所述第一线程的所述原子操作的所述算术运算，并将结果写入所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项，使线程的所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于线程的所述集合的要被执行所述原子操作的相应存储器地址的所述数据项中；并且

所述一个或更多个数据高速缓存被配置成：

当包含要被执行所述原子操作的所述存储器地址或多个存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时，通过针对所述高速缓存行和所述临时高速缓存行中的各个相应的数据项对来执行所述原子操作的所述算术运算，

将所述高速缓存行与所述临时高速缓存行组合起来以提供所述结果高速缓存行。

23. 根据权利要求15所述的系统，其中：

所述临时高速缓存行包含多个数据项，并且所述临时高速缓存行中的每一个数据项都是通过将该数据项标记为未初始化来准备的；并且

所述一个或更多个数据高速缓存被配置成：

通过将所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值写入所述临时高速缓存行中的、用于线程的所述集合的要被执行所述原子操作的所述存储器地址的所述数据项中，使线程的所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于线程的所述集合的要被执行所述原子操作的相应存储器地址的所述数据项中；

并且

所述一个或更多个数据高速缓存被配置成：接着将所述临时高速缓存行中的所述数据项标记为被初始化；

并且所述一个或更多个数据高速缓存还被配置成：当包含要被执行所述原子操作的所述存储器地址或多个存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时：

针对所述临时高速缓存行中的被标记为被初始化的任何数据项，利用所述临时高速缓存行中的用于所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于所述存储器地址的所述数据项作为另一运算对象，来执行所述原子操作的所述算术运算，由此提供所述存储器地址的所述结果数据项；以及

针对所述临时高速缓存行中的任何未初始化的数据项，将传入的高速缓存行中的所述数据项不加以改变地复制到所述结果高速缓存行。

24. 根据权利要求15所述的系统，其中，所述一个或更多个数据高速缓存被配置成：针对返回型原子操作，当针对所述临时高速缓存行执行该原子操作时，不接着将所述线程的所述原子存储器操作返回至所述启动器，而是将所述原子存储器操作放置在用于延迟的原子存储器请求的缓冲器中。

25. 根据权利要求24所述的系统，其中，所述一个或更多个数据高速缓存被配置成：在将原子存储器请求放置在所述用于延迟的原子存储器请求的缓冲器中时，保持该原子存储器请求的原始变元值，或者在将原子存储器请求放置在所述用于延迟的原子存储器请求的缓冲器中时，利用从所述临时高速缓存行读取的数据值来替换所述原子存储器请求的所述原始变元值。

26. 根据权利要求24所述的系统,其中,所述一个或更多个数据高速缓存被配置成:一旦组合了所述临时高速缓存行和所请求的高速缓存行来提供所述结果高速缓存行,就接着利用来自所请求的高速缓存行的所述存储器地址或多个地址中的数据,来执行与当前的所述存储器地址或多个存储器地址相关的任何延迟的存储器请求。

27. 根据权利要求24所述的系统,其中,所述一个或更多个数据高速缓存被配置成:针对临时高速缓存行,跟踪哪些延迟的存储器请求正在等待所请求的高速缓存行。

28. 根据权利要求15所述的系统,其中,所述数据处理系统包括图形处理系统以执行图形处理操作,所述图形处理系统包括多个图形处理流水线,各个图形处理流水线都包括执行图形着色器程序的一个或更多个可编程着色器级。

29. 一种存储计算机程序的计算机可读介质,该计算机程序包括当该程序在数据处理装置上运行时,适于执行根据权利要求1至14中任一项所述的方法的所有步骤的软件代码。

数据处理系统、操作数据处理系统的方法和计算机可读介质

技术领域

[0001] 本发明涉及数据处理系统,并且具体地,涉及包括一个或更多个可编程处理级(“着色器(shader)”)的图形处理系统的操作。

背景技术

[0002] 如本领域中已知的,图形处理通常按流水线方式来执行,其中一个或更多个流水线级针对数据进行操作以生成最终渲染(render)输出,例如,被显示的帧。许多图形处理流水线现在包括一个或更多个可编程处理级,通称为“着色器”。例如,一图形处理流水线可以包括几何着色器、顶点着色器以及片段(像素)着色器中的一个或更多个,并且通常包括全部。这些着色器是可编程处理级,针对输入的数据值执行着色器程序以生成期望的输出数据集(例如,在顶点着色器的情况下,恰当地变换并照亮顶点数据),以供图形流水线的其余部分处理和/或输出。图形处理流水线的着色器可以共用可编程处理电路,或者它们皆可以是独特的可编程处理单元。

[0003] 图形处理单元(GPU)着色器核心是这样的处理单元,即,其通过针对诸如渲染目标的要生成的图形输出(例如,帧)中的每一个图形项运行小程序来执行图形处理(在这点上,“项”通常是顶点或片段(像素))。这通常使能实现高度并行性,因为典型渲染输出(例如,帧)特征化了相当大量的顶点和片段,其中每一个都可以独立处理。

[0004] 如本领域所已知的,要通过图形处理流水线的指定“着色器”执行的着色器程序将通过需要利用高级着色器编程语言(如GLSL、HLSL、OpenCL等)的图形处理的应用来提供。该着色器程序将由指示按相关语言标准(规范)定义的期望编程步骤的“表达式”构成。接着,该高级着色器程序通过着色器语言编译器翻译成用于目标图形处理流水线的二进制代码。该二进制代码将由“指令”构成,所述指令在用于所指定目标图形处理流水线的指令集规范中指定。用于将着色器语言表达式转换成二进制代码指令的编译处理可以经由编译器内的程序的许多中间表述来进行,如本领域所已知的。因此,以高级着色器语言编写的程序可以被翻译成编译器特定中间表述(并且在该编译器内可以有几个连续中间表述),并且最终的中间表述被翻译成用于目标图形处理流水线的二进制代码指令。

[0005] 在现代图形处理单元中,常见的是使指令执行“原子”存储器操作。“原子”存储器操作是读取存储器位置、在存储器值与运算对象值(典型地,寄存器值)之间执行算术运算,并接着将算术运算的结果写回至同一存储器位置的操作序列。执行该序列操作以使得对于每一个观察者来说,看起来好像该序列根本未执行或者全部执行。其作为一个不可分割的单元加以执行,因此,叫做“原子”。

[0006] 作为该原子的部分执行的算术运算通常是既可交换又可结合的简单运算。这种运算的常见例子是:整数加(Add);逐位与(AND)、OR、XOR;最小值;最大值。

[0007] 常见但不可交换的其它原子操作有:

[0008] 交换(Swap)(还已知为互换(Exchange))-寄存器值被写入存储器位置,接着将该存储器位置的旧值返回至着色器(该运算可结合);

[0009] 比较并交换 (Compare-And-Swap) (比较与互换 (Compare and Exchange)) - 提供两个寄存器值, 比较存储器值与两个寄存器值中的第一个, 并且如果它们比较相同, 则将第二个寄存器值写入该存储器位置 (如果该比较值保持恒定, 则该运算可结合)。

[0010] 原子存储器操作通常以两种变型出现: “返回 (Return)” 和 “不返回 (NoReturn)”。“Return” 变型向着色器返回在执行 “原子” 操作之前存在于存储器位置处的数据。“NoReturn” 变型不向着色器返回任何值。典型地讲, 在 GPU 中, 上面列出的所有运算在两个变型中可用 (Swap 除外, 其在 NoReturn 变型中不可用)。

[0011] 用于 GPU 原子的使用案例的典型示例包括增量存储器分配器 (“Return” 类型的整数加)、直方图 (“NoReturn” 类型的整数加) 以及边界框计算 (“NoReturn” 类型的最小/最大值)。

[0012] 如果在同一存储器位置上执行多个原子, 则需要将它们串行化。这是必需的, 以保持它们的每一个已经完全执行或者根本未执行的可观察语义。这因而可能导致具有多个数据高速缓存和相干协议的系统中的吞吐量问题。

[0013] 例如, 在这种排布结构中, 希望针对指定存储器位置执行原子操作的处理核心首先需要将存储针对所讨论的存储器位置的数据的高速缓存行从存储器装入 (fetch) 到其高速缓存的行中。接着, 当该高速缓存行存在于该高速缓存中时, 可以利用高速缓存行中的数据一次一个地顺序执行原子操作。然而, 在该高速缓存行驻留在所讨论的核心的高速缓存中时, 使用不同高速缓存并且需要所讨论的数据 (高速缓存行) 的其它核心无法执行它们的原子操作, 使得那些核心的原子操作必须停顿, 直到当前具有该高速缓存行的高速缓存释放该高速缓存行为止 (以使该高速缓存行可以接着被装入到另一核心的高速缓存中)。

[0014] 由此, 虽然在核心等待高速缓存行 (即, 该高速缓存行未存在于其高速缓存中) 时, 针对该核心的原子操作被迟延并且根本无法执行。类似地, 当该高速缓存行正从一个高速缓存向另一高速缓存传送时, 完全不能针对该高速缓存行中的数据执行原子操作。

[0015] 这都具有这样的影响: 对使用多个高速缓存的系统中的同一存储器位置的原子操作的吞吐量可能相对较慢, 因为处理核心或多个核心在可以执行原子操作之前不得不等待高速缓存行传送至它们的相应高速缓存。

[0016] 图1和图2例示了这种操作。

[0017] 图1示出了具有四个处理核心40、41、42、43的示例性数据处理系统, 每一个处理核心都具有相应的一级 (L1) 高速缓存44、45、46、47。每一个处理核心都能够执行原子操作, 并且当原子操作要针对指定存储器位置执行时, 包含该存储器位置处的数据的高速缓存行首先从主存储器48加载到所讨论的核心的相应的一级高速缓存中的高速缓存行中。如上所述, 如果处理核心40、41、42、43中的超过一个处理核心正在执行针对主存储器48中的同一位置的原子操作, 则来自主存储器48的高速缓存行将需要按串行方式加载到需要该高速缓存行的核心的相应的一级高速缓存中, 以便保持必需的原子操作语义。因而, 这具有这样的影响, 即, 如果高速缓存行处于其中一个处理核心的相应的一级高速缓存中, 则需要该高速缓存行的任何其它处理核心必须迟延其原子操作, 直到该高速缓存行被装入针对所讨论的核心的高速缓存中为止。

[0018] 图2A和图2B例示了该操作, 并且例如以示例的方式示出了要针对其中一个处理核心40执行原子操作 (在这种情况下是原子ADD操作) 的线程A、B、C、D的集合50。在该处理开始

时,要执行的线程50准备就绪由核心40来执行,但假定来自主存储器的相关高速缓存行(数据)还没有被装入到处理核心40的高速缓存41中。

[0019] 发送请求57以将相关的高速缓存行装入处理核心40的高速缓存41中。同时,在高速缓存41中留出了用于所装入的高速缓存行的空间58(但该高速缓存行尚未抵达)。

[0020] 接着,稍后,包含所讨论的存储器位置(地址)处的数据的所请求的高速缓存行51抵达处理核心40的数据高速缓存41。接着,执行第一原子操作52(针对线程A)。如图2所示,针对线程A的原子操作52使存储器地址中的值递增一(+1),并同样地,所讨论存储器位置的高速缓存行51中的值也递增一(在这种情况下,增至“124”)。

[0021] 线程A还将该存储器地址的前一值54(在这种情况下,“123”)返回至该处理核心(如在这种情况下,假定该原子操作具有返回类型)。

[0022] 接着,如图2A所示,假定针对高速缓存行51中的数据进行监听(snoop)请求59(即,该系统的另一部分请求高速缓存行51中的数据),并且响应于此,将高速缓存行51从处理核心40的数据高速缓存41逐出(evict)33。线程B、C、D的原子操作因此必须延迟,因为用于处理核心40的数据高速缓存41不再包含相关的高速缓存行。由此,如图2A所示,必须发送用于装入高速缓存行的进一步请求32。

[0023] 接着,一旦该相关高速缓存行再次抵达所讨论处理核心40的数据高速缓存41,下一线程(线程B)就执行其原子加(add)操作53,将该存储器地址的高速缓存行中的值增加二(+2),并且还将该存储器地址的前一值56(在这种情况下,“124”)返回至该处理核心。

[0024] 接着,假定监听请求再次导致将高速缓存行从数据高速缓存41逐出,同样地,在线程C可以执行其原子操作之前,发出进一步装入请求并等待,直到该高速缓存行抵达数据高速缓存41为止。

[0025] 一旦该高速缓存行再次抵达该处理核心的数据高速缓存,下一线程C就可以执行其原子加操作55,并且返回其相关输出值,对于线程D来说同样,如图2B所示。

[0026] 在该操作结束时,处理核心40的数据高速缓存41中的高速缓存行将存储原子操作的累积结果,该累积结果接着可以返回至主存储器和/或传递至用于另一核心的高速缓存,使得该另一核心针对所讨论存储器位置执行其原子操作。

[0027] 可以看出,在该操作中,如果在执行原子操作的同时接收到大量监听请求,则原子操作的吞吐量可被显著延迟。

发明内容

[0028] 本申请人认为,对于处理使用多个(例如L1)高速缓存的系统(例如,在包括一个或更多个着色器级的图形处理流水线中)中的原子操作来说仍然存在改进的余地。

[0029] 根据本发明第一方面,提供了一种操作数据处理系统的方法,该数据处理系统包括:一个或更多个启动器,在工作上,所述一个或更多个启动器用于启动用于执行线程的原子存储器请求;以及一个或更多个数据高速缓存,所述一个或更多个数据高速缓存被用于在要执行原子存储器操作时存储执行原子存储器操作所需的数据,所述方法包括:

[0030] 针对要一个接一个地执行针对同一存储器地址的特定原子操作的一个或更多个执行线程的集合,所述原子操作具有关联算术运算:

[0031] 在执行所述集合中的第一线程的所述原子操作之前,确定要存储执行所述原子操

作所需的数据的所述数据高速缓存是否已经存储有要被执行所述原子操作的所述存储器地址处的数据,并且如果所述数据高速缓存尚未存储该数据,则:

[0032] 发送存储器请求,以将包括含有用于要被执行所述原子操作的所述存储器地址处的数据的数据项的高速缓存行装入所述数据高速缓存中;

[0033] 在所述数据高速缓存中为所述原子操作分配临时高速缓存行;

[0034] 在所述临时高速缓存行中为所述原子操作准备所述数据项;

[0035] 使所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项中;

[0036] 如果一个或更多个执行线程的所述集合包括超过一个线程,则针对所述数据高速缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项,依次执行所述集合中的一个或更多个随后线程的所述原子操作,直到包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存为止;并且

[0037] 当包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,利用所述临时高速缓存行中的所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的所述存储器地址的所述数据项作为另一运算对象,执行所述原子操作的所述算术运算,由此提供包含要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果的结果数据项;以及

[0038] 存储包含所述结果数据项的结果高速缓存行。

[0039] 根据本发明第二方面,提供了一种数据处理系统,该数据处理系统包括:

[0040] 一个或更多个启动器,所述一个或更多个启动器在工作上可以启动用于执行线程的原子存储器请求;以及

[0041] 一个或更多个数据高速缓存,所述一个或更多个数据高速缓存被用于在执行原子存储器操作时存储执行原子存储器操作所需的数据;

[0042] 其中,所述一个或更多个数据高速缓存被设置成,针对要一个接一个地针对同一存储器地址执行特定原子操作的一个或更多个执行线程的集合,所述原子操作具有关联算术运算:

[0043] 在执行所述集合中的所述第一线程的所述原子操作之前,确定要存储执行所述原子操作所需的所述数据的所述数据高速缓存是否已存储有要被执行所述原子操作的所述存储器地址的数据,并且如果所述数据高速缓存尚未存储所述数据,则:

[0044] 发送存储器请求,以将包括含有用于要被执行所述原子操作的所述存储器地址处的数据的数据项的高速缓存行装入所述数据高速缓存中;

[0045] 在所述数据高速缓存中为所述原子操作分配临时高速缓存行;

[0046] 在所述临时高速缓存行中为所述原子操作准备所述数据项;

[0047] 使所述集合中的所述第一线程的所述原子操作的所述算术运算的运算对象值存储在所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项中;

[0048] 如果一个或更多个执行线程的所述集合包括超过一个线程,则针对所述数据高速

缓存中的所述临时高速缓存行中的、用于要被执行所述原子操作的所述存储器地址的所述数据项,依次执行所述集合中的一个或多个随后线程的所述原子操作,直到包含要被执行所述原子操作的所述存储器地址处的所述数据的所述高速缓存行抵达所述数据高速缓存为止;并且

[0049] 当包含要被执行所述原子操作的所述存储器地址处的数据的所述高速缓存行抵达所述数据高速缓存时,利用所述临时高速缓存行中的用于所述存储器地址的所述数据项作为一个运算对象并且利用抵达的高速缓存行中的用于所述存储器地址的所述数据项作为另一运算对象,执行所述原子操作的所述算术运算,由此提供包含要被执行所述原子操作的所述存储器地址的所述原子存储器操作的组合结果的结果数据项;以及

[0050] 存储包含所述结果数据项的结果高速缓存行。

[0051] 本发明涉及在具有存储来自主存储器的数据的数据的要被用于原子存储器操作的多个数据高速缓存的数据处理系统中执行原子存储器操作。在本发明中,在执行原子存储器操作,并且用于该原子存储器操作的数据未存在于要被执行该原子存储器操作的数据高速缓存中时,不延迟执行该原子存储器操作直到该数据存在于该高速缓存中为止,而是在该高速缓存中分配用于该原子操作的临时高速缓存行,并接着在等待期望的“真实”高速缓存行抵达该高速缓存期间,针对该临时高速缓存行执行该原子存储器操作。而且,如下面将进一步讨论的,可以针对临时高速缓存行执行同一类型的多个原子操作。接着,当包含来自主存储器的数据的高速缓存行被装入该高速缓存中时,将临时高速缓存行与传入的高速缓存行(主存储器数据)组合,以提供用于该原子存储器操作的所需输出(结果)(即,合并以针对所讨论的存储器的地址中的数据执行原子存储器操作)。

[0052] 如下面将进一步讨论的,这允许有效地执行原子存储器操作,使得当用于该原子操作的、包含该存储器位置的数据的高速缓存行抵达该数据高速缓存时,可以立刻将一组累积的原子操作提交给该高速缓存行(按单一时钟周期),同时(如下面将进一步讨论的)仍完全保持原子操作的语义。这因而易于将高速缓存行为了要执行的一组原子存储器操作必须存在的时间量减到最小。这接着可以显著改进原子存储器操作性能,并且导致具有多个数据高速缓存的系统中的原子存储器操作的吞吐量的增加。

[0053] 而且,因为本发明的方式的操作在数据高速缓存的级别执行,所以本发明可以仅需要针对所讨论的数据高速缓存级别的操作的改变,而不需要针对数据处理系统的任何其它子系统(例如,更低的高速缓存级别、主存储器操作等)的任何改变。例如,在已支持相关存储器和原子的系统中,本发明可以仅需要针对高速缓存(例如,L1)的局部改变,而该高速缓存(例如,L1)用于与其所有者(master)并且与主存储器子系统通信的任何协议不受本发明的使用和存在的影响。

[0054] 如下面将进一步讨论的,对于不返回和返回原子来说需要上述步骤,而对返回原子来说,可能需要进一步的步骤(并且被优选地执行)。

[0055] 可以请求原子存储器操作的启动器可以包括任何适合的处理器或处理单元。在优选实施方式中,启动器包括相应的处理核心。在优选实施方式中,每一个启动器都包括执行流水线的执行级,所述执行级优选为可编程执行级,所述执行流水线包括执行用于执行针对执行线程的数据处理操作的指令的一个或多个执行级,所述执行级优选为可编程执行级。

[0056] 如上所述,相信本发明在图形处理系统中将具有特定应用。因而,该数据处理系统优选地包括图形处理系统,并且所述启动器优选地包括图形处理流水线,和/或图形处理流水线的执行级,所述执行级优选地采用执行图形着色器程序以执行图形处理操作的一个或更多个可编程图形着色级(着色器)的形式。然而,本发明可等同地应用于其它形式的数据处理系统,如CPU。

[0057] 本发明可应用于存在可以存储用于原子存储器操作的数据的多个数据高速缓存的情况。本发明可以被用于这样的系统,即,存在可以执行原子操作的单个启动器单元(例如,GPU),其具有可以被用于存储用于执行原子操作的数据的多个数据高速缓存,和/或本发明可以被用于这样情况,即,存在多个启动器单元(例如,GPU),所述多个启动器皆将它们自身的相应数据高速缓存或多个高速缓存用于存储来自主存储器的数据以执行原子存储器操作。在优选实施方式中,所述数据处理系统具有多个启动器单元,每一个启动器单元都具有相应的数据高速缓存。

[0058] 所述数据高速缓存可以是数据处理系统的任何适合的数据高速缓存。在优选实施方式中,它们是一级(L1)高速缓存。然而,这不是必要的,而且若希望的话,本发明可以在其它高速缓存级别来实现(尽管所有数据高速缓存应当(并且优选地)具有同一级别,因此在本发明以数据处理系统的二级高速缓存实现的情况下,所有数据高速缓存优选为L2高速缓存)。

[0059] 可以按本发明的方式处理的原子存储器操作可以是任何适合和希望的原子存储器操作。在优选实施方式中,存在将按照本发明的方式处理的一组特定的选择的原子存储器操作。在优选实施方式中,将按照本发明的方式处理的原子存储器操作包括这样的原子操作,即,对于其来说,作为该原子的一部分执行的算术运算可联合,并且优选为既可交换又可联合。然而,这不是必要的,而且本发明还可以被用于其它原子操作,若希望的话,如没有这些特性的交换(互换)和比较并交换(比较和互换)。

[0060] 在优选实施方式中,将按照本发明的方式处理的原子存储器操作包括以下中的一个或更多个,并且优选为包括全部:整数加(Add);逐位与(AND)、OR、XOR;最小值;最大值;饱和无符号整数加;整数乘;浮点加;浮点乘;整数加、常数K求模。

[0061] 按照本发明的方式处理的原子存储器操作优选地具有恰当的“标识值”(尽管如下将讨论的,这不是必要的)。该“标识值”是用于针对原子操作的算术运算的值,其将使得对于算术运算来说,与该标识值组合的值无变化,(即,使得如果给定的算术运算用运算符%来指示,则该算术运算的标识值为值V,使得对于所有可能的输入值X来说, $V\%X=X\%V=X$)。由此,例如,对于整数ADD操作来说,该标识值为“0”,对逐位OR或逐位XOR操作来说,该标识值=“0”,对于逐位AND来说,该标识值是全为一的位向量,对于最小值操作来说,该标识值是采用存储器值格式的最大可表示数,而对于最大值操作来说,该标识值是采用存储器值格式的最小可表示数(等等)。

[0062] 用于其它原子存储器操作的标识值可以对应地确定。

[0063] 用于向数据高速缓存装入包含要被执行原子操作的存储器地址处的数据的高速缓存行的存储器请求可以如希望地发送和配置,例如根据所讨论的存储器系统。

[0064] 所述临时高速缓存行可以按任何希望和合适的方式针对原子操作加以分配,例如,根据用于所讨论的系统 and/或高速缓存的正常高速缓存行分配处理。

[0065] 所述临时高速缓存行关联(标记)其与以下中的一个或多个,并且优选为全部:针对原子操作的集合的合适的存储器地址;作为出于按本发明的方式操作的目的的临时高速缓存行的事实;针对作为要累积的原子操作的算术运算。

[0066] 所述临时高速缓存行优选地可以仅被用于累积所讨论的合适原子存储器操作(即,匹配该临时高速缓存行的标记(同一原子操作,同一存储器地址(或者至少同一高速缓存行内的存储器地址)),并且优选地不能以其它方式读取、写入、监听或者按任何其它方式相互作用。换句话说,所述临时高速缓存行优选地被配置成,使得出于旨在处理的原子操作的特定子集以外的所有其它目的,,其有效地不可见于该系统(不“存在”)。

[0067] 在优选实施方式中,被用于标记临时高速缓存行的“原子操作”至少针对所选择的原子操作不仅区分要执行的算术运算的类型,而且区分运算大小。例如,32位原子ADD与64位原子ADD被优选地视为两个不同的操作。类似的是,如果底层实现区分字节顺序,则,这也优选地加以考虑,至少针对所选择的类型的原子操作。(然而,这些区别不需要用于所有类型的原子操作)。

[0068] 如一般的高速缓存行那样,所述临时高速缓存行的“数据”部分将包含一个或多个数据项(数据条目)。所述临时高速缓存行中的数据项的数量将取决于高速缓存的配置(例如,每一个高速缓存行可以存储多少数据)和所讨论的原子操作的性质(例如,其是针对32位还是16位值来操作等)。所述临时高速缓存行应当被划分成尺寸适于所执行的原子操作的数据项。所述临时高速缓存行中可以仅存在一个数据项,但通常将存在多个数据项(其中,该高速缓存行中的、对应于用于原子操作的存储器地址的特定数据项接着在等待包含来自该存储器地址的数据项的高速缓存行抵达之际,被用于累积该原子操作的结果)。

[0069] 在临时高速缓存行中准备数据项应当是这样的,即,在等待包含所讨论的存储器地址的数据的高速缓存行抵达之际,允许临时高速缓存行恰当地累积一线程或多个线程的原子操作。

[0070] 在一个优选实施方式中,所述临时高速缓存行中的每一个数据项都通过将其初始化(设置)成用于所讨论的原子操作的所述算术运算的标识值来准备。由此,在优选实施方式中,用于原子操作的临时高速缓存行通过将该临时高速缓存行中的所有数据项设置成所讨论的原子操作的算术运算的标识值来准备。(如上所述,该算术运算的“标识值”是当该算术运算利用另一值来执行时,保留该另一值不变的值)。

[0071] 在另一优选实施方式中,临时高速缓存行中的每一个数据项都通过被标记为无效(即,事实上,通过被标记为“未初始化”)来准备。出于该目的,每一个数据项例如都可以具有关联的无效或未初始化位或标志。这因而可以允许没有标识值的原子算术运算按照本发明的方式来处理。

[0072] 因此,在另一优选实施方式中,在准备临时高速缓存行中每个数据项存储,在临时高速缓存行中每个数据项存储“未初始化”位(“无效”位),其时按照数据项存储,以代替标识值。(若希望的话,即使算术运算具有标识值,也可以进行所述操作)。(应当清楚,在临时高速缓存行中的数据项被标记为“未初始化”的情况下,该数据项不包含有效数据值,但一旦“初始化”,则数据项将包含有效数据值)。

[0073] 在这种情况下,如果所有原子操作都具有作为底层存储单位(如字节)的多倍的尺寸,则可以每存储单位提供未初始化位,以代替每数据项,只要临时高速缓存行中的每一个

数据项存在至少一个这种位即可。在这种情况下,如果临时高速缓存行被存储在已经能够例如跟踪每字节页面重写标志位(dirty-bit)的高速缓存中,则该页面重写标志位可以被用于提供未初始化位(无效位)功能(若希望的话)。

[0074] 在准备了临时高速缓存行之后,但在来自存储器的高速缓存行抵达之前,针对数据高速缓存中的临时高速缓存行执行针对线程的集合的原子操作(即,针对临时高速缓存行中的、对应于用于要被执行原子操作的存储器地址的数据项来执行该线程的原子操作)。

[0075] 本发明可以在这样的情况下使用,即,正好存在要针对所讨论的存储器地址执行的单个原子存储器请求(即,执行线程的集合仅包括单个线程)。然而,本申请人已经认识到,可以存在这样的情况,即,例如,执行线程的给定集合例如用于一个接一个地针对同一存储器地址执行同一原子操作。在这种情况下,临时高速缓存行可以(并且优选地)被用于累积所述多个原子操作的结果,同时等待来自存储器的高速缓存行抵达该数据高速缓存。

[0076] 由此,在特别优选实施方式中,存在多个执行线程的集合,所述多个执行线程要针对同一存储器地址执行同一原子操作,并且一旦准备了用于所述原子操作的临时高速缓存行,就依次针对该线程集合中的每一个线程执行针对数据高速缓存中的临时高速缓存行的原子操作(使得该临时高速缓存行接着存储该线程集合的原子操作的组合结果),直到包含要执行原子操作的存储器地址处的数据的高速缓存行抵达该数据高速缓存为止。接着,对于针对临时高速缓存行执行了原子操作的线程的集合来说,可以将临时高速缓存行与抵达高速缓存行相组合,以提供包含针对要被执行原子操作的存储器地址的原子存储器操作的结果的结果高速缓存行。

[0077] 在这种情况下,可能的是,在包含该存储器地址的数据的高速缓存行抵达之前,能够针对临时高速缓存行来执行针对线程集合中的所有线程的原子操作,或者可能的是,在包含存储器地址的数据的高速缓存行抵达之前,可以针对临时高速缓存行来处理线程集合中的仅一些线程而非全部线程。

[0078] 为促进该操作,每一个数据高速缓存都优选地具有关联的队列,所述队列存储具有关联的原子操作的执行线程(保持传入原子存储器请求队列)。接着,当临时高速缓存行被分配用于指定原子操作和存储器地址时,可以依次从该队列取得合适的请求,并且针对该临时高速缓存行执行。所述队列中的每一个条目都优选地具有存储器地址、数据项以及与其相关联的存储器请求所需的任何其它元数据。

[0079] 由此,在优选实施方式中,在准备了临时高速缓存行之后,但在来自存储器的数据(高速缓存行)抵达之前,通过依次从该队列取得每一个原子存储器请求并且针对该临时高速缓存行(就如同是正常高速缓存行一样)执行该请求,数据高速缓存将要使用同一存储器地址的排队的(对应)原子存储器操作累积到临时高速缓存行中。这针对使用同一存储器位置并且针对该存储器位置执行同一算术运算(若有的话)的每一个排队的原子存储器请求进行重复,直到包含所讨论存储器位置的“真实”数据的高速缓存行抵达该数据高速缓存为止。

[0080] 针对临时高速缓存行执行第一原子操作应当(并且优选地)将用于第一原子操作(针对第一线程)的算术运算(原子操作对象)的运算对象值存储在临时高速缓存行中的恰当数据项中。如本领域已知的,该原子操作对象值在正常情况下并且在优选实施方式中是用于第一线程的原子操作的寄存器值,但可以在原子操作没有显式寄存器值的情况下等同

为隐式运算对象值。

[0081] 由此,对于通过将用于原子操作的标识值存储在临时高速缓存行中来准备临时高速缓存行中的数据项的情况来说,可以并且优选地这样来完成,即,通过将在所讨论的临时高速缓存行中读取(装入)存储器位置的当前内容(其将成为标识值),利用该标识值和用于该原子操作的第一线程的(第一原子操作的)运算对象值执行算术运算,并接着将结果写回至临时高速缓存行(中的恰当数据项)。

[0082] 由此,在一个优选实施方式中,通过利用存储在临时高速缓存行中的、用于原子操作的算术运算的标识值以及用于原子操作的第一线程的运算对象值,针对第一线程执行原子操作的算术运算,将用于第一线程的原子操作的运算对象值存储在临时高速缓存行中。

[0083] 在通过利用无效(未初始化)位标志临时高速缓存行中的数据项来准备临时高速缓存行中的数据项的情况下,将用于第一原子操作的运算对象值存储在临时高速缓存行中的恰当数据项中可以并且优选地通过将用于所讨论的原子存储器请求的运算对象值存储(写入)临时高速缓存行中的所讨论的数据项中来完成,其中该数据项接着被标志为初始化(有效)。另选地,用于算术运算的标识值可以被写入到临时高速缓存行中的未初始化(无效)数据项中,并且接着是利用如上所述的标识值执行的原子操作(并且该数据项然后再次被标志为初始化(有效))。

[0084] 若希望的话,可以使用用于将用于第一原子操作(用于第一线程)的运算对象值存储在临时高速缓存行中的恰当数据项中的其它安排。例如,若希望的话,可以使用将运算对象值传递至临时高速缓存行中的数据项的“移动”或“绕过”操作或指令。一般来说,可以使用可将用于第一原子操作的运算对象值存储在临时高速缓存行中的恰当数据项中的任何操作。

[0085] 对于要在临时高速缓存行上执行的任何随后的原子操作来说,优选地,将所讨论的存储器位置的当前内容(数据项)装入临时高速缓存行中,利用用于所讨论的线程的原子操作的运算对象(例如,寄存器)值和存储在临时高速缓存行中的值来执行原子操作的算术运算,接着是将临时高速缓存行中的值的原子操作的结果写回至临时高速缓存行中的恰当数据项(位置)。

[0086] 由此,在优选实施方式中,在存在具有要针对临时高速缓存行执行的原子操作的线程集合,并且已利用用于原子操作的标识值初始化所述临时高速缓存行的情况下,第一线程将利用用于原子操作的算术运算的标识值(存储在临时高速缓存行中)以及用于原子操作的第一线程的运算对象值来执行原子操作的算术运算,并将结果写回到临时高速缓存行中。接着,第二线程将利用原子操作的第二线程的运算对象值以及存储在临时高速缓存行中的值(现在将是原子操作的第一线程的运算对象值)来执行原子操作的算术运算,并将结果写回到临时高速缓存行中。接着,第三线程(若存在的话)将利用原子操作的第三线程的运算对象值以及存储在临时高速缓存行中的值(现在将是用于原子操作的第一线程和第二线程的运算对象值的组合结果)来执行原子操作的算术运算,并将结果写回到临时高速缓存行中,并且对于该组线程中的其它线程来说同样如此(若有的话),由此在临时高速缓存行中存储所述线程集合的算术运算的组合结果。

[0087] 对应地,当要针对临时高速缓存行中的“未初始化”数据项执行原子操作时,用于所讨论的原子存储器请求的运算对象值被优选地存储在临时高速缓存行中的所讨论的数

据项中,接着将该数据项标志为初始化(有效)。针对临时高速缓存行中的数据项(即,针对“初始化”(“有效”)数据项)的随后原子操作接着可以利用存储在临时高速缓存行中的值按正常方式来执行。

[0088] 由此,在优选实施方式中,在存在具有要针对临时高速缓存行执行的原子操作的线程集合并且已经利用未初始化(无效)位准备了临时高速缓存行的情况下,第一线程将有效地将其用于原子存储器请求的运算对象值写入临时高速缓存行中的所讨论的数据项中,并将该数据项标志为初始化(有效),接着,第二线程将利用针对原子操作的第二线程的运算对象值以及存储在临时高速缓存行中的值(现在将是用于原子操作的第一线程的运算对象值)来执行原子操作的算术运算,并将结果写回到临时高速缓存行中。接着,第三线程(若存在的话)将利用原子操作的第三线程的运算对象值以及存储在临时高速缓存行中的值(现在将是用于原子操作的第一线程和第二线程的运算对象值的组合结果)来执行原子操作的算术运算,并将结果写回到临时高速缓存行中,并且对于线程集合中的其它线程来说同样如此(若有的话),由此,在临时高速缓存行中存储线程集合的算术运算的组合结果。

[0089] 一旦包含要被执行原子操作的存储器地址处的数据的高速缓存行抵达数据高速缓存,则通过使用高速缓存行中的相应的数据项对(涉及所讨论的存储器地址)作为运算对象来执行原子操作的算术运算(例如,通过利用相应的数据项对(每一个高速缓存行一个数据项)执行原子加,其中,该原子操作是原子加),将高速缓存行与临时高速缓存行相组合,以提供“结果”高速缓存行。

[0090] 在优选实施方式中,针对那些数据项,该组合操作还确保“结果”高速缓存行中的任何其它数据项包含从存储器装入的值(即,之前存在于“真实”高速缓存行中)。

[0091] 由此,例如,在通过将每一个数据项设置成用于算术运算的标识值来准备临时高速缓存行的情况下,当包含要被执行所述原子操作的存储器地址处的数据的高速缓存行抵达数据高速缓存时,优选地通过针对高速缓存行中的每一个相应的数据项对执行原子操作的算术运算(例如,通过针对相应的数据项对(每一个高速缓存行一个数据项)执行原子加,其中,该原子操作是原子加),将高速缓存行与临时高速缓存行相组合,以提供“结果”高速缓存行。

[0092] 对应地,在“未初始化”位被用于准备临时高速缓存行的情况下,处理优选地包括针对临时高速缓存中的任何“未初始化”数据项,将传入高速缓存行中的数据项复制至未改变的“结果”高速缓存行(即,临时高速缓存行中的任何未初始化数据项使该组合操作正好使用来自未修改的传入高速缓存行的对应数据)。

[0093] 该组合的结果高速缓存行应当被写入到数据高速缓存中。可以为此目的分配新的高速缓存行,但在优选实施方式中,至少针对不返回型原子,使用现有的临时高速缓存行存储该结果高速缓存行。一旦生成了该结果高速缓存行,其就可以(并且优选地)按正常方式立即释放以供使用(例如,由另一数据高速缓存使用)(即,其不被和/或不再被标记为“临时”高速缓存行)。

[0094] 利用原子操作组合高速缓存行的上述操作将为原子操作集合在存储器位置中提供恰当的结果值。由此,对于“不返回”原子来说,该处理可以在该点停止。

[0095] 然而,在原子是“返回”变型的情况下,每一个单独线程还必需在执行该单独线程的原子操作之前返回存在于该存储器位置处的数据。为实现这点,并且按上述方式组合高

速缓存行以提供存储器地址的最终结果,还必需确保每一个单独线程返回其恰当值。

[0096] 为促进该操作,在优选实施方式中,至少针对返回型原子操作,当原子操作(针对一线程)针对临时高速缓存行执行时,该线程的原子存储器操作未被返回至启动器,而相反被延迟,直到完成了该存储器地址的临时高速缓存行和装入高速缓存行的组合操作为止。要做到这一点,原子存储器操作被优选地放置在用于延迟的存储器请求的缓冲器中。存储在缓冲器中的每一个原子存储器操作已关联了其来自的线程的标识(这样,该原子操作的结果可以返回至正确位置)、所讨论的原子操作以及针对(需要用于)该原子操作的任何变元。对应地,用于延迟的存储器请求的缓冲器中的每一个条目都优选地具有存储器地址、数据项以及与其相关联的存储器请求所需的任何其它元数据。

[0097] 在一个优选实施方式中,在将原子存储器请求放入“延迟的请求”缓冲器中时,所述请求保持其原始变元值。在另一优选实施方式中,其利用从临时高速缓存行读取的数据值来替换其变元值。在优选实施方式中,可以基于所讨论的原子操作的类型来选择使用这些方法中的哪一个。

[0098] 所述原子存储器请求因而被优选地标志为“延迟”。

[0099] 接着,一旦组合了临时高速缓存行和传入的“真实”高速缓存行来提供结果高速缓存行,就接着优选地利用来自传入高速缓存行的存储器地址中的数据来执行与所讨论的存储器地址相关的任何延迟的存储器请求。要做到这一点,优选地依次从缓冲器读出并执行恰当的延迟存储器请求。

[0100] 为促进该操作,在执行延迟的原子操作的同时,传入高速缓存行中的数据优选地被保持在数据高速缓存中。这可以如所希望地,例如通过将传入高速缓存行的数据复制至另一高速缓存行(其接着被优选地标记为临时高速缓存行)来实现。在优选实施方式中,一旦完成了组合操作,就可以优选地将传入高速缓存行标记为临时高速缓存行(使得在原子操作集合之前来自存储器地址的原始数据将被保持在数据高速缓存中)。这是可能的,因为新的组合结果高速缓存行现在将包含其它原子操作等可能接着需要的、所讨论的存储器地址的“真实”数据。

[0101] 所述延迟的存储器请求应当针对所讨论的原子操作按恰当方式来完成。

[0102] 由此,针对每一个延迟的原子存储器请求,被延迟的请求的存储的数据项(变元值)和临时高速缓存行的内容被优选地读取,并接着优选地使用原子算术运算以恰当地组合它们,从而将恰当的值返回至启动器。

[0103] 在这点上,当将原子操作放置在延迟请求缓冲器中时,用于执行以完成原子操作的精确动作将取决于被延迟的原子操作的原子变元值是被“保持”还是“替换”(如上所述)。

[0104] 如果该原子变元值被“替换”,则在优选实施方式中,针对每一个延迟的原子存储器请求,读取包含来自存储器的高速缓存行的先前内容的临时高速缓存行,利用原子操作的算术运算将读取的值与(延迟请求缓冲器中的)原子请求的运算对象值(数据项)组合,并接着将组合值返回至启动器。包含来自存储器的高速缓存行的先前内容的临时高速缓存行本身保持未修改。

[0105] 如果该原子变元值被“保持”,则在优选实施方式中,针对每一个延迟的原子请求,读取包含来自存储器的高速缓存行的先前内容的临时高速缓存行,针对该临时高速缓存行执行原子读取-修改-写入操作,并且将从临时高速缓存行读取的旧值返回至启动器。换句

话说,在这种情况下,被延迟的原子存储器操作优选地从高速缓存行读取值,并将读取的值返回至其启动器,接着利用读取的值执行其原子操作,并将该原子操作的结果写回至高速缓存行。

[0106] 在一优选实施方式中,该(并且每一个)数据高速缓存针对指定临时高速缓存行,跟踪哪一个延迟存储器条目被停放以等待来自存储器的数据(“真实”高速缓存行)。接着,在组合了临时和“真实”高速缓存行之后,优选地利用跟踪信息来标识要执行的相关的延迟的原子存储器请求。

[0107] 如果在延迟的原子存储器请求用从临时高速缓存行读取的数据值替换它们的变元值的方案中使用未初始化位,则从临时高速缓存行读取并在存储器请求被放置在延迟存储器请求缓冲器中时与所述存储器请求一起存储的值应当并且优选地携带有未初始化位。

[0108] 还可能必需的是,即使在不返回型原子的情况下,也在需要实施严格存储器次序的数据处理系统中执行上述延迟原子存储器请求“重放(replaying)”操作。

[0109] 在要针对将全部出现在同一高速缓存行中的位置靠近的存储器地址来执行多个原子操作的情况下,单个临时高速缓存行可以并且优选地用于累积所述多个原子存储器操作的结果。由此,在优选实施方式中,单个临时高速缓存行被用于累积多个原子存储器操作(均使用不同存储器地址)的结果。由此,在优选实施方式中,在要被执行原子操作的指定临时高速缓存行中存在多个数据项。

[0110] 还可能的是,同时分配多个临时高速缓存行并且每一个临时高速缓存行都与全部出现在同一高速缓存行中的存储器地址相关联,并接着在“真实”高速缓存行抵达数据高速缓存时单独地组合该“真实”高速缓存行与每一个相应的临时高速缓存行。

[0111] 将从以上理解的是,在执行了高速缓存行组合(合并)操作之后,来自其它数据高速缓存的监听(snoop)可以使得“真实”高速缓存行在完成延迟的存储器请求的假脱机(spooling)之前从该数据高速缓存消失。在这种情况下,如果出现针对同一存储器地址的新原子请求,则即使仍然存在同一存储器地址的旧的临时高速缓存行,也优选地为该存储器地址分配新的临时高速缓存行。在这种情况下,任何临时高速缓存行命中(hit)应当并且优选地命中新的临时高速缓存行,而非旧的临时高速缓存行。为此,高速缓存行组合处理还优选地修改所讨论的临时高速缓存行的状态,以防止命中旧的临时高速缓存行。

[0112] 在优选实施方式中,如果尝试匹配临时高速缓存行的地址但不匹配临时高速缓存行的原子操作的存储器存取,则延迟所尝试的存取直到已完成现有的临时高速缓存行的高速缓存行合并处理为止,或者为针对该存储器地址的每一个单独原子操作分配了附加的临时高速缓存行。在后一种情况下,当外部数据(高速缓存行)从存储器抵达时,优选地执行多个高速缓存行“合并”操作,针对被绑定至该存储器地址的每一个临时高速缓存行执行一次。优选地,可以针对不同原子操作来选择使用这些方法中的哪一个。

[0113] 上面讨论了原子操作所需的数据未存在于数据高速缓存中的情况。如果原子操作所需的、来自主存储器的数据已经存在于相关的数据高速缓存中,则原子操作可以简单地按正常方式针对数据高速缓存中存储有该数据的高速缓存行执行。

[0114] 用于执行原子操作的指令将通过用于执行级的编译器而包括在用于(执行流水线的)执行级的指令流中。由此,在图形处理系统的情况下,用于图形处理流水线的着色器编译器优选地执行该操作。该编译器例如可以并且在优选实施方式中运行在包括执行流水线

的数据处理系统的主处理器上(其中,该执行流水线在另一处理器上,诸如与主处理器相关联的图形处理器(以使编译器和编译代码全部运行于总数据处理系统(例如,图形处理系统)内的分离的处理器上)。然而,其它排布结构也是可能的,诸如和编译代码运行于同一处理器上的编译器,或者运行于分离的处理器上的编译器,诸如程序在分离的系统上预先编译并且按编译的格式分发。

[0115] 在于图形处理流水线的情况下,执行单元(级)优选地包括图形处理流水线的可编程着色器级,诸如顶点着色器、片段着色器等。这些级可以如所希望地并且按任何合适方式实现,并且可以分别和在合适时候执行任何希望和合适的着色(例如,顶点着色、片段着色等)功能。在片段着色器的情况下,例如,该片段着色器可以渲染图元或多个图元以生成一组渲染输出值,例如,表示用于显示的帧。这些输出值接着可以被导出至外部存储器以供存储和使用,诸如导出到显示器的帧缓冲器。

[0116] 优选地,执行流水线的所有执行级(每一个执行级)可以按本发明的方式来操作。

[0117] 每一个可编程处理级(执行单元)可以包括任何合适的可编程硬件部件,诸如可编程处理电路。每一个可编程处理级可以作为单独的电路部件被提供给处理流水线的其它可编程级,或者所述可编程处理级可以共用一些或全部的可编程处理电路(即,被不同地编程以用作希望的可编程处理级)。

[0118] 除了可编程处理(着色器)级,图形处理流水线还可以包含任何其它合适和希望的处理级,图形处理流水线可以包含诸如光栅化器(rasteriser)、早期深度(或早期深度和模板)测试器、后期深度(或深度和模板)测试器、混合器(blender)、图块(tile)缓冲器、写出单元等。

[0119] 本发明可以被用于图形处理流水线可被用于生成的所有形式的输出,诸如用于显示的帧、渲染-至-纹理输出等。来自图形处理的输出(例如,片段着色的)数据值优选地导出至外部(例如,主)存储器,以供存储和使用,如用于显示器的帧缓冲器。

[0120] 在一些实施方式中,图形处理流水线包括存储在此描述的数据和/或存储用于执行在此描述的处理的软件的一个或更多个存储器和/或存储器装置,和/或与一个或更多个存储器和/或存储器装置通信。该图形处理流水线还可以与主微处理器和/或与用于基于由图形处理器生成的数据来显示图形的显示器进行通信。

[0121] 本发明可应用于图形处理器的任何合适形式或配置。本发明特别可应用于基于图块的图形处理器和图形处理系统。由此,在优选实施方式中,该图形处理系统和图形处理流水线分别是基于图块的系统和流水线。

[0122] 在特别优选的实施方式中,本发明的各种功能在单个图形处理平台上执行,所述平台生成并输出所渲染的片段数据(所述数据例如被写入到用于显示装置的帧缓冲器)。

[0123] 本发明可以按任何合适系统来实现,诸如适当配置的基于微处理器的系统。在优选实施方式中,本发明在计算机和/或基于微处理器的系统中实现。

[0124] 本发明的各种功能可以按任何希望和合适方式来执行。例如,本发明的功能可以如所希望地按硬件或软件来实现。由此,例如,除非另外指出,本发明的各种功能性部件和“装置”可以包括在工作上能够执行各种功能等的合适的处理器或多个处理器、控制器或多个控制器、功能性单元、电路、处理逻辑、微处理器排布结构等,诸如可以被编程成为按希望方式操作的适当专用的硬件部件和/或可编程硬件部件。

[0125] 在此,还应注意到,如本领域技术人员应当理解的,本发明的各种功能等可以在指定处理器上复制和/或并行执行。等地,若希望的话,各种处理级可以共用处理电路等。

[0126] 针对必需执行上述特定功能的任何硬件的主旨,该数据处理系统和流水线可以另外包括数据处理流水线所包括的常见功能单元等中的任一个或更多个或全部。

[0127] 本领域技术人员还应清楚,本发明的全部描述方面和实施方式在合适时候可以并优选地包括在此描述的优选和可选特征中的任一个或更多个或全部。

[0128] 根据本发明的方法可以至少部分地利用软件(例如,计算机程序)来实现。由此,可以看到,当从另一方面观看时,本发明提供在被安装在数据处理装置上时特别适于执行在此描述的方法的计算机软件,包括当在数据处理装置上运行该程序部件时用于执行在此所描述的方法的计算机软件代码的计算机程序部件以及包括当在数据处理系统上运行该程序时适于执行在此描述的方法的或多个方法的全部步骤的代码手段的计算机程序。该数据处理装置可以是微处理器系统、可编程FPGA(现场可编程门阵列)等。

[0129] 本发明还扩展至包括这种软件的计算机软件载体,所述软件在被用于操作包括数据处理装置的图形处理器、渲染器或微处理器系统时,与所述数据处理装置结合地使所述处理器、渲染器或系统执行本发明的方法的步骤。这种计算机软件载体可以是物理存储介质,诸如ROM芯片、CD ROM、RAM、闪速存储器或磁盘,或者可以是诸如导线传输电子信号的信号、光学信号或诸如卫星的无线电信号等。

[0130] 还应清楚,不是本发明方法的所有步骤都需要通过计算机软件来执行,并由此,根据本发明的更广泛方面提供计算机软件,这种计算机软件安装在计算机软件载体上以用于执行在此阐述的方法的步骤中的至少一个步骤。

[0131] 因此,本发明的装置可以适当地具体实施为与计算机系统一起使用的计算机程序产品。这种实现可以包括一系列计算机可读指令,所述指令被固定在诸如计算机可读介质(例如,软盘、CD-ROM、ROM、RAM、闪速存储器,或硬盘)的有形非暂时介质上。还可以包括可通过有形介质(包括但不限于光学或模拟通信线路)或利用无线技术(包括但不限于微波、红外线或其他传输技术)无形地经由调制解调器或其它接口装置传送至计算机系统的一系列计算机可读指令。该系列计算机可读指令具体实施先前在此描述的全部或部分功能。

[0132] 本领域技术人员应当清楚,这种计算机可读指令可以采用许多编程语言来编写,以与许多计算机架构或操作系统一起使用。而且,这种指令可以利用当前或将来的任何存储器技术来存储,包括但不限于半导体、磁性光学,或者利用当前或将来的任何通信技术来传送,包括但不限于光、红外线或微波。设想的是,这种计算机程序产品可以作为具有附随打印或电子文档(例如,收缩-包装软件)的可去除介质来分发,利用计算机系统预先-加载(例如,预先加载在系统ROM或固定盘上),或者通过网络(例如,因特网或万维网)从服务器或电子公告板分发。

附图说明

[0133] 下面,仅通过示例的方式并且参照附图对本发明的许多优选实施方式进行描述,其中:

[0134] 图1示出了具有多个数据高速缓存的示例性数据处理系统;

[0135] 图2A和图2B示出了在具有多个数据高速缓存的数据处理系统中的常规原子操作

处理；

[0136] 图3示出了具有多个数据高速缓存的示例性计算机图形处理系统；

[0137] 图4示意性地示出了图3的计算机图形处理系统的图形处理单元，并且该图形处理单元可以按本发明的方式操作；以及

[0138] 图5A、图5B和图5C以及图6A、图6B和图6C例示了本发明的优选实施方式的操作。

[0139] 相同标号在合适情况下被用于图中的相同组件。

具体实施方式

[0140] 下面，将在处理用于显示的计算机图形的背景下对本发明的许多优选实施方式进行描述。

[0141] 图3示出了示例性计算机图形处理系统，该系统包括主处理器1以及在主处理器1的控制下执行处理操作的四个关联的图形处理单元 (GPU)。每一个GPU都采用图形处理流水线的形式，并且具有用于存储来自主存储器12的数据以供用于所讨论的GPU的操作的关联1级 (L1) 高速缓存8、9、10、11。

[0142] 在主处理器1上执行的诸如游戏的应用2将需要通过关联的图形处理单元 (图形处理流水线) 4、5、6、7执行的图形处理操作。要做到这一点，该应用将生成API (应用编程接口) 调用，所述API调用被运行在主处理器1上的用于图形处理单元的驱动器3来解释，以生成针对图形处理单元的恰当命令，从而生成应用2所需要的图形输出。为此，响应于来自运行在用于图形输出 (例如，用于生成要显示的帧) 的主系统1上的应用2的命令，将一组“命令”提供给图形处理单元。

[0143] 图4更详细地示出了本实施方式的图形处理单元中的一个。图3所示的每一个图形处理单元都具有图4所示的配置。

[0144] 图4中示出的GPU (图形处理流水线) 是基于图块的渲染器，因此如本领域所已知的，所述GPU生成渲染输出数据阵列的图块，如要生成的输出帧。

[0145] (如本领域所已知的，在基于图块的渲染方面，不是将整个渲染输出 (例如，帧) 按即时模式有效地一次性地处理，而是将该渲染输出 (例如，要显示的帧) 划分成多个较小的子区域 (通常称为“图块”)。每一个图块 (子区域) 都单独地渲染 (通常是一个一个地)，并接着将所渲染的图块 (子区域) 重新组合，以提供完整的渲染输出 (例如，用于显示的帧)。在这种布置中，渲染输出通常被划分成规则尺寸和形状的子区域 (图块) (例如通常是方形或矩形的)，但这不是必要的)。

[0146] 如本领域所已知的，渲染输出数据可以通常是旨在用于在显示装置 (如屏幕或打印机) 上显示的输出帧，但例如也可以包括旨在供以后渲染通道 (rendering pass) 中使用的中间数据 (还已知为“渲染至纹理”输出) 等。

[0147] (如本领域所已知的，在要显示计算机图形图像时，计算机图形图像通常首先被定义为一组图元 (多边形)，所述图元接着依次被划分 (光栅化) 成用于图形渲染的图形片段。在正常的图形渲染操作期间，渲染器将修改与每一个片段相关联的 (例如) 颜色 (红、绿及蓝RGB) 和透明度 (alpha a) 数据，使得可以正确显示所述片段。一旦所述片段已经完全经过渲染器，那么就将它们的关联数据值存储在存储器中，准备好输出，例如用于显示)。

[0148] 图4示出了与本实施方式的操作相关的图形处理流水线的主要部件和流水线级。

如本领域技术人员应理解的,可以存在图4中未例示的图形处理流水线的其它部件。在此,还应注意,图4仅是示意性的,而且例如,实际上所示的功能性单元和流水线级可以共用大量的硬件电路,即使它们在图4中被示意性地示出为单独的级。还应清楚,如图4所示的图形处理流水线的每一个级、部件及单元等都可以如所希望地实现,并因此例如包括用于执行必需操作和功能的恰当电路和/或处理逻辑等。

[0149] 如图4所示,图形处理流水线3包括多个级,所述多个级包括顶点着色器20、外壳着色器21、曲面细分器(tessellator)22、域着色器23、几何着色器24、光栅化级25、早期Z(深度)和模板测试级26、采用片段着色级27形式的渲染器、后期Z(深度)和模板测试级28、混合级29、图块缓冲器30以及下采样与写出(多重采样分辨)级31。

[0150] 如本领域所已知的,顶点着色器20获取与针对要生成的输出定义的顶点等相关联的输入数据值,并且通过图形处理流水线3的随后级来处理那些数据值,以生成供使用的一组对应的“顶点着色”输出数据值。顶点着色例如修改输入的数据以考虑要渲染的图像中的光照效果。

[0151] 如本领域所已知的,外壳着色器21针对多组片(patch)控制点执行操作,并且生成已知为片常数的附加数据,曲面细化级22细分几何形状以创建外壳的较高级表述,域着色器23针对曲面细化级输出的顶点执行操作(类似于顶点着色器),并且几何着色器24处理整个图元,诸如三角形、点或线。响应于提供给图形处理流水线3的命令和顶点数据,这些级与顶点着色器21一起有效地执行所有必需的片段前端操作(诸如变换与光照操作)和图元设置以设置要渲染的图元。

[0152] 如本领域所已知地,图形处理流水线3的光栅化级25操作以将组成渲染输出(例如,要显示的图像)的图元光栅化成用于处理的单独图形片段。为此,光栅化器25接收用于渲染的图形图元,针对采样点光栅化该图元,并且生成具有用于渲染图元的恰当位置(表示恰当采样位置)的图形片段。

[0153] 通过光栅化器生成的片段接着被向前发送至流水线的其余部分以供处理。

[0154] 如本领域所已知地,早期Z/模板级26针对其从光栅化器25接收到的片段执行Z(深度)测试,以查看在该级是否可以废弃(剔除)任何片段。要做到这一点,早期Z/模板级26将从光栅化器25发出的片段(和其相关联)的深度值与已经渲染的片段的深度值(这些深度值被存储在作为图块缓冲器30的一部分的深度(Z)缓冲器中)进行比较,以确定新的片段是否会被已经渲染的片段遮蔽(或者不封闭)。同时,执行早期模板测试。

[0155] 通过片段早期Z和模板测试级26的片段接着被发送至片段着色级27。该片段着色级27针对通过早期Z和模板测试的片段执行恰当的片段处理操作,以处理该片段来生成恰当的渲染片段数据,如本领域所已知的。

[0156] 该片段处理可以包括任何合适和希望的片段着色处理(诸如针对片段执行片段着色器程序、向片段应用纹理、向片段应用雾化或其它操作等)以生成恰当的片段数据,如本领域所已知的。在本实施方式中,片段着色级27采用着色器流水线的形式(可编程片段着色器)。

[0157] 因而,存在“后期”片段Z和模板测试级28,尤其是,“后期”片段Z和模板测试级28针对着色片段执行流水线深度测试的末端,以确定是否实际上在最终图像中看到渲染的片段。该深度测试针对存储在图块缓冲器30的Z缓冲器中的片段位置使用Z缓冲器值,以如本

领域所已知的那样通过将从片段着色器27发出的片段(和其相关联)的深度值与已经渲染的片段的深度值(如存储在深度缓冲器中)进行比较来确定新片段的片段数据是否应当替换已经渲染的片段的片段数据,。该后期片段深度和模板测试级28还针对片段执行任何必需的“后期”alpha和/或模板测试。

[0158] 通过后期片段测试级28的片段接着在混合器29中经受(若需要的话)与已经存储在图块缓冲器30中的片段的任何必需的混合操作。针对片段必需的任何其它剩余操作(如抖动(dither)等)(未示出)也在该级执行。

[0159] 最后,(混合的)输出片段数据(值)被写入至图块缓冲器30,从图块缓存器30,所述片段数据(值)例如可以被输出至用于显示的帧缓冲器。输出的片段的深度值还被恰当地写入缓冲器30内的Z缓冲器。(如本领域所已知的,图块缓冲器将存储颜色和深度缓冲器,颜色和深度缓冲器针对所述缓冲器表示的每一个采样点(本质上,针对正在被处理的图块的每一个采样点)分别存储恰当的颜色等或Z值)。如本领域所已知的,这些缓冲器存储表示总的渲染输出的一部分(图块)的一批片段数据(例如,要显示的图像),并且缓冲器中的相应组的采样值对应于总渲染输出的相应像素(例如,每 2×2 组的采样值可以对应于输出像素,其中,使用4x多重采样)。

[0160] 图块缓冲器被提供为位于图形处理流水线(芯片)上的RAM(本地)的一部分。

[0161] 来自图块缓冲器30的数据被输入至下采样(多重采样分辨)写出单元31,并且从该处输出(写回)至外部存储器输出缓冲器,诸如显示装置(未示出)的帧缓冲器。(该显示装置例如可以包括具有像素阵列的显示器,诸如计算机监视器或打印机)。

[0162] 下采样与写出单元31将存储在图块缓冲器30中的片段数据下采样为用于输出缓冲器(装置)的恰当分辨率(即,使得生成与输出装置的像素相对应的一批像素数据),以生成用于输出至输出缓冲器的输出值(像素)。

[0163] 一旦渲染输出的图块已被处理,并且其输出被导出至主存储器(例如,主存储器中的帧缓冲器(未示出))以供存储,就接着处理下一图块,等等,直到处理了足够图块以生成渲染输出(例如,要显示的帧(图像))为止。接着,针对下一渲染输出(例如,帧)重复该处理。

[0164] 用于图形处理流水线3的其它方案当然是可能的。

[0165] 上面描述了图3中示出的图形处理系统的操作的某些特征。下面,对根据本发明的实施方式的、图3中示出的图形处理系统的操作的进一步特征进行描述。

[0166] 如从图4可以看出,每一个GPU(图形处理流水线)都包括多个可编程处理或“着色器”级,即,顶点着色器20、外壳着色器21、域着色器23、几何着色器24以及片段着色器27。这些可编程着色器级执行具有一个或更多个输入变量并生成多组输出变量的相应的着色器程序(由应用来提供)。要做到这一点,应用2提供利用诸如GLSL、HLSL、OpenCL等的高级着色器编程语言实现的着色器程序。接着,这些着色器程序通过着色器语言编译器翻译成用于目标图形处理单元的二进制代码。如本领域所已知的,这可以包括在编译器内创建该程序的一个或更多个中间表示。(编译器例如可以是驱动器3的一部分,并且存在使该编译器运行的专用API调用。编译器执行由此可以被看作是响应于由应用生成的API调用而由驱动器进行的绘制调用准备的一部分)。

[0167] 如上所述,图形处理流水线中的每一个着色器是这样的处理单元,即,其通过针对要生成的图形输出中的每一个图形项运行小程序来执行图形处理(在这点上,“项”通常是

顶点、片段或像素)。本实施方式涉及这样的情况,即,要由图形处理单元的着色器执行的执行线程(其中,每一个线程都对应于一个图形项)要执行原子存储器操作。

[0168] 如上所述,“原子”存储器操作是这样的操作序列:读取存储器位置、在存储器值与运算对象值(例如,寄存器)值之间执行算术运算,并接着将算术运算的结果写回至同一存储器位置。执行该序列操作以使得对于每一个观察者来说,看起来好像该序列根本未执行或者已经全部执行。其作为一个不可分割的单元加以执行,因此,叫做“原子”。作为该原子的一部分而执行的算术运算通常是既可交换又可联合的简单运算,如:整数加(Add);逐位的与(AND)、OR、XOR;最小值;最大值。

[0169] 原子存储器操作通常以两种变型出现:“Return”和“NoReturn”。“Return”变型向着色器返回在执行“原子”操作之前存在于存储器位置处的数据。“NoReturn”变型不向着色器返回任何值。

[0170] 在本实施方式中,指定图形处理单元的原子操作利用存储在用于所讨论的图形处理单元的对应的L1高速缓存中的数据(高速缓存行)来执行(针对执行)。由此,为了能够执行原子存储器操作,相关数据(高速缓存行)必须存在于所讨论的L1高速缓存中。

[0171] 每一个L1高速缓存都保持传入原子存储器请求的队列和延迟原子存储器请求的缓冲器。该队列和延迟请求缓冲器中的每一个条目都具有用于一个地址、一个数据项以及一个原子存储器请求所需的任何其它元数据的空间。

[0172] 当队列中的原子存储器操作要由图形处理单元的着色器执行时,首先检查相关数据(高速缓存行)是否存储在所讨论的L1高速缓存中。如果是,则该原子存储器操作按正常方式继续进行。

[0173] 然而,如果该数据(高速缓存行)不存在于L1高速缓存中,则向该原子存储器操作分配临时高速缓存行。该临时高速缓存行利用该原子操作的存储器地址和该原子操作本身(的类型)来标记,并且将包含与用于所讨论的原子操作的、系统中的“真实”高速缓存行一样多的数据项(数据值)。

[0174] 临时高速缓存行是占用数据高速缓存中的资源(空间)的“高速缓存行”,但仅可以被用于累积适合的原子存储器操作(即,匹配该临时高速缓存行的标记(同一原子操作,同一存储器地址(或者至少同一高速缓存行内的存储器地址)),并且不能以其它方式读、写、监听或者按任何其它方式相互作用。

[0175] 接着,通过将该临时高速缓存行中的所有数据项设置成用于所讨论的原子操作的算术运算的标识值来准备用于原子操作的临时高速缓存行。

[0176] 被用于初始地填充临时高速缓存行的算术运算的“标识值”是当该算术运算利用另一值来执行时使该另一值无变化的值。由此,如果指定算术运算用运算符%来表示,则该算术运算的标识值为值V,使得对于所有可能的输入值X来说, $V\%X=X\%V=X$ 。例如,针对下列算术运算,对应标识值V是:

[0177] 整数加: $V=0$

[0178] 逐位OR、逐位XOR: $V=0$

[0179] 逐位AND: $V=$ 全为1的位向量

[0180] 最小值: $V=$ 采用存储器值的格式的最大可表示数

[0181] 最大值: $V=$ 采用存储器值的格式的最小可表示数。

[0182] 存储器请求还被送出到存储器子系统发出,以将所讨论的存储器地址的内容(即,需要原子操作的数据)装入到L1高速缓存中。

[0183] 在准备了临时高速缓存行之后,但在来自存储器的数据(高速缓存行)抵达之前,该L1高速缓存将要使用同一存储器地址的排队的原子存储器操作累积到该临时高速缓存行中。要做到这一点,从该队列取得原子存储器请求,并且针对该临时高速缓存行执行,就好像该临时高速缓存行是正常高速缓存行。以此,将临时高速缓存行中的所讨论的存储器位置的当前内容(数据项)装入,接着是将针对临时高速缓存行中的值的原子操作的结果写回至临时高速缓存行中的恰当数据项(位置)。

[0184] 该原子存储器请求因而未被返回至做出该请求的启动器(在这种情况下,GPU的着色器),而相反,而是提交给延迟存储器请求缓冲器。在将该存储器请求放入延迟请求缓冲器中时,该存储器请求可以保持其原始变元值,或者利用从临时高速缓存行读取的数据值来替换其变元值。(一种实现可以针对不同原子操作选择这些方法中的任一个。例如,由于“保持”方法或“替换”方法可能具有不等的功耗,因而可能的是,一个方法或其它方法对于不同原子操作来说较好)。

[0185] 原子存储器请求因而被标志为“延迟”。

[0186] 接着,这对于每一个排队的原子存储器请求(若有的话,使用同一存储器位置并且针对该存储器位置执行同一算术运算)进行重复,直到包含所讨论存储器位置的“真实”数据的高速缓存行抵达该L1高速缓存为止。

[0187] 当来自存储器的“真实”数据抵达L1高速缓存(具有该存储器地址内容的高速缓存行抵达)时,该L1高速缓存停止其正常操作,而使用该原子操作的算术运算来组合该临时高速缓存行的内容和传入的高速缓存行(“合并”该临时高速缓存行与传入的高速缓存行)。

[0188] 例如,如果该原子操作是32位ADD,则在临时高速缓存行与传入的高速缓存行的每一个相应32位数据项之间执行32位ADD。

[0189] 组合(合并)的高速缓存行的结果接着被放入到正常高速缓存行中,而传入的高速缓存行(具有来自存储器的数据)被无修改地复制到临时高速缓存行中。

[0190] 接着,L1高速缓存行恢复其正常操作。由此,所得到的合并高速缓存行可以用于正常服务所有种类的存储器存取(加载、存储以及原子)并且可以响应于监听。(如果该高速缓存行在该合并过程期间被监听,则抑制监听响应,直到完成合并为止)。

[0191] 在完成了合并处理之后,完成之前放置在延迟存储器请求缓冲器中的、针对所讨论的存储器地址的延迟原子存储器请求。要做到这一点,该原子存储器请求被一次一个地假脱机输出到延迟存储器请求缓冲器。

[0192] 针对每一个停放的原子存储器请求,读取该请求的数据项和临时行的内容(其现在将延续来自存储器的高速缓存行的先前内容),并接着使用原子算术运算来组合它们。

[0193] 在这点上,当(如上所述)将原子操作“停放”在延迟请求缓冲器中时,用于执行以完成原子操作的精确动作将取决于原子操作的原子变元值是被“保持”还是“替换”。

[0194] 如果该原子变元值被“替换”,则针对每一个延迟的原子请求,读取该临时高速缓存行,并且利用用于原子操作的算术运算将该结果与原子请求的数据项进行组合,接着将组合值返回至着色器(启动器)。临时高速缓存行本身保持未修改。

[0195] 如果该原子变元值被“保持”,则针对每一个延迟的原子请求,读取临时高速缓存

行,对该临时高速缓存行执行原子读取-修改-写入操作,并且将从临时高速缓存行读取的旧值返回至着色器。

[0196] 在本实施方式中,每一个L1高速缓存都针对指定的临时高速缓存行,跟踪哪一个延迟存储器请求缓冲器条目被停放以等待来自存储器的数据(“真实”高速缓存行)。这可以如所希望地进行。这样,在“合并”了临时和“真实”高速缓存行后,相关被停放原子存储器请求可以利用跟踪信息一个接一个地假脱机输出。这有助于使该处理更有效。

[0197] 上述实现向原子操作的“返回(Return)”变型提供支持。

[0198] 对于不需要严格的存储器排序的“不返回(NoReturn)”原子(其可以在图形处理中常见)来说,可以省略在延迟存储器请求缓冲器中“停放”原子存储器请求并且在合并(组合)临时高速缓存行与具有来自存储器的值的传入高速缓存行之后完成那些请求。(换句话说,不需要严格存储器排序的不返回原子所需的全部是将原子操作累积到临时高速缓存行中,并接着在“真实”高速缓存行从存储器抵达时将那些值与“真实”高速缓存行组合(合并)(并接着提供组合(合并)值以供使用)。

[0199] 对于需要严格存储器排序的不返回原子来说,为了满足排序保证,该原子操作被优选地保持在延迟请求缓冲器中,直到临时行和装入的高速缓存行的合并(组合)完成为止(在该点,该原子操作可以被报告为完成),但不需要如返回原子所需那样对延迟存储器请求进行任何进一步处理。

[0200] 在本实施方式中,被用于针对选定(并且大部分)原子操作标记临时高速缓存行的“原子操作”不只是针对要执行的操作来区分,而是同样针对操作大小来区分。例如,32位原子ADD与64位原子ADD被视为两个不同的操作。类似地,如果底层实现区分字节顺序,则这也加以考虑:例如,32位小字节序(little-endian)原子ADD被视为与32位大字节序原子ADD不同的操作。(然而,这些区别不需要用于所有原子操作。例如,原子AND/OR/XOR不需要运算对象尺寸或字节顺序(endianness)区分)。

[0201] 如果提供“整数ADD模常数K(Integer Add Modulo a constant K)”原子操作,则出于临时高速缓存行的标记的目的,将“K”的每一个不同值视为不同的操作。类似地,如果提供“比较并交换(Compare-And-Swap)”原子操作,则出于临时高速缓存行的标记的目的,将每一个不同的比较值视为不同的操作。

[0202] 图5A、图5B和图5C以及图6A、图6B和图6C示意性地示出了针对本发明的两个实施方式的、用于指定图形处理单元的上述操作。

[0203] 图5A、图5B和图5C示意性地示出了第一实施方式中的操作,其中,每一个延迟原子操作保持其原始变元值。

[0204] 如图5A所示,所讨论的图形处理单元的L1数据高速缓存61具有用于存储要执行的传入原子存储器操作的队列60以及用于存储延迟原子存储器请求的缓冲器62。

[0205] 如图5A所示,假定原子存储器请求队列60初始地包含用于要向同一存储器地址(位置)执行原子操作(在这种情况下是原子ADD运算)的相应线程A、B、C、D的四个未决原子存储器请求。还假定数据高速缓存61当前未存储该原子存储器操作所需的、来自存储器的数据(高速缓存行)。

[0206] 有鉴于此,如图5A所示,在数据高速缓存61中分配临时高速缓存行64,并且利用排队原子操作所需的存储器地址和所讨论的原子操作的类型来进行标记。

[0207] 该临时高速缓存行中的数据项接着被初始化成用于所讨论的原子操作的标识值(由于原子操作是原子ADD,因此在这种情况下是“0”)。同时,向存储器子系统发出存储器请求57,以装入存储器中包含所讨论的存储器地址处的数据的高速缓存行,并且在缓冲器61中分配用于该高速缓存行的空间58。(在此假定(并且在图6A、图6B、图6C所示实施方式中)数据高速缓存61中的每一个高速缓存行仅包含一个数据项(如图5A所示))。然而,每一个高速缓存行可以包含超过一个数据项。在该情况下,临时高速缓存行中的每一个数据项都被初始化成用于所讨论的原子操作的标识值)。

[0208] 如图5A所示,在图形处理单元等待来自主存储器中的存储器地址的数据抵达的同时,针对数据高速缓存61中的临时行64执行排队的原子存储器操作,就好像临时行64是正常高速缓存行一样,以将原子存储器操作的结果累积到该临时行64中。

[0209] 因此,如图5A所示,针对数据高速缓存61中的临时行64来执行第一线程A 67的原子ADD操作。

[0210] 为执行原子操作,将与数据高速缓存61中的临时高速缓存行64中的所讨论特定存储器位置相对应的数据项的内容(数据值)装入,利用该数据值来执行原子操作的算术运算,并接着将该原子操作的结果写回至临时高速缓存行中的数据项(存储器位置)(即,临时高速缓存行本身被修改)。

[0211] 因此,线程A利用临时高速缓存行64中的所存储的标识值(“0”)来执行其原子ADD操作,并将该操作的结果(“1”)存储在临时高速缓存行中。因此,在线程A的原子操作之后,临时行中的相关数据项现在将具有如图5A所示的值“1”。

[0212] 一旦线程A执行了其原子操作,则线程A的存储器请求就不返回至启动器(着色器),相反,如图5A所示,其被放置在延迟存储器请求缓冲器62中。在图5A、图5B及图5C所示的实施方式中,假定延迟的存储器请求将如图5A所示那样保持它们的原始变元值(由此,对于线程A“+1”)。该存储器请求还被标志为延迟。

[0213] 一旦线程A针对该临时高速缓存行执行了其原子操作,则队列中的下一线程(线程B)就可以针对临时高速缓存行64执行其原子操作。如图5A所示,要做到这一点,线程B读取临时高速缓存行的存储器位置中的、高速缓存的值(现为“1”),针对临时行中的该存储器位置中的数据值执行原子ADD操作(+2),并将其原子操作的结果(“3”)写回至临时高速缓存行中的存储器位置。由此,如图5A所示,临时高速缓存行64中的存储器位置现在包含值“3”。

[0214] 同样,线程B的原子操作被放置在延迟存储器请求缓冲器62中,在这种情况下,保持其原始变元值“+2”。

[0215] 如图5A所示,该过程对于队列60中的存取同一存储器地址的其余原子操作继续进行,使得一旦针对该临时高速缓存行执行了队列60中的所有恰当原子操作,数据高速缓存61中的临时高速缓存行64在这种情况下,将存储针对共用同一存储器地址的列队原子存储器请求的累积(组合)递增值11。对应地,延迟存储器请求缓冲器62将包含之前处于队列60中但现在已经针对数据格式61中的临时高速缓存行64执行的原子存储器请求。

[0216] 该操作接着等待,直到具有期望的存储器内容(即,所讨论存储器地址)的高速缓存行抵达数据高速缓存61为止。

[0217] 图5B示出了当具有主存储器内容的高速缓存行65抵达数据高速缓存61时的操作。如图5B所示,在该点,数据高速缓存61将存储临时高速缓存行64和来自存储器的“真实”高

速缓存行65。

[0218] 接着,如图5B所示,所讨论的原子操作的算术运算被用于将临时高速缓存行64的内容与传入的“主存储器”高速缓存行65的内容组合。由此,在这种情况下,由于原子操作是ADD运算,因而在临时高速缓存行64的每一个数据元与传入的“真实”高速缓存行65的每一个数据元之间执行加(add)。该操作的结果(在这种情况下,“134”)接着被存储在数据高速缓存61中的正常高速缓存行66中,同时将来自主存储器的传入速缓存行(传入数据)无修改地复制到临时高速缓存64中。

[0219] 因而,如图5B所示,该结果是数据高速缓存61将包含:“正常”高速缓存行66,其包含原子操作的累积结果(在这种情况下,“134”);以及临时高速缓存行64,其包含来自主存储器的原始值(在这种情况下,“123”)。

[0220] 包含累积原子操作的结果的“正常”高速缓存行66因而可以在合适时候被释放以供其它图形处理单元、原子操作等使用,或者返回至主存储器。

[0221] 在这之后,如图5B所示,数据高速缓存61将包含来自存储器中的原始高速缓存行的数据的临时高速缓存行64(即,在执行原子操作之前)。

[0222] 该过程完成针对该组排队的原子操作的组合原子操作。然而,至少在返回原子操作的情况下,还必需恰当地执行单独的原子操作。如图5C所示,针对本实施方式执行这个过程。

[0223] 如图5C所示,一旦数据高速缓存61在临时高速缓存行64中存储原始存储器值(即,在排队的原子操作之前的存储器中的值),则就可以利用临时高速缓存行中的数据来完成延迟存储器请求缓冲器62中停放的原子操作。这是这样完成的:通过将停放的原子操作一次一个地假脱机输出延迟存储器请求缓冲器,并且针对每一个操作,读取原子操作的数据项和临时高速缓存行的内容,并接着利用原子操作的算术运算将它们组合以提供原子操作返回值(接着被返回至着色器(启动器))。

[0224] 在这个实施方式中,因为每一个停放的原子操作都保持其原始变元值,所以针对每一个延迟的原子请求,该操作包括读取临时高速缓存行,对该临时高速缓存行执行原子读取-修改-写入操作以更新该临时高速缓存行中的值,并且将从临时高速缓存行读取的旧值返回着色器(根据返回原子操作)。

[0225] 由此,如图5C所示,线程A将利用临时高速缓存行中的值(“123”)来执行其原子加(“+1”)操作,将恰当地修改的值(在这种情况下,“124”)写入临时高速缓存行64中,并且返回(69)临时高速缓存行的原始值(在这种情况下,“123”)。接着,线程B将对该临时高速缓存行执行其原子加操作,由此针对延迟存储器请求缓冲器62中的每一个停放的原子操作,将临时高速缓存行中的值修改成“126”,并且返回临时高速缓存行中的之前的值(即,“124”),等等,如图5C所示。

[0226] 一旦完成了所有停放的原子操作,就可以释放数据高速缓存61中的临时行,如图5C所示。

[0227] 应当清楚,图5C所示的方案需要原子操作的“返回(Return)”变型。对于“不返回”原子的情况来说,可以省略图5C所示的针对临时高速缓存行执行每一个停放的原子操作并且返回数据值的操作,如在不需要严格存储器排序的不返回原子的情况下可以将原子请求停放在延迟存储器请求缓冲器62中。

[0228] 图6A、图6B及图6C示出了图5A、图5B及图5C所示操作的另选实施方式。

[0229] 一般来说,由此,图6A、图6B及图6C所示操作对应于图5A、图5B及图5C所示操作(同样地,未再次加以描述)。然而,如图6A所示,不是停放在延迟存储器请求缓冲器62中的每一个原子存储器请求都保持其原始变元值,而是利用从临时高速缓存行读取的数据值来替换该原子请求的变元值。

[0230] 由此,如图6A所示,例如,针对线程A 70存储变元值“+0”而不是“+1”。对应地,针对线程B,延迟存储器请求缓冲器62中针对该线程存储的变元值是“+1”而非“+2”71,对于线程C来说,变元值是“+3”而非“+5”,等等。

[0231] 接着,如图6C所示,当在执行了合并操作之后完成了停放的原子存储器请求时,不读取临时高速缓存行并针对每一个停放原子请求对临时高速缓存行执行原子读取-修改-写入操作,而是相反地读取临时行的值并且与存储在延迟存储器请求缓冲器62中的原子请求的变元值组合,,并且接着返回该组合值作为用于原子操作的返回值,但临时高速缓存行本身保持未修改。

[0232] 此外,对于不返回原子来说,在不需要严格的存储器排序延迟存储器请求缓冲器中的原子存储器请求的停放的不返回原子的情况下,可以省略图6C所示操作。

[0233] 如希望的话,针对上述实施方式的各种另选和添加都是可能的。

[0234] 例如,上述实施方式例示了本发明的操作,其中,在执行原子操作所应当针对的每一个高速缓存行中存在单个数据值。然而,在要被执行原子操作的指定高速缓存行中存在多个数据值(数据项、存储器地址)的情况下,可以对应地使用这些方案。

[0235] 例如,尽管上面的实施方式已经参照原子加(add)操作进行了描述,但上面的方案可以相应地用于其它原子操作。由此,其它原子操作可以被实现为上述处理的恰当变型。

[0236] 本实施方式可以应用至的原子操作的示例(和它们的对应标识值V(若有的话))为:

[0237] 整数加:V=0

[0238] 逐位OR、逐位XOR:V=0

[0239] 逐位AND:V=全为1的位向量

[0240] 最小值:V=采用存储器值的格式的最大可表示数

[0241] 最大值:V=采用存储器值的格式的最小可表示数

[0242] 饱和和无符号整数加:V=0

[0243] 整数乘:V=1

[0244] 浮点加:V=-0.0

[0245] 浮点乘:V=1.0

[0246] 整数加,模常数K:V=0(该操作的变型可在某些GPU中出现,采用原子递增/递减指令的形式,其中,“寄存器”输入是常数1,而值K被提供为实际变元)。

[0247] 而且,在上述实施方式中,使用原子操作的算术运算的标识值来初始化临时高速缓存行中的数据项。然而,不能以这种方式处理不具有标识值的原子算术运算,即使它们可交换和联合。(例如,浮点min/max可交换和联合,然而,如果希望静止存储器中的sNaN表述,则其不具有标识值)。

[0248] 因此,在另一实施方式中,在准备临时高速缓存行时在临时高速缓存行中每数据

项地存储“未初始化”位(“无效”位),以代替标识值。(若希望的话,即使算术运算具有标识值,也可以进行)。这样做的代价是额外的存储,但利用未初始化位的优点在于未初始化位使得能够按本发明的方式来处理不具有标识值的原子操作(例如,浮点min/max和原子交换)。

[0249] 在要针对临时高速缓存行中的“未初始化”数据项执行原子操作时,所讨论的原子存储器请求的变元值被存储在临时高速缓存行中的数据项中,该数据项接着被标志为初始化。利用存储在临时高速缓存行中的值,可以按正常方式来执行针对该临时高速缓存行中的数据项(即,针对“初始化”(“有效”)数据项)的随后原子操作。

[0250] 接着,在临时高速缓存行要与来自存储器的传入“真实”高速缓存行组合时(即,在执行合并操作时),针对临时高速缓存中的任何“未初始化”数据项,将传入的高速缓存行中的数据项不改变地复制到“结果”高速缓存行(即,临时高速缓存行中的任何未初始化数据项使该合并操作不修改地使用来自传入高速缓存行的对应数据(即,将传入高速缓存行中的数据项不修改地存储在临时高速缓存行中))。

[0251] 如果在延迟的原子存储器请求用从临时高速缓存行读取的数据值来替换它们的变元值的方案中使用未初始化位,则从临时高速缓存行读取并在且存储器请求被放置在被延迟存储器请求缓冲器中时与存储器请求一起存储的值应当携带未初始化位。

[0252] 如果所有原子操作都具有底层存储单位(如字节)的多倍的尺寸,则可以每存储单位而不是每数据项地提供未初始化位。只要针对高速缓存行中的每一个数据项存在至少一个这种位,这就有效。在这种情况下,如果临时高速缓存行被存储在已经能够例如跟踪每字节页面重写标志位的高速缓存中,则该页面重写标志位可以用于提供未初始化位功能。

[0253] 从上述内容可以理解,在执行了合并操作之后,来自其它L1高速缓存的监听可以造成“真实”高速缓存行在完成假脱机所停放的存储器请求之前从该L1高速缓存消失(即,使得旧的临时高速缓存行仍存在于该高速缓存中)。在这种情况下,如果出现针对同一存储器地址的新原子请求,则即使仍然存在针对同一存储器地址的旧的临时高速缓存行,也优选地为该存储器地址分配新的临时高速缓存行。在这种情况下,任何临时高速缓存行命中应当并且优选地命中新的临时高速缓存行,而非旧的临时高速缓存行。为此,合并处理还优选地修改所讨论的临时高速缓存行的高速缓存行状态,以防止命中旧的临时高速缓存行。

[0254] 在优选的实施方式中,高速缓存行监听处理是循环法(即,如果高速缓存N保持该行,并且所有其它高速缓存行想要该行,则将该高速缓存行交给高速缓存N+1),或者说是公平的。(然而,这不是必要的,并且即使无法满足这种公平性保证,也可以实现实质的性能改进)。

[0255] 使用存储器地址和原子操作二者来标记上述实施方式中的临时高速缓存行。同样地,如果尝试匹配临时高速缓存行的地址但不匹配临时高速缓存行的原子操作的存储器存取,则这种存储器存取无法累积到临时高速缓存行中。如果这种情况出现,则存在两个可能方法:延迟所尝试的存取,直到针对现有临时高速缓存行完成了高速缓存行合并处理为止;或者为针对该存储器地址的每一个分离原子操作分配附加的临时高速缓存行。在后一情况下,当外部数据(高速缓存行)从存储器抵达时,接着可以执行多个高速缓存行“合并”操作,针对绑定至该存储器地址的每一个临时高速缓存行执行一次。优选地,一种实现可以针对不同原子操作选择这些方法中的哪一个。例如,分配附加的临时高速缓存行还可能特别有

益于原子min/max操作,但对于比较并交换来说不是很好。

[0256] 可以看出,在上述实施方式中,L1高速缓存实际上仅需要在高速缓存行合并处理期间保持对高速缓存行的所有权:在完成合并处理后,该结果合并高速缓存行就可以被立即交给另一个L1高速缓存。即使延迟的原子操作尚未被完全处理,也是这种情况(因为在合并处理之后,即使没有结果合并高速缓存行,L1高速缓存行也包含为重建正确的原子返回值所需的全部信息)。

[0257] 而且,从任何观察者的观点来看,原子操作表现得好像在合并阶段提交了高速缓存行中的全部原子操作。(为此,即使它们具有“不返回(NoReturn)”类型,已经针对临时高速缓存行提交了它们的原子的原子请求在L1高速缓存中也被抑制,直到完成了合并之后为止(否则,它们的排序将变得在观察上不连续))。

[0258] 这是真实的,而与已经针对临时高速缓存行执行的原子操作的数量无关。而且,该合并阶段花费了恒定的最坏情况下的时间(其运行时间与高速缓存行长度成比例)。

[0259] 这是这样的情况,如果大量的L1高速缓存全部希望针对同一存储器位置执行原子操作,则该高速缓存行恒定地从一个L1高速缓存向另一高速缓存传递,并且每一个L1高速缓存在将该行传递至下一高速缓存之前都刚刚经受单个“合并”。这有助于将全局观测到的原子更新速率与L1高速缓存中内部的原子更新速率分离。因而,更新速率仅受限于高速缓存行可以多快地传递。

[0260] 这因而使得能够实现原子操作的极高的观察速率,即使单个高速缓存无法维持非常高的原子速率也同样如此。例如,在具有40个L1高速缓存的系统中,利用在20个周期内从一个高速缓存向另一高速缓存传递高速缓存行的能力和在5个周期内执行“合并”的能力,高速缓存行可以按每25个周期一个L1高速缓存的速率从一个L1高速缓存向另一L1高速缓存传递。如果假定每一个L1高速缓存服务1000个线程,全部线程希望递增单个计数器,则在这种系统中,每一个L1高速缓存将跟踪存储有1000个计数器增量的累积结果的临时高速缓存行。在包含该计数器的高速缓存行从一个高速缓存向另一高速缓存跳转时,对于所观察地的每时钟周期40个增量的全局递增速率来说,能够每25个周期服务1000个原子。(这需要每一个L1高速缓存能够内部地每时钟服务1个原子增量)。

[0261] 而且,本实施方式仅需要针对L1高速缓存的操作的局部改变。L1高速缓存用于与它们的所有者并且与存储器子系统通信的任何协议(典型地讲,用于该所有者的简单请求响应协议和用于存储器子系统的全部相关协议)未受影响。

[0262] 根据上述,可以看出,本发明至少在其优选实施方式中提供了一种用于按照更有效的方式来促进在具有多个数据高速缓存的系统中执行原子操作的机制。

[0263] 在本发明的优选实施方式中,这至少通过以下来实现:在等待来自存储器的“真实”高速缓存行抵达高速缓存之际,把将要存取同一存储器位置的原子操作的结果累积在临时高速缓存行中,接着在“真实”高速缓存行抵达高速缓存时将存储在临时高速缓存行中的原子操作的累积结果与来自存储器的“真实”高速缓存行组合。本发明使能累积存储器原子操作,以使当高速缓存行抵达指定高速缓存时,可以将整组累积的原子操作一次提交给该高速缓存行。

[0264] 至少在优选实施方式中,本发明还特征化了一种用于重建原子值以便产生在观察上与顺序执行的所有原子操作相同的行为的方案,。

[0265] 所有这些仅利用针对高速缓存行操作的改变来实现,没有针对任何其它子系统的改变。

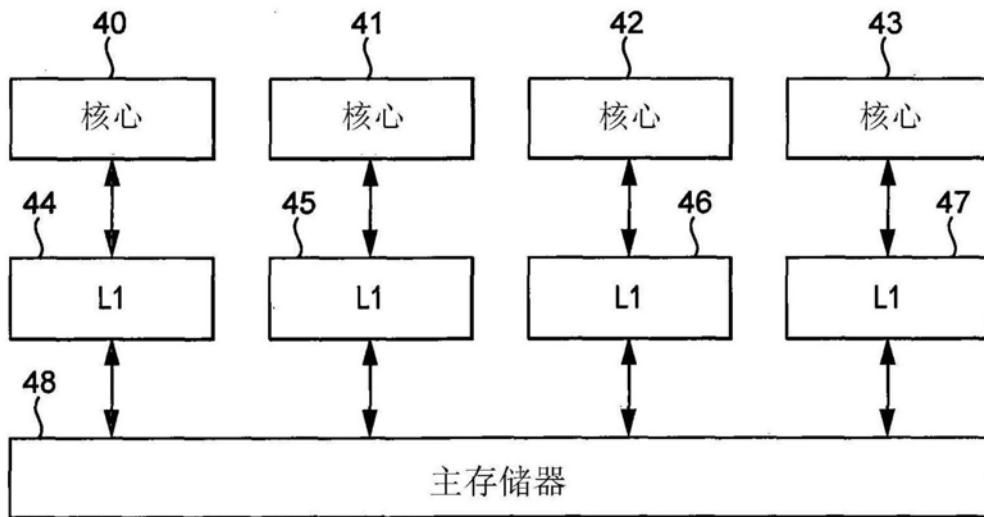


图1

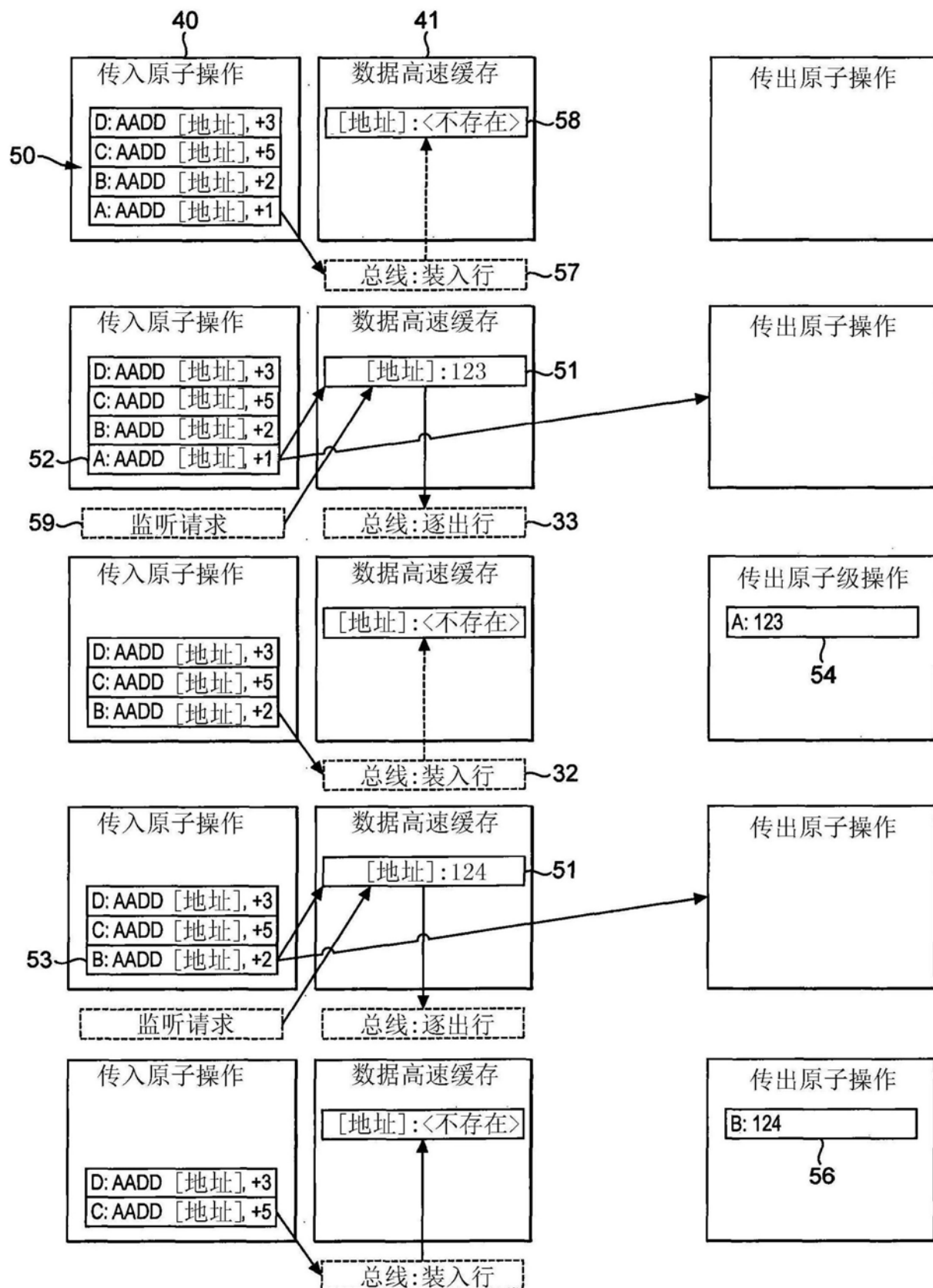


图2A

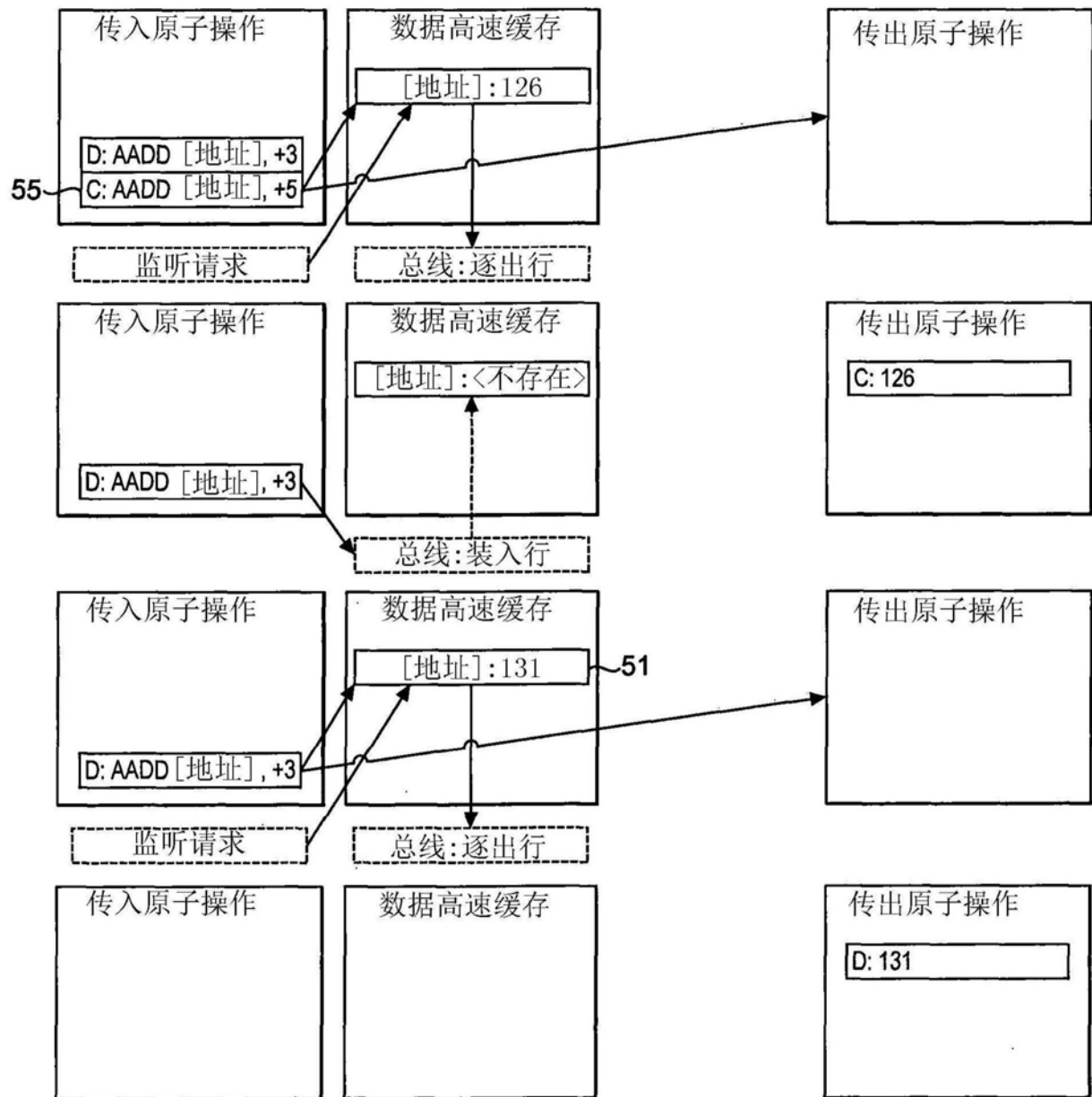


图2B

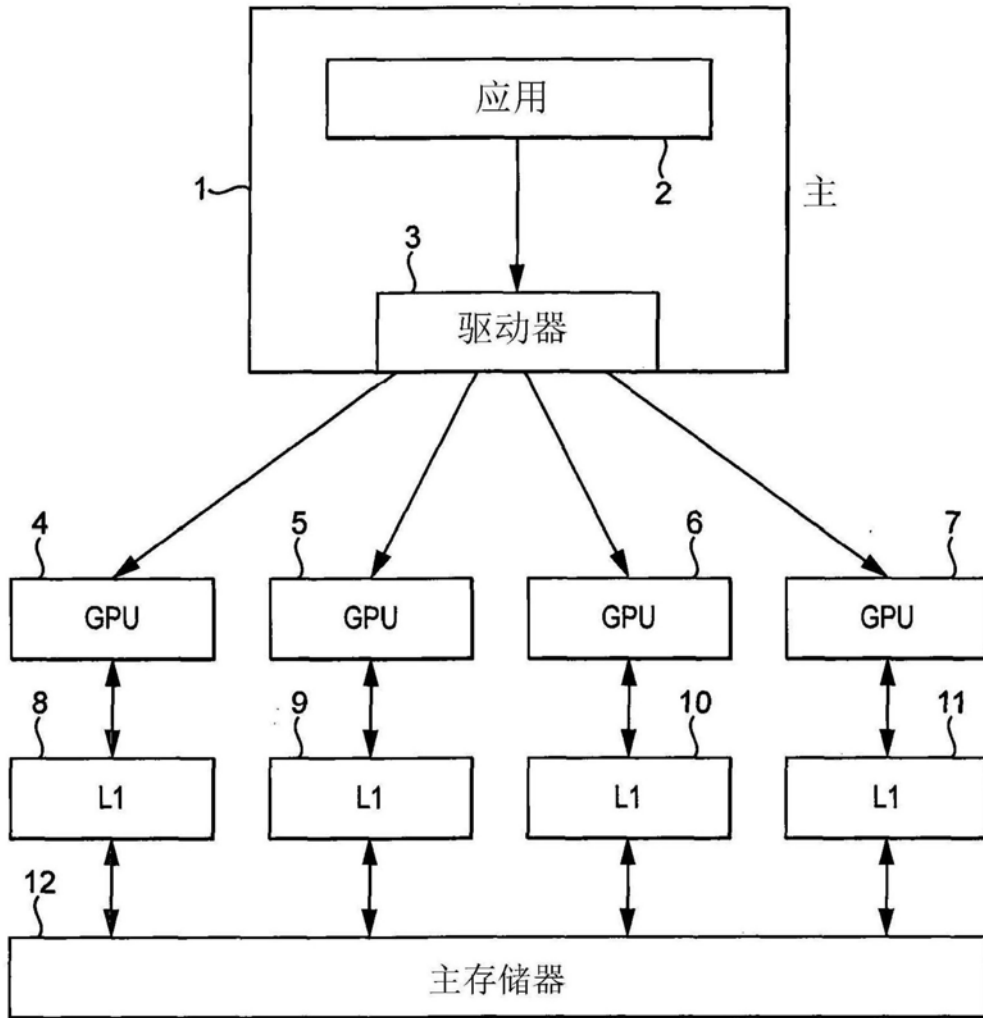


图3

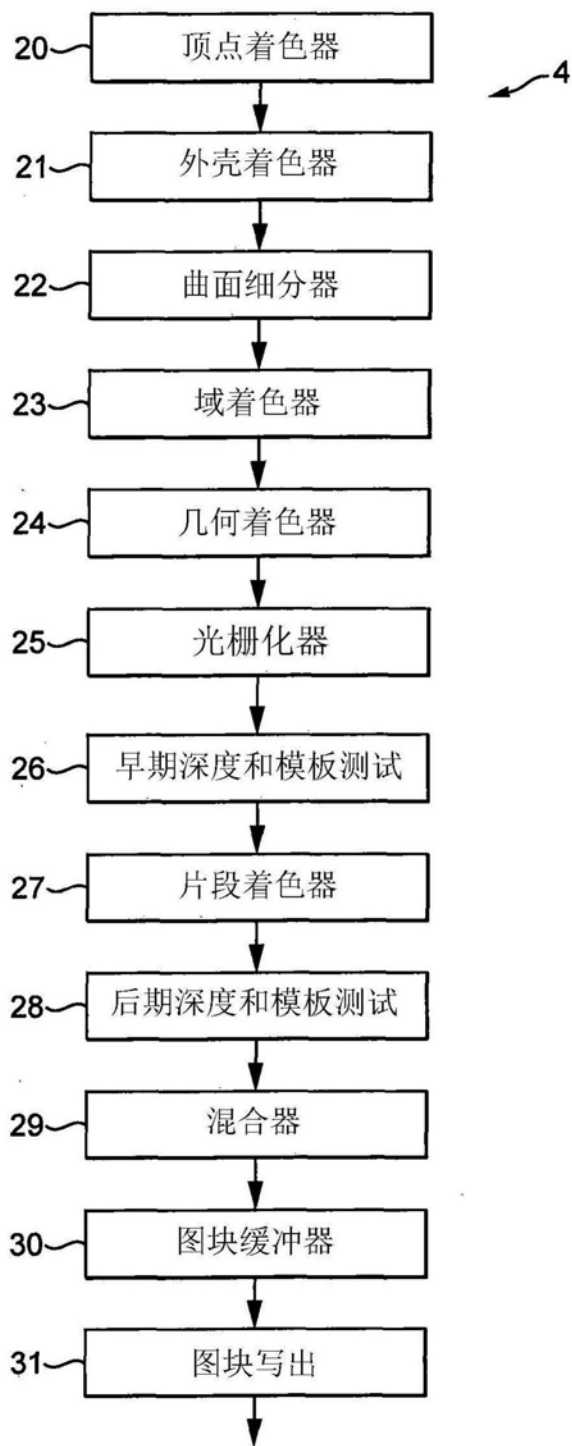


图4

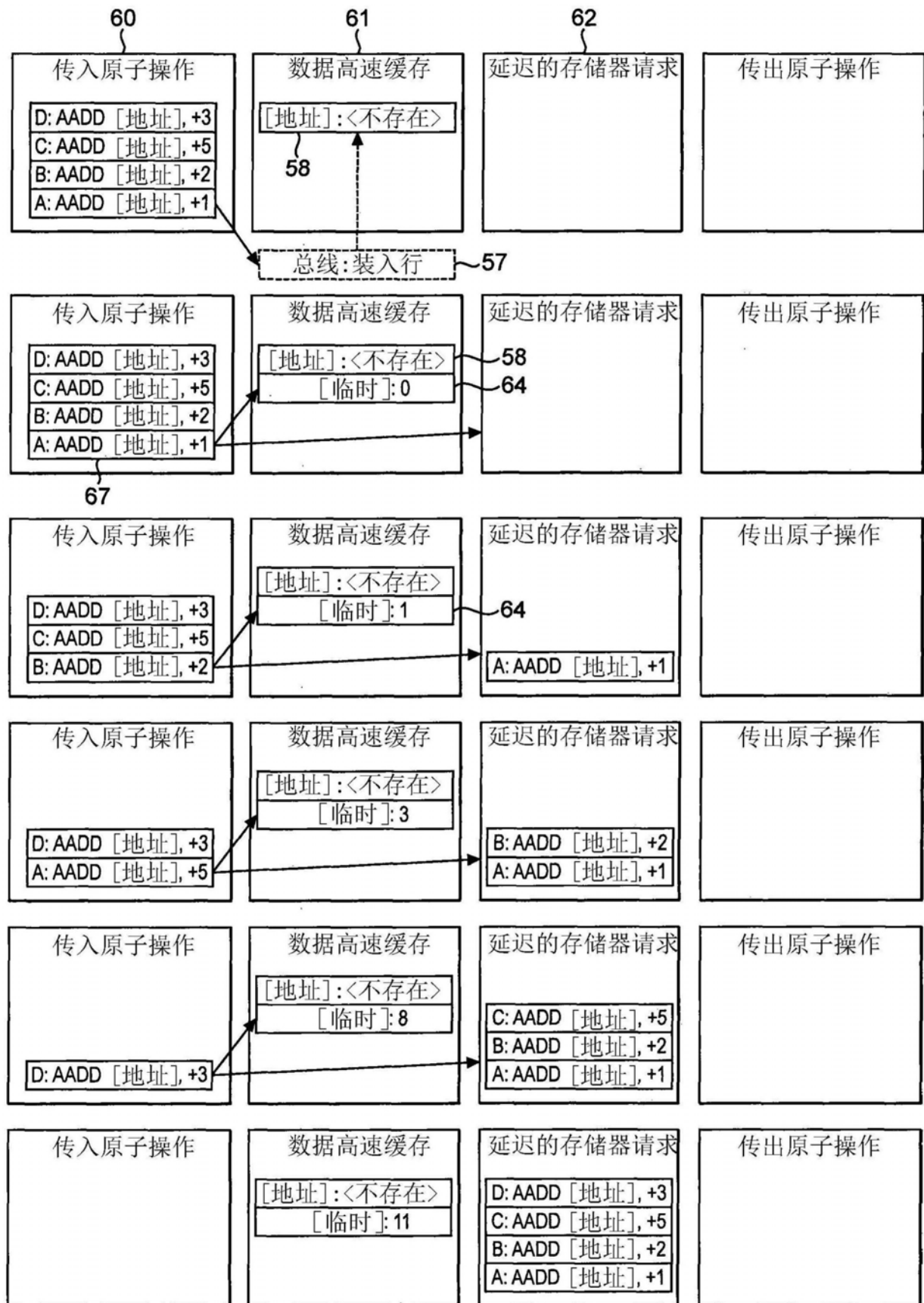


图5A

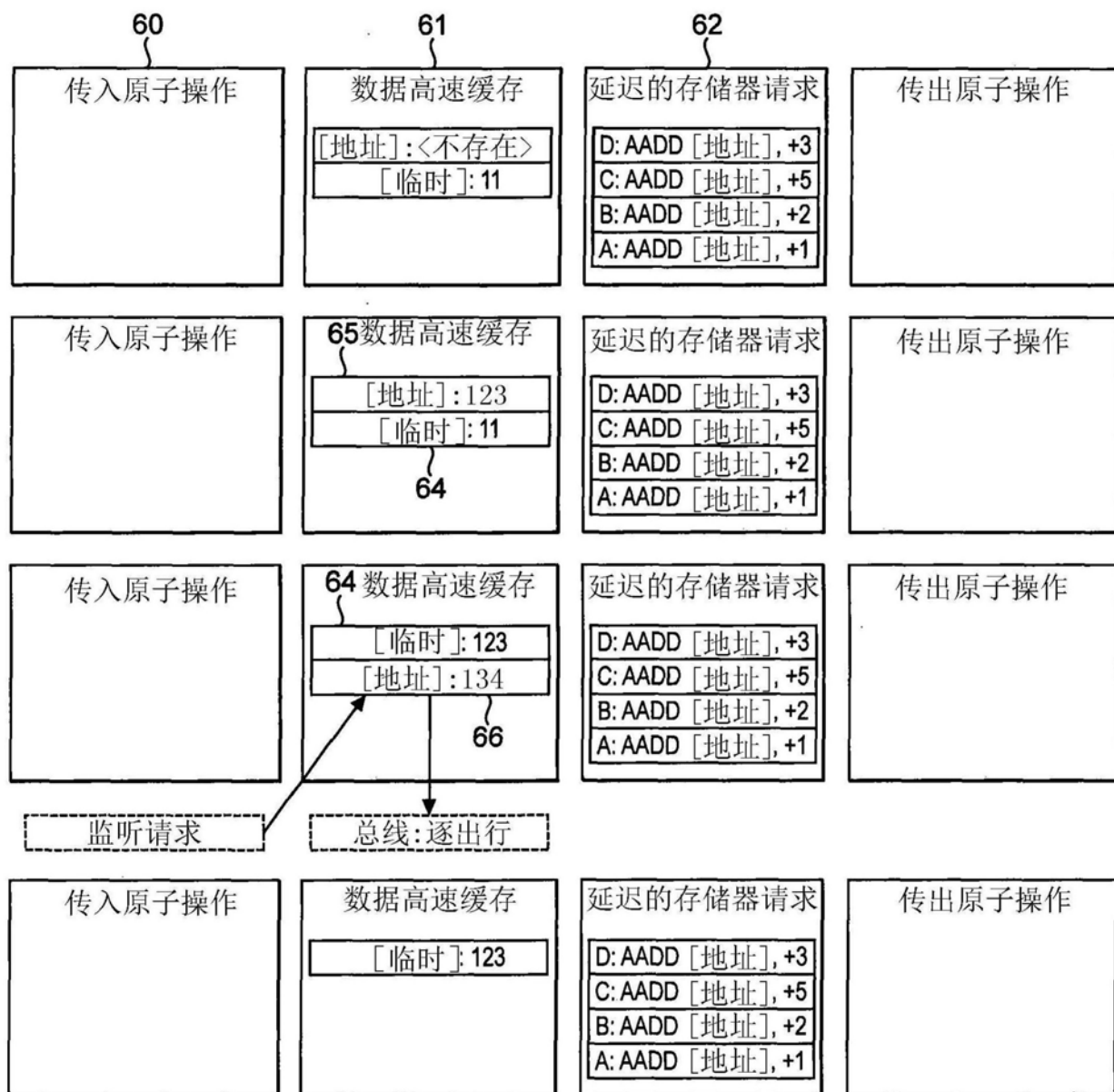


图5B

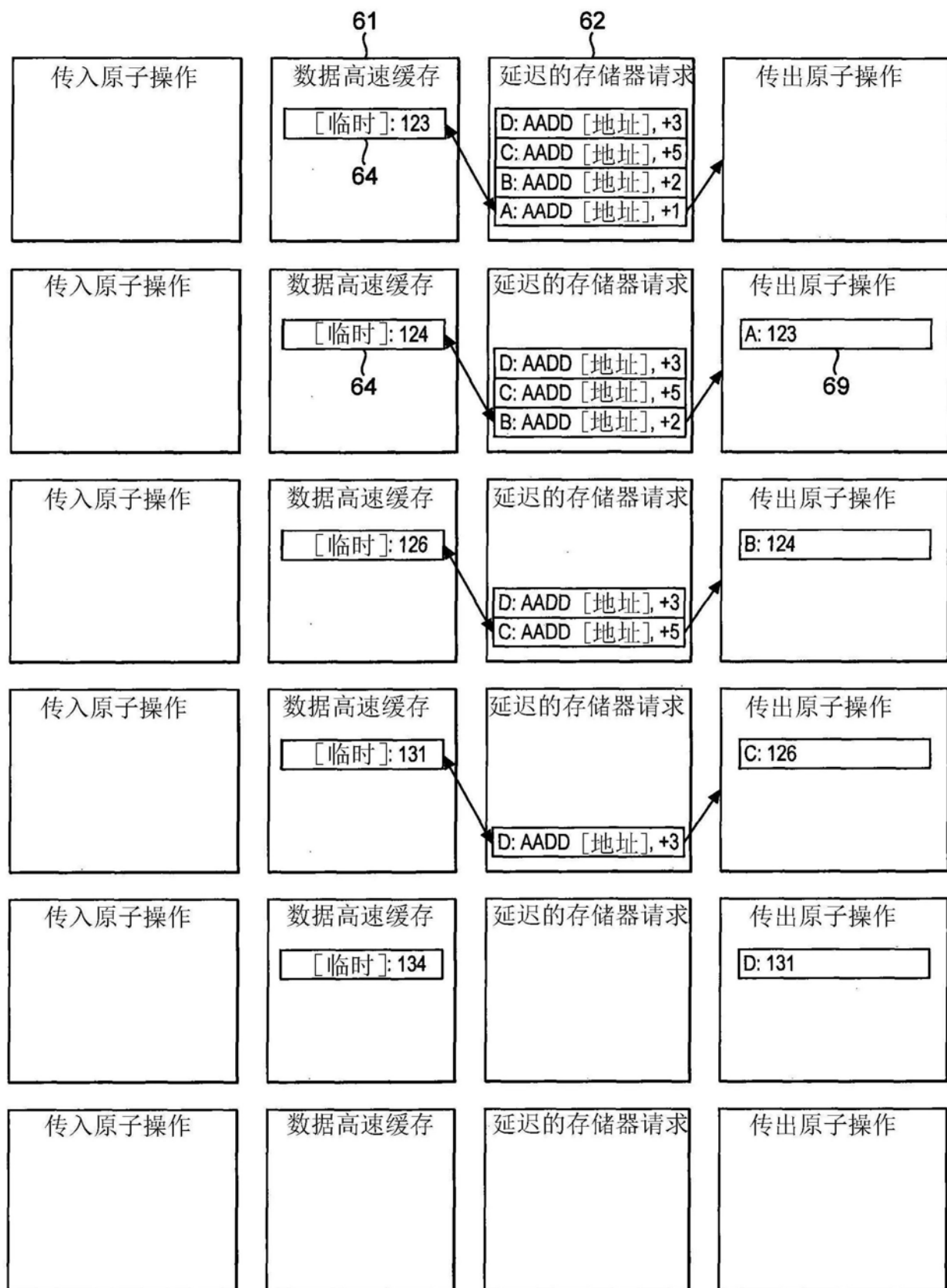


图5C

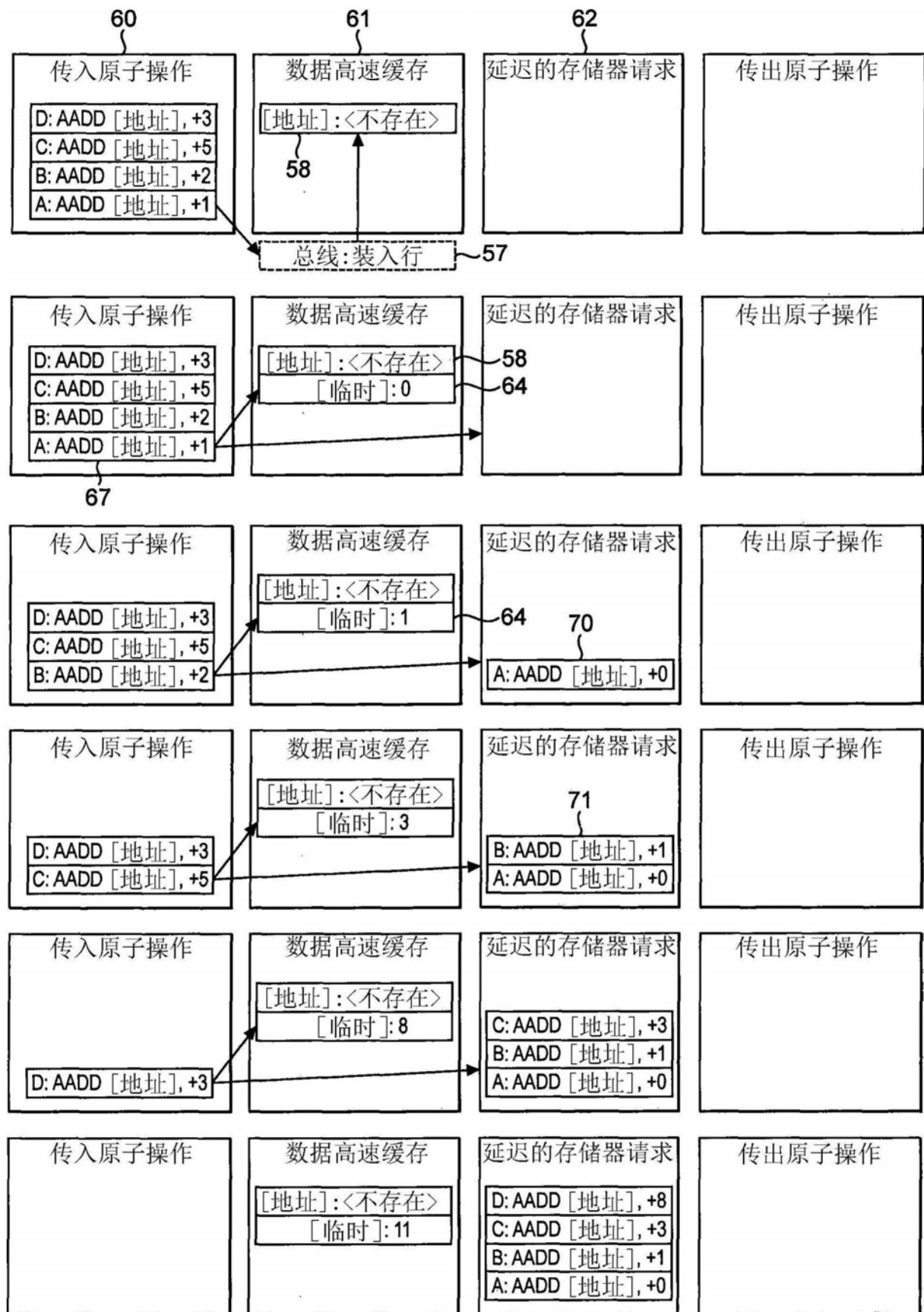


图6A

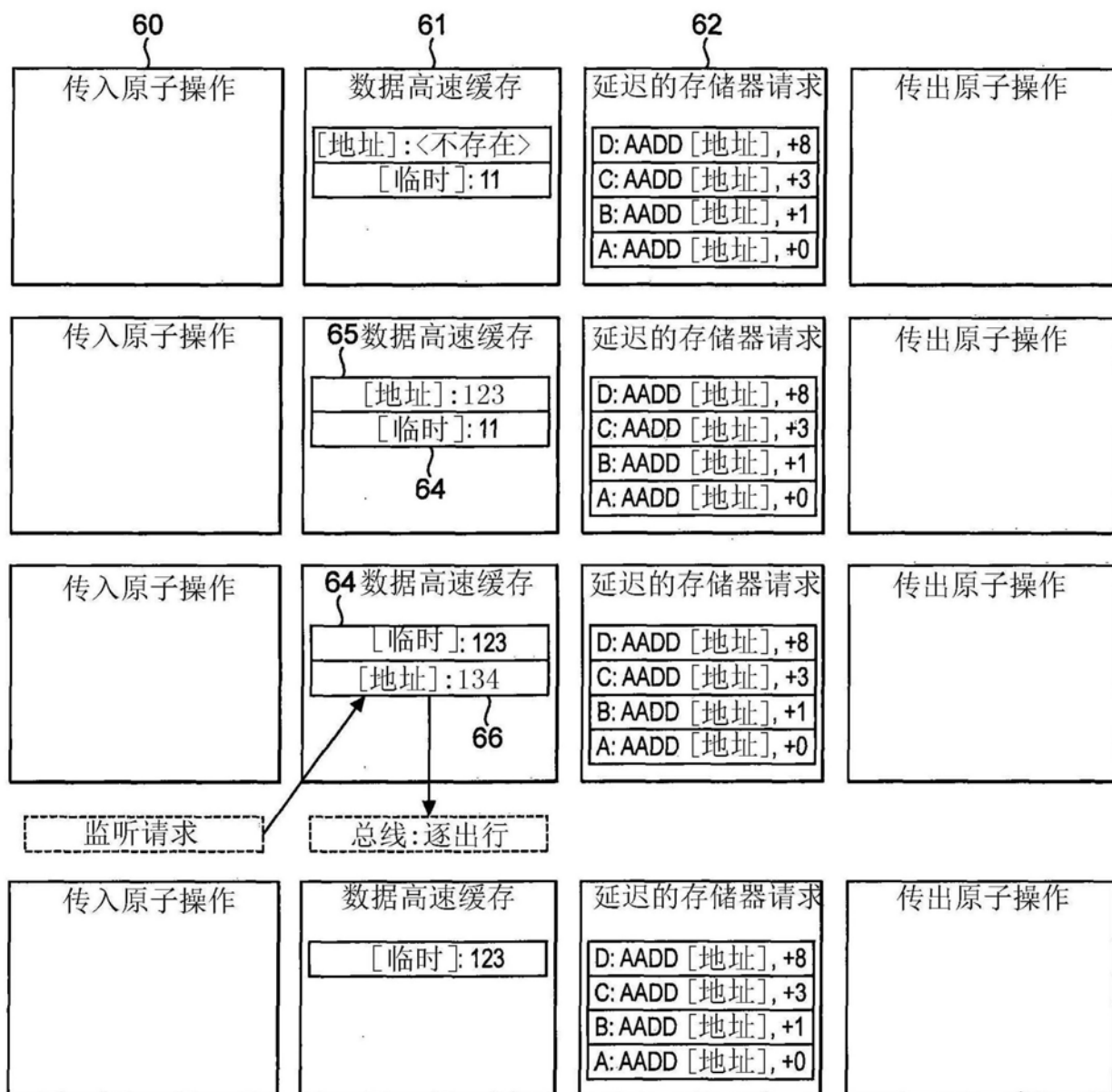


图6B

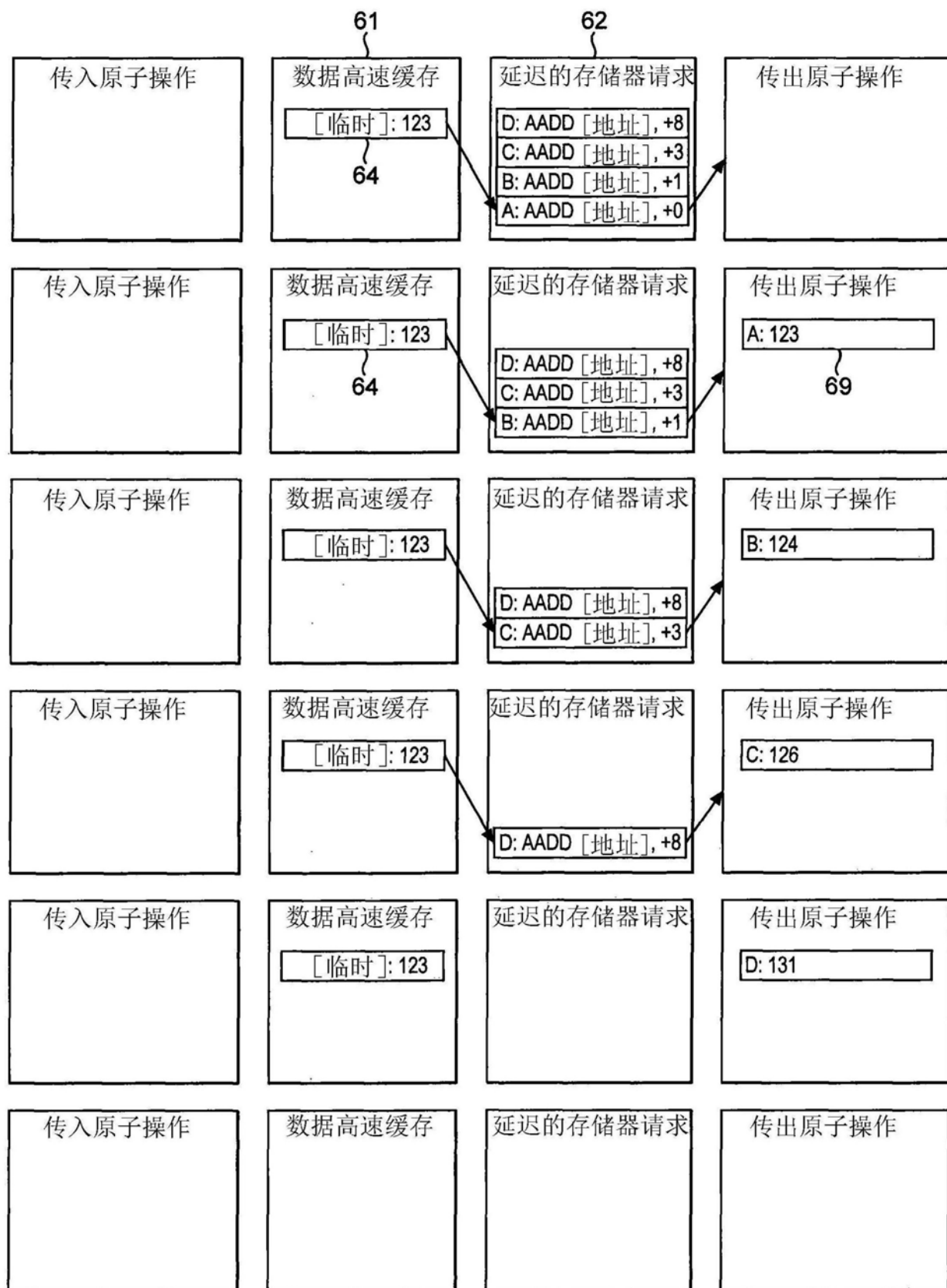


图6C