(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 102867018 B
(45)授权公告日 2015.04.22

(21)申请号 201210265064.5

(22)申请日 2012.07.27

(73)专利权人 北大方正集团有限公司
地址 100871 北京市海淀区成府路298号方正大厦5层
专利权人 方正信息产业控股有限公司
上海方正数字出版技术有限公司

(72)发明人 李书淦 李浩 赵伟 郑程光 孙伟丰 罗正海 李泉 程仁波

(74)专利代理机构 北京英赛嘉华知识产权代理有限责任公司 11204
代理人 王达佐

(51)Int.Cl.
G06F 17/30(2006.01)
G06F 9/54(2006.01)

(56)对比文件
CN 102591726 A,2012.07.18, 说明书第6-10段.
CN 1716209 A,2006.01.04, 说明书第3页第8段-第4页第3段.
JP 2004030312 A,2004.01.29, 全文.
CN 1949206 A,2007.04.18, 全文.

审查员 李燕东

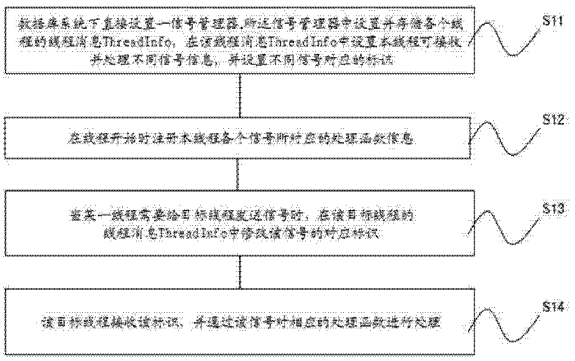
权利要求书1页 说明书6页 附图3页

(54)发明名称

一种数据库系统中线程间的模拟信号通信方法

(57)摘要

本发明公开了一种数据库系统中线程间的模拟信号通信方法，包括：数据库系统下直接设置一信号管理器，信号管理器中设置并存储各个线程的线程消息ThreadInfo，在该线程消息ThreadInfo中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识；在线程开始时注册本线程各个信号所对应的处理函数信息；当某一线程需要给目标线程发送信号时，在该目标线程的线程消息ThreadInfo中修改该信号对应标识；该目标线程接收该标识，并通过该信号对于应的处理函数进行处理。本发明实施方式的线程间的模拟信号通信方法，采用类似进程间通信的信号机制，由信号管理器统一管理和调度各线程信号处理函数，去除了代码中为通信而定义的大量状态变量，使得各线程的任务单一化，从而使得代码的更强的可读性和可维护性，技术方案简单，方便实用。



CN 102867018 B

1. 一种数据库系统中线程间的模拟信号通信方法，其特征在于，包括：

数据库系统下直接设置一信号管理器，所述信号管理器中设置并存储各个线程的线程消息 ThreadInfo，在该线程消息 ThreadInfo 中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识，所述信号管理器为一数组或一队列；

在线程开始时注册本线程各个信号所对应的处理函数信息；

当某一线程需要给目标线程发送信号时，在该目标线程的线程消息 ThreadInfo 中修改该信号的对应标识；

该目标线程接收该标识，并通过该信号对相应的处理函数进行处理。

2. 如权利要求 1 所述的方法，其特征在于，

线程消息 ThreadInfo 通过位置信息来设置不同信息对应的标识，每一位表示一模拟信号及对应的处理函数；

当某一线程需要给该目标线程发送信号时，先找到该目标线程对应的线程消息 ThreadInfo，再找到该信号所在的位置，后将该位置进行修改；

当该目标线程处理好该信号后，将该线程消息 ThreadInfo 对应位置信息重新进行修改。

3. 如权利要求 2 所述的方法，其特征在于，

在该线程消息 ThreadInfo 中设置一信号队列 signalQueue，该队列每一位的位置上表示对应的处理函数，每一位置上的数字 N 表示当前接收到 N 个线程发送当前处理函数需处理的该信号。

4. 如权利要求 3 所述的方法，其特征在于，信号队列 signalQueue 有 32 位，每一位对应一处理函数，并且处理函数的处理结果有两种状态。

一种数据库系统中线程间的模拟信号通信方法

技术领域

[0001] 本发明涉及通信技术领域，特别涉及一种数据库系统中线程间的模拟信号通信方法。

背景技术

[0002] 随着现代信息产业的不断深入发展，对于信息的集成和共享的需求也变得日益迫切。为了增大吞吐量，提高 CPU 的利用率，服务器端的程序一般都会采用多线程技术或者多进程技术。相对于单线程 / 进程来说，多进程 / 线程模式的效率更高，但是随之而来的，程序复杂性会急剧增大，因为多个线程 / 进程为了完成特定的功能必须要相互协作，也就是它们需要同步和通信。

[0003] 在 unix 系统中，进程间的通信可以通过信号来实现，在进程启动时，首先调用 signal(signo,handle) 注册信号处理函数，然后等待其他进程发送信号来作出相应的动作。这种机制非常地方便。而对于多线程来说，通常是通过全局变量来实现的，工作线程通过检测全局变量值的变化来作出相应的动作。这种方式在简单的应用中已经足够了，但是在像数据库系统这种复杂的系统中使用将会使得代码的可读性及可维护性下降。这种通过全局状态变量来通信的做法，对于线程数量少，通信需求不大的情形来说已经足够了，但是对于数据库这种复杂的系统来说，这种技术方案主要有以下几个缺点：全局变量泛滥；代码的可读性下降；可维护性下降；代码的复杂性增加，难以测试等。

[0004] 以某一 XML 数据库使用多线程模式为例，其中涉及的线程主要包括：

[0005] 1. Mater 线程：接收客户端请求，并启动 Backend 线程来处理这些请求；

[0006] 2. Backend 线程：解析并处理客户端的请求；

[0007] 3. BgWriter 线程：将对数据库的改动写回磁盘；

[0008] 4. WalLogger 线程：写日志

[0009] 这些线程之间的协作需要大量的通信，比如 WalLogger 线程会通知 BgWriter 线程，让其写磁盘或者是做 check point，大多数这种通信都是告知性质的。对于这种情况，为了实现他们之间的通信，就会定义大量的全局状态变量，比如说，BgWriter 为了判断是否要写磁盘就需要一个 bool 型的全局变量：Bool g_bNeedSync2Disk；而 WalLogger 在需要写磁盘时就是将这个变量设置成 true，从而就达到了通知 BgWriter 写磁盘的目的。

[0010] 这种通过全局状态变量来通信的做法，对于线程数量少，通信需求不太太的情况来说实现起来还方便，但是对于数据库这种复杂的系统来说，定义过多的全局变量就使得整个系统的代码可读性下降、可维护性差。

发明内容

[0011] 为解决上述问题，本发明技术方案提供了一种数据库管理系统中线程间的模拟信号通信方法，包括：

[0012] 数据库系统下直接设置一信号管理器，信号管理器中设置并存储各个线程的线

程消息 ThreadInfo，在该线程消息 ThreadInfo 中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识，所述信号管理器为一数组或一队列；

[0013] 在线程开始时注册本线程各个信号所对应的处理函数信息；

[0014] 当某一线程需要给目标线程发送信号时，在该目标线程的线程消息 ThreadInfo 中修改该信号对应标识；

[0015] 该目标线程接收该标识，并通过该信号对于应的处理函数进行处理。

[0016] 可选地，线程消息 ThreadInfo 通过位置信息来设置不同信息对应的标识，每一位表示一信号及对应的处理函数；

[0017] 当某一线程需要给该目标线程发送信号时，先找到该目标线程对应的线程消息 ThreadInfo，再找到该信号所在的位置，后将该位置进行修改；

[0018] 当该目标线程处理好该信号后，将该线程消息 ThreadInfo 对应位置信息重新进行修改。

[0019] 可选地，在该线程消息 ThreadInfo 中设置一信号队列 signalQueue，该队列每一位的位置上表示对应的处理函数，每一位置上的数字 N 表示当前接收到 N 个线程发送当前处理函数需处理的该信号。

[0020] 可选地，signalQueue 有 32 位，每一位对应一处理函数，并且处理函数的处理结果有两种状态。

[0021] 与现有技术相比，上述技术方案具有下优点：

[0022] 本发明实施方式的线程间的模拟信号通信方法，采用类似进程间通信的信号机制，由信号管理器统一管理和调度各线程信号处理函数，去除了代码中为通信而定义的大量状态变量，使得各线程的任务单一化，从而使得代码的更强的可读性和可维护性，技术方案简单，方便实用。

附图说明

[0023] 图 1 是本发明实施方式的数据库管理系统中线程间的模拟信号通信方法的流程图；

[0024] 图 2 是本发明实施方式的数据库管理系统中线程间的模拟信号通信方法中的信号管理器与数组结构之间的关系示意图；

[0025] 图 3 是本发明的实施方式的数据库管理系统中的线程间的模拟信号通信方法的一个应用例示意图。

具体实施方式

[0026] 为使本发明的上述目的、特征和优点能够更为明显易懂，下面结合附图对本发明的具体实施方式做详细的说明。在以下描述中阐述了具体细节以便于充分理解本发明。但是本发明能够以多种不同于在此描述的其它方式来实施，本领域技术人员可以在不违背本发明内涵的情况下做类似推广。因此本发明不受下面公开的具体实施方式的限制。

[0027] 本领域的技术人员知道，在 unix 系统中，进程间的通信可以通过信号来实现，在进程启动时，首先调用 signal(signo,handle) 注册信号处理函数，然后等待其他进程发送信号来作出相应的动作。这种机制非常地方便。而对于多线程来说，通常是通过全局变量

来实现的，工作线程通过检测全局变量值的变化作出相应的动作。这种方式在简单的应用中已经足够了，但是在像数据库系统这种复杂的系统中使用将会使得代码的可读性及可维护性下降。

[0028] 为解决现有技术中的问题，本发明的发明人经过研究，提出了一种线程间的模拟信号通信方法。参阅图 1，图 1 是本发明实施方式的数据库管理系统中线程间的模拟信号通信方法的流程图。本发明实施方式的线程间的数据库管理系统中线程间的模拟信号通信方法，包括：

[0029] 数据库系统下直接设置一信号管理器，信号管理器中设置并存储各个线程的线程消息 ThreadInfo，在该线程消息 ThreadInfo 中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识；

[0030] 在线程开始时注册本线程各个信号所对应的处理函数信息；

[0031] 当某一线程需要给目标线程发送信号时，在该目标线程的线程消息 ThreadInfo 中修改该信号对应标识；

[0032] 该目标线程接收该标识，并通过该信号对于应的处理函数进行处理。

[0033] 下面结合说明书附图对本发明实施方式的线程间的模拟信号通信方法做进一步详细的说明。

[0034] 应用例

[0035] 一种数据库系统中线程间的模拟信号通信方法，包括：

[0036] S11：数据库系统下直接设置一信号管理器，信号管理器中设置并存储各个线程的线程消息 ThreadInfo，在该线程消息 ThreadInfo 中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识。

[0037] 信号管理器直接设置在系统下面，其可以为一大数组或一队列。各个线程设置一线程消息 ThreadInfo。信号管理器为每个线程维护着一个结构 ThreadInfo。

[0038] 还请参阅图 2，其给出的是一个最多支持 256 个线程通信的实例，同进从图中可以看出，当前一共管理着两个线程的通信信息。

[0039] 其中 ThreadInfo 的定义的实例如下：

[0040]

```
typedef void (ThreadSigHandler*) ( int signo );
struct ThreadInfo
{
pthread_t tid;                  // tid of this thread
pthread_mutex_t mut;            //protect the access to signalQueue
pthread_cond_t cond;            // wait until some bits in signalQueue been set

int signalQueue;                // one bit per signal
ThreadSigHandler sigHandle[32]; //each handle response to one bit in signalQueue
};
```

[0041] 一个线程对应一个 ThreadInfo，每个 ThreadInfo 可以设置一 signalQueue。signalQueue，该队列每一位的位置上表示对应的处理函数，每一位置上的数字 N 表示当前接收到 N 个线程发送当前处理函数需处理的该信号。比如，某一线程的第三位表示对该数据库的改动写回磁盘，则线程检测到第三位不为“0”时，即启动对该数据库的改动写回磁盘所对应的处理函数，处理完毕后，并将该第三位设置为“0”。

[0042] S12 :在线程开始时注册本线程各个信号所对应的处理函数信息 ;

[0043] 每一位表示哪个处理函数，需要线程可预先识别，则定义该处理函数之初，就先在线程开始时注册。比如

[0044] 为了处理其他线程发送来的信号，程序员要做的只是在线程开始时调用 thread_signal(signo,handle) 注册信号处理函数。其申明如下 :void thread_signal(int signo,ThreadSigHandler handle) ;

[0045] S13 :当某一线程需要给目标线程发送信号时，在该目标线程的线程消息 ThreadInfo 中修改该信号对应标识 ;

[0046] S14 :该目标线程接收该标识，并通过该信号对于应的处理函数进行处理。

[0047] 即 :线程消息 ThreadInfo 通过位置信息来设置不同信息对应的标识，每一位表示一信号及对应的处理函数 ;当某一线程需要给该目标线程发送信号时，先找到该目标线程对应的线程消息 ThreadInfo，再找到该信号所在的位置，后将该位置进行修改 ;当该目标线程处理好该信号后，将该线程消息 ThreadInfo 对应位置信息重新进行修改。

[0048] 更进一步地，在该线程消息 ThreadInfo 中设置一信号队列 signalQueue，该队列每一位的位置上表示对应的处理函数，每一位置上的数字 N 表示当前接收到 N 个线程发送当前处理函数需处理的该信号。signalQueue 有 32 位，每一位对应一处理函数，并且处理函数的处理结果有两种状态。

[0049] 所述的信号处理器通过设置所述发送线程的信号队列控制所述发送线程处理模拟信号。

[0050] 也就是说，在线程中第一次调用这个函数的时候，会创建一个 ThreadInfo 的实例并注册到信号管理器 ThreadSigMgr 中。

[0051] 现在线程可以进入一个死循环，等待别的线程发送信号：

[0052] while(thread_wait())

[0053] {

[0054] thread_process_signals()；

[0055] }

[0056] 这两个函数的申明如下：

[0057] bool thread_wait(void)；

[0058] bool thread_timed_wait(long microseconds)；

[0059] void thread_process_signals(void)；

[0060] 第一个函数会一直等待，直到收到信号时返回，其返回值永远为 true；而 thread_process_signals 会根据发来的信号，调用相应的处理函数，并将 signalQueue 置空。

[0061] 接收信号的线程现在已经作好了准备，其他线程可以通过调用 thread_kill(tid,signo) 发送一个信号给它。然后接收线程就能够像机械一样正常运行了。

[0062] thread_kill 的申明如下：

[0063] int thread_kill(pthread_t tid,int signo)；

[0064] 这个函数给 id 为 tid 的线程发送一个 signo 的信号，如果这个线程不存在，thread_kill 会返回 -1，否则返回 0。

[0065] 有了这个机制，程序员只要把主要精力放在编写处理函数之上，大大简化了代码的复杂度。

[0066] 图 3 给出了一个使用本通信方法进行通信的例子，在这个实例之中，参与通信的是两个线程：Master 线程和 BgWriter 线程，Master 线程接收并处理从客户端发过来的请求并适时地发送信号给 BgWriter，让其写磁盘。BgWriter 线程在启动的时候首先注册了两个信号处理函数：0--sync2Disk，1--exitThread，然后进入到了信号处理循环，如果 thread_wait 返回了，thread_process_signals 中就会去检查是否接收到信号 0，如果接收到了，就调用 sync2Disk 将来 Master 线程作的改动写回磁盘，然后再检查是否接收到信号 1，如果接收到了就会调用 exitThread，在这里也的响应是退出线程。

[0067] 综上所述，本发明技术方案具有下优点：

[0068] 本发明实施方式的线程间的模拟信号通信方法，采用类似进程间通信的信号机制，由信号管理器统一管理和调度各线程信号处理函数，去除了代码中为通信而定义的大量状态变量，使得各线程的任务单一化，从而使得代码的更强的可读性和可维护性，技术方案简单，方便实用。

[0069] 应当理解的是这里所描述的方法和系统可以以各种形式的硬件、软件、固件、专用处理机或者它们的组合实现。尤其是，至少本发明的一部分包括程序指令的应用程序优选实现。这些程序指令被确实地包括在一个或者多个程序存储设备（包括但不限于硬盘，磁性软盘，RAM，ROM，CD，ROM 等）里，并且可由任何包括适当结构的设备或者机器，例如一种具有处理器、内存和输入 / 输出接口的通用数字计算机执行。还应当理解由于附图中描述的一些系统的组成部件和处理步骤优选地以软件实现，所以，系统模块（或者方法步骤的逻

辑流程）之间的连接可能不同，这取决于本发明的编程方式。根据这里给出的指导，相关领域的普通技术人员将能够设计出本发明的这些以及类似的实施方式。

[0070] 以上公开了本发明的多个方面和实施方式，本领域的技术人员会明白本发明的其它方面和实施方式。本发明中公开的多个方面和实施方式只是用于举例说明，并非是对本发明的限定，本发明的真正保护范围和精神应当以权利要求书为准。

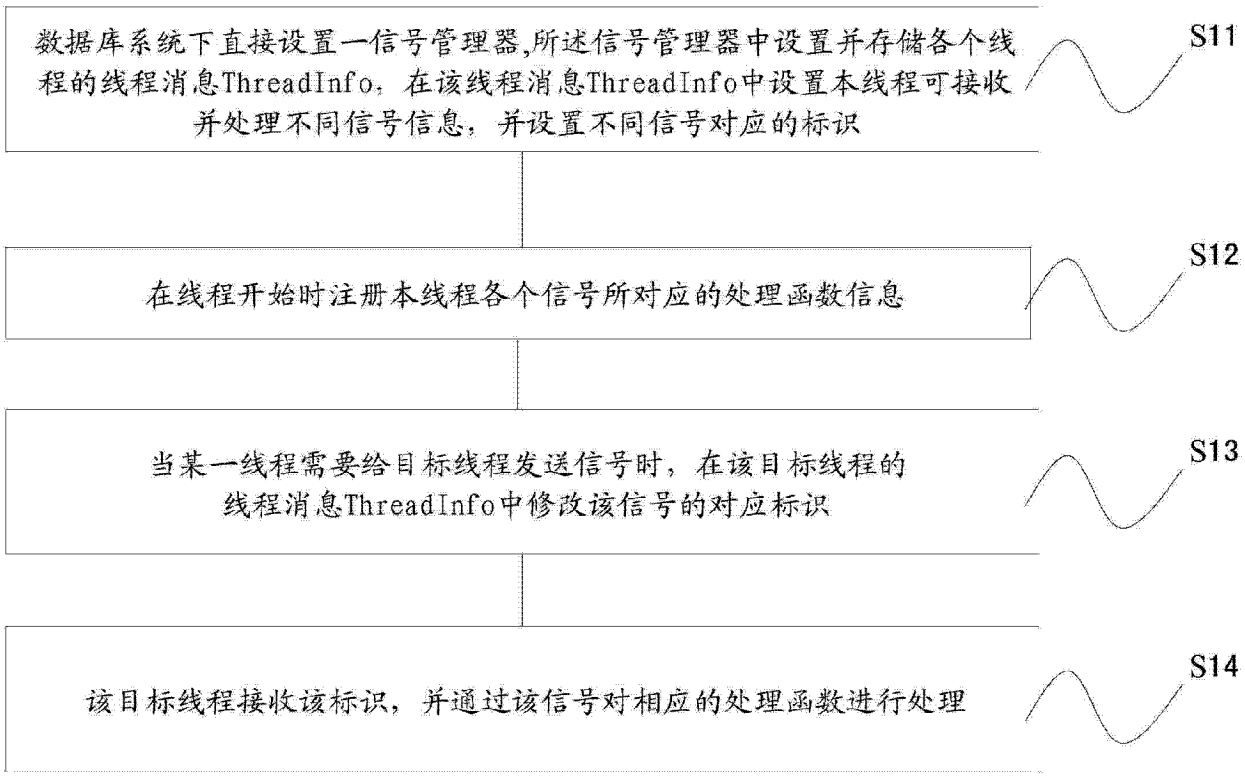

数据库系统下直接设置一信号管理器,所述信号管理器中设置并存储各个线程的线程消息ThreadInfo，在该线程消息ThreadInfo中设置本线程可接收并处理不同信号信息，并设置不同信号对应的标识
S11
在线程开始时注册本线程各个信号所对应的处理函数信息
S12
当某一线程需要给目标线程发送信号时，在该目标线程的线程消息ThreadInfo中修改该信号的对应标识
S13
该目标线程接收该标识，并通过该信号对相应的处理函数进行处理
S14

图 1

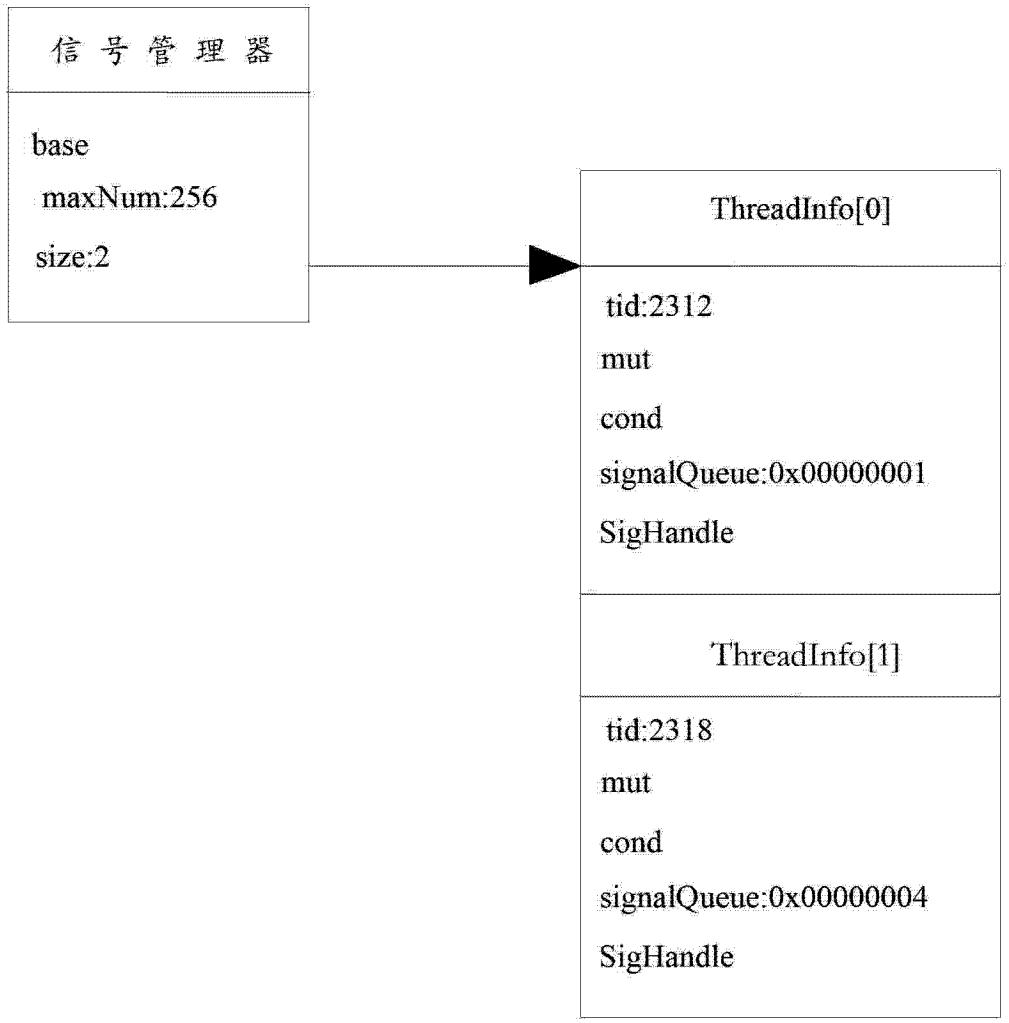

信号管理器
base
maxNum:256
size:2
ThreadInfo[0]
tid:2312
mut
cond
signalQueue:0x00000001
SigHandle
ThreadInfo[1]
tid:2318
mut
cond
signalQueue:0x00000004
SigHandle

图 2

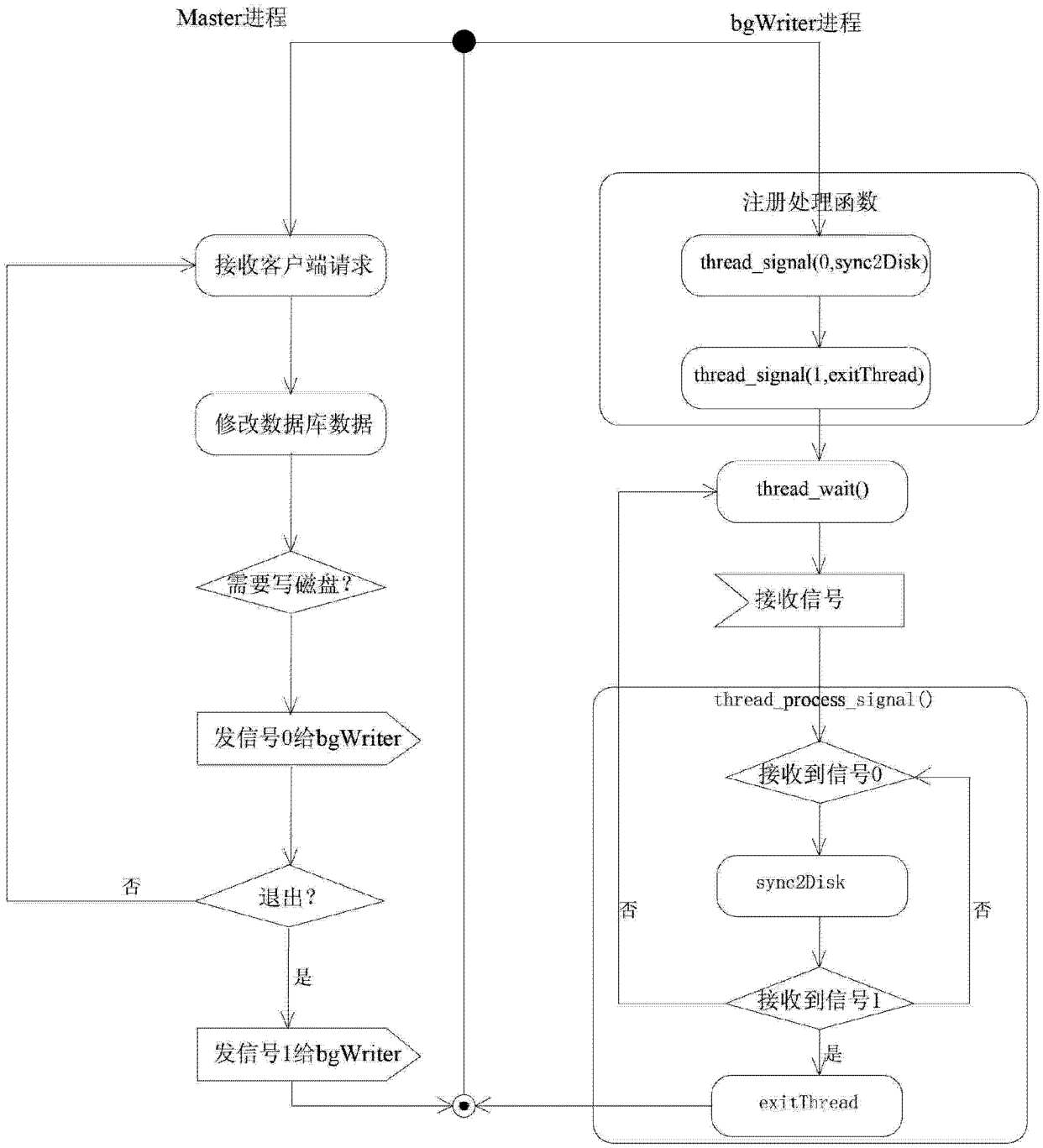

Master进程
bgWriter进程
注册处理函数
thread_signal(0,sync2Disk)
thread_signal(1,exitThread)
接收客户端请求
修改数据库数据
需要写磁盘?
发信号0给bgWriter
否
退出?
是
发信号1给bgWriter
thread_wait()
接收信号
thread_process_signal()
接收到信号0
sync2Disk
否
否
接收到信号1
是
exitThread

图 3