[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/46 (2006.01)
G06F 9/54 (2006.01)

[12] 发明专利说明书

专利号 ZL 200410074122.1

[45] 授权公告日 2007 年 5 月 23 日

[11] 授权公告号 CN 1317640C

[22] 申请日 2004.8.31
[21] 申请号 200410074122.1
[73] 专利权人 华为技术有限公司
地址 518129 广东省深圳市龙岗区坂田华为总部办公楼
[72] 发明人 李 隆
[56] 参考文献
CN1038712 A 1990.1.10
JP2001184221 A 2001.7.6
US5469571 A 1995.11.21
CN1409209 A 2003.4.9
CN1361478 A 2002.7.31
审查员 周述虹

[74] 专利代理机构 北京德琦知识产权代理有限公司
代理人 张颖玲 王 琦

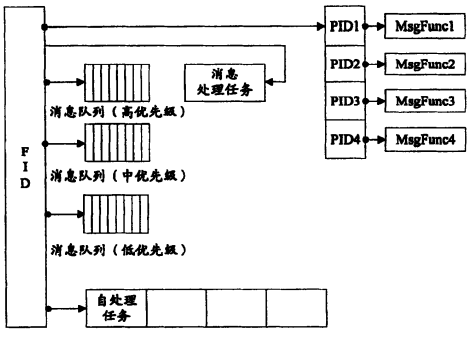
权利要求书 3 页 说明书 12 页 附图 2 页

[54] 发明名称

实时操作系统环境下多任务应用软件模块的管理方法

[57] 摘要

本发明公开了一种实时操作系统环境下多任务应用软件模块的管理方法，在多任务应用程序与实时操作系统之间增加应用软件模块管理层，该方法还包括：将应用程序划分为一个或多个 FID；将各 FID 划分为一个或多个 PID；应用软件模块管理层登记 FID 优先级、创建消息处理任务和消息队列，登记 PID 功能处理函数；还包括：a. 源 PID 申请、填写消息包并发送；b. 应用软件模块管理层将该消息包添加至目的 PID 所属 FID 的消息队列；c. 实时操作系统根据该 FID 的优先级调度消息处理任务；d. 该消息处理任务从消息队列中读取消息包，调用目的 PID 的功能函数。 采用本发明方法为多任务应用程序屏蔽软件模块间同步/互斥等问题，简化开发过程。



1、一种实时操作系统环境下多任务应用软件模块的管理方法，其特征在于，在多任务应用程序与实时操作系统之间设置应用软件模块管理层，该方法还包括：建立应用软件模块与任务之间关系的方法和应用软件模块之间通讯的方法；

所述建立应用软件模块与任务之间关系的方法，包括：

将整个应用程序划分为一个或一个以上功能标识 FID，为各 FID 配置优先级；将每个 FID 划分为一个或一个以上处理标识 PID；应用软件模块管理层为每个 FID 登记优先级、创建消息处理任务和消息队列，为每个 PID 登记功能处理函数；

所述应用软件模块之间通讯的方法，包括：

a. 源 PID 通过应用软件模块管理层申请消息包，源 PID 填写并发送该消息包；

b. 应用软件模块管理层将该消息包添加至目的 PID 所属 FID 的消息队列；

c. 实时操作系统根据 FID 的优先级调度该目的 PID 所属 FID 的消息处理任务；

d. 该消息处理任务从自身的消息队列中读取步骤 b 所述的消息包，通过应用软件模块管理层调用目的 PID 的功能函数。

2、根据权利要求 1 所述的方法，其特征在于，建立应用软件模块与任务之间关系的方法中，所述划分 FID 的方法为：根据应用程序实现的基本功能以及各基本功能的优先级划分。

3、根据权利要求 1 所述的方法，其特征在于，建立应用软件模块与任务之间关系的方法中，所述划分 PID 的方法为：根据各 FID 内部实现的具体功能划分。

4、根据权利要求 1 所述的方法，其特征在于，建立应用软件模块与任务之间关系的方法中，所述应用软件模块管理层还进一步为 FID 登记一个或一个

以上自处理任务及其优先级。

5、根据权利要求 1 所述的方法，其特征在于，建立应用软件模块与任务之间关系的方法中，所述 FID 的消息队列按优先级划分为一个以上的消息队列。

6、根据权利要求 1 或 4 所述的方法，其特征在于，建立应用软件模块与任务之间关系的方法中，所述应用软件模块管理层通过运行应用程序提供的配置文件完成 FID 和 PID 的登记。

7、根据权利要求 1 所述的方法，其特征在于，步骤 b 中，应用软件模块管理层根据消息包中的目的 PID ID 将该消息包添加至目的 PID 所属 FID 的消息队列。

8、根据权利要求 1 所述的方法，其特征在于，步骤 c 中，实时操作系统调度消息处理任务的方法为：

实时操作系统通过应用软件模块管理层提供的任务接口按应用软件模块管理层登记的消息处理任务优先级进行调度。

9、根据权利要求 1 所述的方法，其特征在于，步骤 d 中，所述消息处理任务读取消息包时，应用软件模块管理层为该消息处理任务加入信号量保护。

10、根据权利要求 1 所述的方法，其特征在于，在所述步骤 d 之后，该方法进一步包括：

应用软件模块管理层判断目的 PID 是否转发该消息包，如果是，转发该消息包；否则释放该消息包。

11、根据权利要求 10 所述的方法，其特征在于，所述判断是否转发消息包的方法为：通过识别该消息包的状态进行判断。

12、根据权利要求 10 所述的方法，其特征在于，所述目的 PID 转发消息包时，该目的 PID 根据当前接收方 PID 的 ID 对该消息包的目的 PID ID 进行修改，并再次发送此消息包；所述转发消息包为：执行步骤 b。

13、根据权利要求 1 或 12 所述的方法，其特征在于，源 PID 对应的 CPU

与目的PID对应的CPU不相同时，将源PID对应的CPU作为源CPU，目的PID对应的CPU作为目的CPU，在执行所述步骤b之前，该方法进一步包括：源CPU对应的应用软件模块管理层通过网络发送消息包至目的CPU对应的应用软件模块管理层，收到消息包的应用软件模块管理层申请并填写消息包；

则步骤b至步骤d所述应用软件模块管理层为目的CPU对应的应用软件模块管理层。

14、根据权利要求13所述的方法，其特征在于，根据当前要发送消息包中源CPU ID和目的CPU ID确定源PID对应的CPU与目的PID对应的CPU是否相同。

15、根据权利要求13所述的方法，其特征在于，所述通过网络发送该消息包的方法为：应用软件模块管理层通过自身为应用程序提供的目的CPU对应的网络通讯接口进行发送。

16、根据权利要求13所述的方法，其特征在于，所述通过网络发送消息包的同时，源CPU对应的应用软件模块管理层释放该消息包。

17、根据权利要求13所述的方法，其特征在于，

所述应用软件模块管理层申请消息包之后，进一步保存用于指示该消息包在实时操作系统中内存首地址的指针；

步骤b中，所述应用软件模块管理层将消息包添加至消息队列的处理中，所添加的是消息包的指针；

步骤d中，所述读取消息包的处理中，消息处理任务通过识别消息队列中该消息包的指针读取该消息包。

实时操作系统环境下多任务应用软件模块的管理方法

技术领域

本发明涉及软件模块管理技术，特别是指一种实时操作系统环境下多任务应用软件模块的管理方法。

背景技术

随着科技的发展，嵌入式系统的应用日趋广泛。嵌入式系统一般指非PC 系统，包括硬件和软件两部分，其中，软件部分包括实时操作系统软件和应用软件。应用软件控制着系统的运作和行为，而操作系统控制着应用程序与硬件的交互作用。

其中，任务是实时操作系统中参与 CPU 调度的单位，不同的任务有自己的任务上下文（Context）和堆栈空间，各任务之间共享内存，每个任务都有相应的任务入口函数，不同的任务可以有相同的任务入口函数。每个任务都有优先级，实时操作系统一般采用优先级抢占的调度策略，在多个任务就绪的情况下，只有优先级最高的就绪任务能够获得 CPU 运行。

在实时操作系统环境下，整个实时应用程序按任务被分割成若干个独立的软件模块，每个模块处理应用程序的一部分功能，各个软件模块功能相对独立，耦合性尽量地小并通过多任务通讯机制，如：共享内存、消息队列、管道、信号量等，完成模块之间的同步和协调工作。

但是，这种按照任务来划分软件模块的方法需要上层应用充分了解多任务操作系统软件的原理和机制，实际实现中，需要应用软件开发人员综合考虑任务间通讯、临界资源的保护、同步、互斥、任务间死锁的产生等因素，否则将因不熟悉实时操作系统的机制导致一系列应用软件缺陷。

发明内容

有鉴于此，本发明的目的在于提供一种实时操作系统环境下多任务应用软件模块的管理方法，能够屏蔽掉实时多任务应用软件开发过程中任务间临界资源保护、同步、互斥，防止死锁等问题，简化应用程序开发过程。

为了达到上述目的，本发明的技术方案是这样实现的：

本发明提供了一种实时操作系统环境下多任务应用软件模块的管理方法，在多任务应用程序与实时操作系统之间设置应用软件模块管理层，该方法还包括：建立应用软件模块与任务之间关系的方法和应用软件模块之间通讯的方法；

所述建立应用软件模块与任务之间关系的方法，包括：

将整个应用程序划分为一个或一个以上功能标识（FID），为各FID配置优先级；将每个FID划分为一个或一个以上处理标识（PID）；应用软件模块管理层为每个FID登记优先级、创建消息处理任务和消息队列，为每个PID登记功能处理函数；

所述应用软件模块之间通讯的方法，包括：

a. 源PID通过应用软件模块管理层申请消息包，源PID填写并发送该消息包；

b. 应用软件模块管理层将该消息包添加至目的PID所属FID的消息队列；

c. 实时操作系统根据FID的优先级调度该目的PID所属FID的消息处理任务；

d. 该消息处理任务从自身的消息队列中读取步骤b所述的消息包，通过应用软件模块管理层调用目的PID的功能函数。

建立应用软件模块与任务之间关系的方法中，所述划分FID的方法为：根据应用程序实现的基本功能以及各基本功能的优先级划分。

建立应用软件模块与任务之间关系的方法中，所述划分PID的方法为：根据各FID内部实现的具体功能划分。

建立应用软件模块与任务之间关系的方法中，所述应用软件模块管理层还进一步为 FID 登记一个或一个以上自处理任务及其优先级。

建立应用软件模块与任务之间关系的方法中，所述 FID 的消息队列按优先级划分为一个以上的消息队列。

建立应用软件模块与任务之间关系的方法中，所述应用软件模块管理层通过运行应用程序提供的配置文件完成 FID 和 PID 的登记。

步骤 b 中，应用软件模块管理层根据消息包中的目的 PID ID 将该消息包添加至目的 PID 所属 FID 的消息队列。

步骤 c 中，实时操作系统调度消息处理任务的方法为：

实时操作系统通过应用软件模块管理层提供的任务接口按应用软件模块管理层登记的消息处理任务优先级进行调度。

步骤 d 中，所述消息处理任务读取消息包时，应用软件模块管理层为该消息处理任务加入信号量保护。

在所述步骤 d 之后，该方法进一步包括：

应用软件模块管理层判断目的 PID 是否转发该消息包，如果是，转发该消息包；否则释放该消息包。

其中，所述判断是否转发消息包的方法为：通过识别该消息包的状态进行判断。

其中，所述目的 PID 转发消息包时，该目的 PID 根据当前接收方 PID 的 ID 对该消息包的目的 PID ID 进行修改，并再次发送此消息包；所述转发消息包为：执行步骤 b。

其中，源 PID 对应的 CPU 与目的 PID 对应的 CPU 不相同时，将源 PID 对应的 CPU 作为源 CPU，目的 PID 对应的 CPU 作为目的 CPU，在执行所述步骤 b 之前，该方法进一步包括：源 CPU 对应的应用软件模块管理层通过网络发送消息包至目的 CPU 对应的应用软件模块管理层，收到消息包的应用软件模块管理层申请并填写消息包；

则步骤 b 至步骤 d 所述应用软件模块管理层为目的 CPU 对应的应用软件模块管理层。

其中，根据当前要发送消息包中源 CPU ID 和目的 CPU ID 确定源 PID 对应的 CPU 与目的 PID 对应的 CPU 是否相同。

其中，所述通过网络发送该消息包的方法为：应用软件模块管理层通过自身为应用程序提供的目的 CPU 对应的网络通讯接口进行发送。

其中，所述通过网络发送消息包的同时，源 CPU 对应的应用软件模块管理层释放该消息包。

其中，所述应用软件模块管理层申请消息包之后，进一步保存用于指示该消息包在实时操作系统中内存首地址的指针；

步骤 b 中，所述应用软件模块管理层将消息包添加至消息队列的处理中，所添加的是消息包的指针；

步骤 d 中，所述读取消息包的处理中，消息处理任务通过识别消息队列中该消息包的指针读取该消息包。

综上所述，本发明关键在于：在多任务应用程序与实时操作系统之间增加了应用软件模块管理层，使用建立应用软件模块与任务之间关系的方法，将整个应用程序划分为一个或多个 FID，将各 FID 划分为一个或多个 PID，由应用软件模块管理层为 FID 登记优先级、创建消息处理任务和消息队列，为 PID 登记功能处理函数。在应用软件模块管理层建立起 FID、PID、以及消息处理任务之间的关系之后，使用所述应用软件模块之间通讯的方法来完成应用软件模块之间的通讯；其中，源 PID 通过发送消息包来与目的 PID 进行通讯；应用软件模块管理层负责申请消息包，并将源 PID 所填写的消息包添加至目的 PID 所属 FID 的消息队列；实时操作系统根据 FID 的优先级调度 FID 的消息处理任务，该消息处理任务从自身的消息队列中读取消息包，通过应用软件模块管理层调用目的 PID 的功能函数，完成通讯过程。

可见，本方法通过按照 FID、PID 以及消息处理任务之间关系组织应用

软件模块，并基于此种关系使用本发明方法进行应用软件模块之间的通讯，屏蔽了实时多任务软件中任务划分、任务间通讯、同步、互斥、临界区保护以及函数重入等复杂的技术问题，提高了多任务应用软件的开发效率，避免了因不熟悉多任务特点造成的多任务应用软件系统缺陷。

附图说明

图 1 为本发明方法中 FID、PID 以及任务之间关系示意图；

图 2 为本发明方法的消息包发送一较佳实施例处理流程示意图；

图 3 为本发明方法的消息包处理一较佳实施例处理流程示意图。

具体实施方式

下面结合附图对本发明作进一步的详细描述。

本发明所提供的实时操作系统环境下多任务应用软件模块管理方法，在多任务应用程序与实时操作系统之间增加了应用软件模块管理层，该方法还包括：建立应用软件模块与任务之间关系的方法、和应用软件模块之间通讯的方法。

所述建立应用软件模块与任务之间关系的方法，包括：将整个应用程序划分为一个或多个 FID，为各 FID 配置优先级；将每一个 FID 划分为一个或多个 PID；应用软件模块管理层为每一个 FID 登记优先级、创建消息处理任务和消息队列，为每一个 PID 登记功能处理函数。

其中，可以根据应用程序实现的基本功能以及各基本功能的优先级来划分 FID，并根据各 FID 内部实现的具体功能划分 PID。比如：涉及多协议层功能处理的系统，可以将若干协议层划分为一个 FID，并且通常较低层协议层功能模块优先级较高，因此，将位于低层的协议层划分为优先级高的 FID，将位于高层次的协议层划分为优先级低的 FID；然后，针对各 FID 对应的协议层的功能处理，将各 FID 划分为一个或多个 PID，可以将 FID 按其所包含的协议层划分 PID，也可以根据需要将其包含的每一个协议层划分为多个 PID。

为了实现应用程序功能的扩展，本发明应用软件模块管理层还提供为FID登记自处理任务以及该自处理任务优先级的功能，应用程序需要自己定义任务的优先级和任务入口函数。该自处理任务用于完成一些特殊功能处理，比如：一些软件模块需要周期性地完成一些功能，如，周期性检测底层链路的连接状态、或检查资源状况等等。

应用程序开发时，按上述原则划分FID和PID、定义PID对应的功能处理函数、配置FID的自处理任务及任务入口函数，并将FID与PID的从属关系、PID 对应的功能处理函数、以及自处理任务组织成配置文件，在应用程序运行时，本发明应用软件模块管理层通过运行配置文件中FID和PID的初始化程序，完成FID优先级、FID和PID之间从属关系、以及FID的自处理任务的登记，按FID的优先级为各FID在实时操作系统中创建消息处理任务、以及各消息处理任务对应的消息队列，并为各消息处理任务准备必要的资源如：定时器资源、静态内存资源等。其中，本发明应用软件模块管理层提供了FID消息处理任务的任务入口函数，并根据应用程序的配置文件，在实时操作系统中创建消息处理任务和/或自处理任务、以及消息队列。

其中，所述消息处理任务负责循环读取消息队列中的消息包从而调用PID中的功能处理函数。所述一个FID的消息队列也可以按优先级划分为多个不同优先级的消息队列，高优先级消息队列中的消息包将被优先处理，根据当前接收的消息包的优先级将其放入不同优先级的消息队列。其中，可以根据类型为消息包划分优先级，比如：将由于定时器超时而发送的消息包优先级最高，应被放入优先级最高的消息队列，从而在消息处理任务处理消息时，该消息包将被优先处理。这里，具体消息队列被划分为几个优先级，以及何种类型消息包被划分为何种优先级、放入何种优先级的消息队列根据应用程序需求确定，本发明不进行限定。

图1所示为一个FID内部的PID、消息处理任务、以及消息队列的逻辑关系。在图1中，所述FID对应四个PID，分别为：PID1、PID2、PID3、

PID4；该 FID 还对应一个消息处理任务和一个自处理任务；该消息处理任务对应的消息队列被分为三个优先级，包括：高、中、低优先级三个消息队列。其中，PID1、PID2、PID3 和 PID4 分别对应功能函数 MsgFunc1、MsgFunc2、MsgFunc3 和 MsgFunc4；在应用程序运行时，PID 代表各个应用软件模块，PID 之间通过消息包的发送实现应用软件模块之间的通讯，从而可以调用 PID 的功能函数，实现系统功能。

基于上述建立应用软件模块与任务之间关系的方法，本发明所述应用软件模块之间通讯的方法，包括：

a. 源 PID 通过应用软件模块管理层申请消息包，源 PID 填写并发送该消息包；

b. 应用软件模块管理层将该消息包添加至目的 PID 所属 FID 的消息队列；

c. 实时操作系统根据该 FID 的优先级调度该 FID 的消息处理任务；

d. 该消息处理任务从自身的消息队列中读取步骤 b 所述的消息包，通过应用软件模块管理层调用目的 PID 的功能函数。

下面结合图 2 和图 3，对本发明应用软件模块之间通讯的方法加以详细说明。图 2 所示为消息包发送处理流程示意图，具体处理步骤如下：

步骤 201～202：源 PID 通过应用软件模块管理层申请消息包并填写消息包，该消息包中包含目的 PID 的 ID。

这里，所述申请并填写消息包的方法为：源 PID 对应的应用软件模块利用本发明应用软件模块管理层提供的消息包申请接口向实时操作系统申请内存，应用软件模块管理层返回指针指示该消息包在内存中的首地址；源 PID 通过对该指针所指向的内存地址进行赋值来填写消息包，填写的内容包括：该消息包的消息内容、源 PID ID、目的 PID ID 等信息，从而，该应用软件模块管理层就为该消息包在实时操作系统开辟了一块内存空间用于保存消息包内容、源/目的 PID ID 等信息，并可以通过指针指示该消息包在内存中的首地址。

步骤 203：应用软件模块管理层的消息包发送接口根据所述目的 PID 的 ID 将消息包发送至目的 PID 所属的 FID，将该消息包放入该 FID 的消息队列，结束当前处理。这里，源 PID 所属的 FID 与目的 PID 所属的 FID 可以相同也可以不同。

其中，所述消息包发送的过程中，传递的是消息包在内存中的首地址即消息包的指针，在消息队列中也仅存放消息包的指针，而非真正的消息包。这样就避免了消息处理任务在写消息队列时的赋值或内存拷贝操作，实现系统性能上的腾飞。

以上所述为 CPU 内部应用软件模块之间的消息包发送处理，当应用程序跨 CPU 进行处理时，CPU 之间的应用软件模块也将通过交互消息包进行通讯，因此，在填写消息包时还要进一步填写源 CPU ID 和目的 CPU ID，并在上述步骤 203 之前进一步包括步骤 203'：应用软件模块管理层根据消息包中的源 CPU ID 和目的 CPU ID 是否一致判断源 CPU 与目的 CPU 是否相同，如果是，则执行步骤 203；否则执行步骤 204。所述源 CPU 即为源 PID 所在的 CPU；所述目的 CPU 即为目的 PID 所在的 CPU。

步骤 204：应用软件模块管理层通过网络发送该消息包，并通过自身的消息包释放接口释放实时操作系统中源 CPU 所管理的内存中的消息包，即释放该消息包所占用的内存空间。

这里，所述消息包的网络发送，传递的是完整的消息包，包括：消息内容、源/目的 PID ID 和源/目的 CPU ID。

其中，本发明的应用软件模块管理层为应用程序提供了网络通讯接口，应用程序在开始运行时通过配置文件将各 CPU ID 及其对应的网络通讯接口在应用软件模块管理层中进行登记。因此，本步骤中，源 CPU 所在应用软件模块管理层可以通过识别目的 CPU ID 调用自身与目的 CPU 进行通讯的网络接口，将消息包发送至目的 CPU 所在的应用软件模块管理层。这样，应用本发明方法进行应用软件模块间通讯时，应用软件模块管理层为应用程

序提供统一的消息包发送接口，屏蔽掉 CPU 之间通讯的复杂性，应用程序不必创建和维护多个网络通讯接口，使应用程序的开发更加易于实现。

这里，所述网络通讯模块可以为基于传输控制协议/互联网协议（TCP/IP）的 Socket 接口。但关于具体如何实现所述网络通讯接口，不属本发明重点，这里不作进一步详述。

步骤 205：消息包被发送至目的 CPU 所在的应用软件模块管理层，该应用软件模块管理层通过消息包申请接口向实时操作系统申请内存，通过指针指示该消息包在当前内存中的首地址，并根据接收到的消息包对指针所指向的内存地址进行赋值来填写消息包，在内存中写入该消息包的消息内容、源/目的 PID ID、源/目的 CPU ID 等信息。然后，根据所述目的 PID ID 将消息包发送至目的 PID 所属的 FID，将该消息包的指针放入该 FID 的消息队列。

根据图 2 所述可见，应用本发明方法能够将发送方应用软件模块即源 PID 发送的消息包发送至接收方应用软件模块即目的 PID 所属 FID 的消息队列中，应用程序不用关心软件模块之间同步、互斥的问题，仅关心软件模块实现的功能，将目的 PID ID 包含在消息包中发送至应用软件模块管理层提供的统一消息包发送接口即可，大大简化应用程序的开发过程。

当消息包被发送至目的 PID 所属 FID 的消息处理队列之后，该 FID 的消息处理任务将对该消息包进行处理。图 3 所示为本发明方法消息包处理的一较佳实施例处理流程示意图，具体处理步骤包括：

步骤 301：实时操作系统根据目的 PID 所属 FID 的优先级调度该 FID 的消息处理任务。这里，所述实时操作系统为目的 CPU 所在的实时操作系统，根据图 2 中的描述目的 CPU 与源 CPU 可以相同也可以不同。

其中，由于在应用程序运行时，应用软件模块管理层通过配置文件完成了各 FID 优先级的登记，并按 FID 的优先级为各 FID 创建消息处理任务。这里，该应用软件模块管理层提供了任务接口，通过该任务接口底层的实时操作系统可以按各 FID 的优先级为其创建消息处理任务，并建立消息处理任

务的任务调度队列。如果应用程序还配置了自处理任务及其优先级，则应用软件模块管理层还将通过所述任务接口按自处理任务的优先级将其添加至所述任务调度队列中。实时操作系统的CPU将根据消息处理任务和/或自处理任务的优先级进行任务调度。

由于，实时操作系统一般采用优先级抢占的调度策略，在多个任务就绪的情况下，只有优先级最高的任务获得CPU运行。因此，在调度队列中，当多个消息处理任务和/或自处理任务就绪时，CPU 将调度优先级最高的消息处理任务或自处理任务。这里，所谓任务就绪，就是该任务除了 CPU 资源以外，已经获取了所有运行所必须的资源；对于消息处理任务而言往往是，其所循环读取的消息队列中有消息包等待处理；对于自处理任务来说就是，根据应用程序自行定义的触发机制，其触发条件已经满足，如定时器到时，任务挂起时间到等。

步骤 302：步骤 301 所述 FID 的消息处理任务获得 CPU 运行之后，该消息处理任务从自身的消息队列中读取消息包，根据消息包中的目的 PID ID 调用目的 PID 的功能函数，完成当前应用软件模块之间的通讯。

由于，应用程序通常需要实现两个以上模块之间的通讯，因此在步骤 302 之后，根据具体功能函数的执行结果，所述目的 PID 还有可能将该消息包进行转发，或发送新的消息包给其它 PID。因此，如图 3 所示，在步骤 302 之后，进一步包括：

步骤 303：应用软件模块管理层判断该目的 PID 是否需要转发此消息包，如果是，执行步骤 304；否则执行步骤 305。

这里，在消息包处理过程中，应用软件模块管理层还要实时更新各消息包的状态，所述状态可以为：申请态、准备释放或发送等。当消息申请后之后，其状态为申请态；当消息包被放入消息队列时，其状态为发送；当消息包被消息处理任务读取进行处理时，其状态为准备释放，如果消息包的处理结果为需要转发该消息包，则消息包的状态将变为发送。

其中，应用程序通过应用软件模块管理层提供的消息包发送接口来发送该消息包，如果应用程序调用该消息包发送接口转发该消息包时，应用软件管理层将修改该消息包的状态为“发送”。因此，本步骤中应用软件模块管理层可以根据该消息包的状态判断释放需要转发该消息包，即：如果消息包状态为发送，则该消息包需要转发。

步骤 304：应用软件模块管理层转发该消息包，结束当前处理。

这里，所述转发消息包时，PID 再次对指示该消息包内存首地址的指针所指向的内存地址进行赋值，从而根据当前该消息包接收方 PID 的信息重新填写源/目的 PID ID、源/目的 CPU ID 等参数，再调用应用软件模块管理层的消息包发送接口来转发该消息包。然后，执行所述步骤 203 或 203’，后续处理如图 2 和图 3 所示，不再进一步详述。

步骤 305：应用软件管理层通过消息包释放接口释放该消息包。

其中，消息包的处理结果分为三类：一、当前 PID 准备发送新的消息包给其它 PID；二、当前 PID 准备转发该消息包给其它 PID；三、当前 PID 不再发送消息包给其它 PID。在其中的第二种情况下，该消息包不必释放，在其它两种情况下，由于应用程序暂时不再使用此消息包，则需要将其释放。但是在释放消息包之后，对于第一种情况来说，PID 需要重新申请并填写消息包，即重新执行图 2 和图 3 所示的处理，从而完成与另一应用软件模块的通讯；对于第三种请求来说，PID 停止发送消息包，因此该 PID 不再做任何有关应用软件模块间通讯的操作。

本发明中，消息处理任务在处理消息包时加入信号量进行保护，使得一个消息的处理过程不会被另外一个消息处理任务抢占，从而可以解决应用层不同软件模块之间同步、互斥等问题。其中，所述信号量实现不同软件模块之间同步/互斥，具体原理为：将信号量作为任务等待的一个状态/事件，初始信号量是不可用的，任务或中断处理程序通过释放该信号量来通知这个状态/事件的发生，其它任务由于等待该信号量而阻塞，直到该信号量的释放。

也就是说：消息处理任务在对一个消息包进行处理时，该消息包占用的资源由于存在信号量的保护，不会被其它任务占用，直到消息处理任务完成对该消息包的处理才会将信号量释放，从而其它任务可以对该消息包所占用的资源进行处理。这里，关于具体如何实现信号量机制属公知技术，本文不作进一步详述。

综上所述，应用本发明建立应用软件模块与任务之间关系的方法，将整个应用程序按功能划分为 FID 和 PID，并将 FID 和 PID 的从属关系以及 FID 和/自处理任务的优先级在系统初始化时登记在应用软件模块管理层中，应用软件模块管理层通过自身的任务接口为各 FID 在底层实时操作系统中创建消息处理任务和消息队列，并建立消息处理任务和/或自处理任务的任务调度队列。在此基础之上，采用本发明的应用软件模块之间通讯的方法，能够实现应用软件模块之间通过消息包的发送完成功能函数的调用，从而完成应用软件模块之间的通讯。在发送消息包时，应用程序不必关心各软件模块之间同步互斥的问题，发送方应用软件模块仅需要填写接收方应用软件模块对应的 CPU ID 和 PID ID 即可；在进行消息包处理时，消息处理任务按优先级获得 CPU 的运行，通过读取消息队列中的消息包调用接收方应用软件模块的功能函数，并且为每一次消息包处理提供了信号量保护，保证消息处理的同步和互斥。

可见，本发明方法为多任务应用软件系统提供了应用软件模块管理层，该应用软件模块管理层为应用程序屏蔽了任务、消息队列、信号量、共享内存等操作系统的基本概念，应用程序不需要创建和管理这些相互关联的操作系统对象，只需要关心业务的处理和系统实现的功能，不仅使应用程序开发时软件模块划分等需求分析过程大大简化，并且使各应用软件模块的开发基于功能处理，从而易于实现并保证多任务应用软件系统的稳定性。

总之，以上所述仅为本发明的较佳实施例而已，并非用于限定本发明的保护范围。

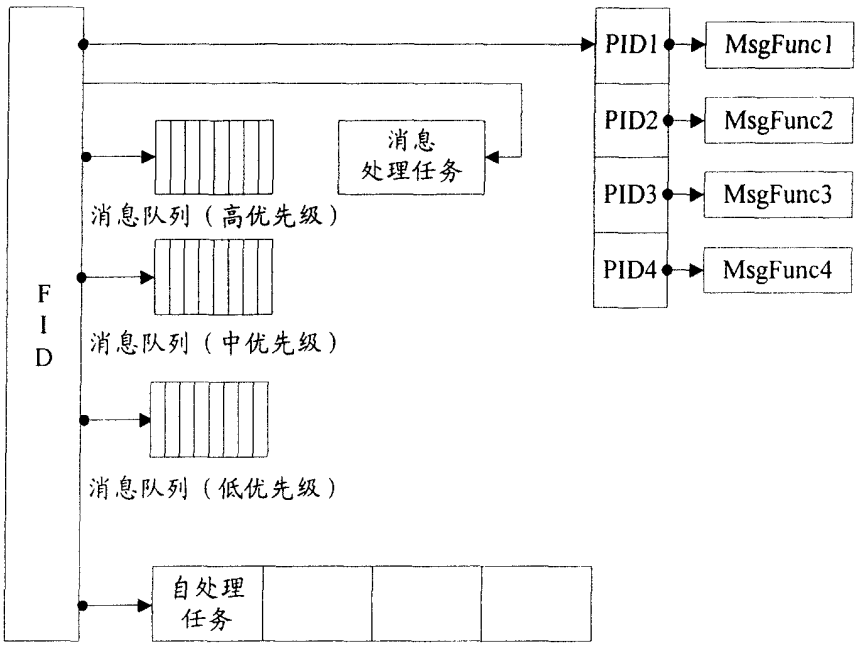

FID
消息队列（高优先级）
消息队列（中优先级）
消息队列（低优先级）
消息处理任务
自处理任务
PID1
PID2
PID3
PID4
MsgFunc1
MsgFunc2
MsgFunc3
MsgFunc4

图 1

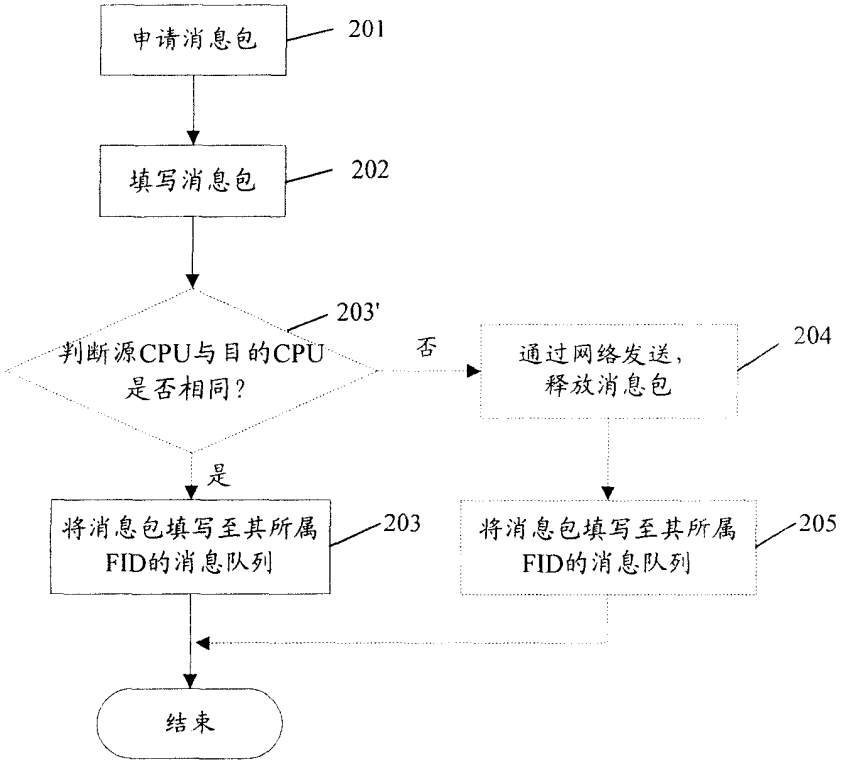

申请消息包
201
填写消息包
202
判断源CPU与目的CPU是否相同？
203'
否
通过网络发送，释放消息包
204
是
将消息包填写至其所属FID的消息队列
203
将消息包填写至其所属FID的消息队列
205
结束

图 2

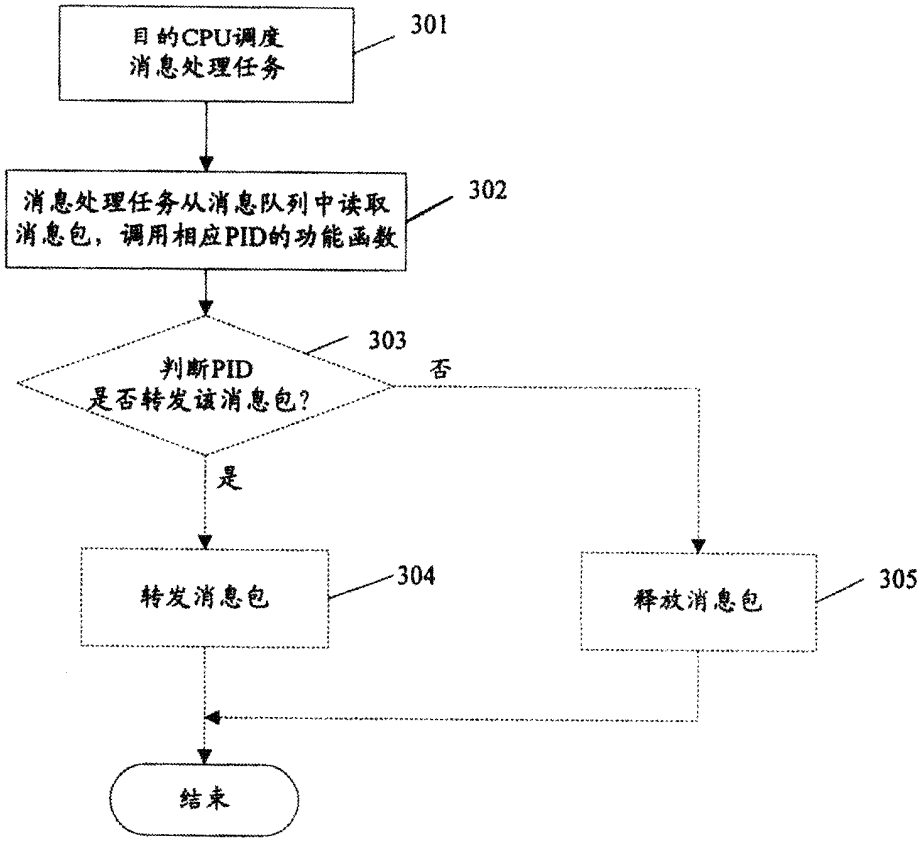

目的CPU调度
消息处理任务
301
消息处理任务从消息队列中读取
消息包，调用相应PID的功能函数
302
判断PID
是否转发该消息包?
303
否
是
转发消息包
304
释放消息包
305
结束

图 3