(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 109933358 B
(45)授权公告日 2022.06.24

(21)申请号 201910080814.3

(22)申请日 2019.01.28

(65)同一申请的已公布的文献号
申请公布号 CN 109933358 A

(43)申请公布日 2019.06.25

(73)专利权人 金卡智能集团股份有限公司
地址 325600 浙江省温州市乐清经济开发区纬十七路291号

(72)发明人 张维史 林志蒙 胡中卫 王浩 赵继高

(74)专利代理机构 杭州华鼎知识产权代理事务所(普通合伙) 33217
专利代理师 项军

(51)Int.Cl.
G06F 8/658(2018.01)
G06F 8/30(2018.01)

(56)对比文件
CN 105487909 A,2016.04.13
CN 107894899 A,2018.04.10
CN 101739288 A,2010.06.16
CN 102375957 A,2012.03.14
CN 101599114 A,2009.12.09
US 2014059525 A1,2014.02.27
US 6212632 B1,2001.04.03
US 2008155510 A1,2008.06.26
US 2015370560 A1,2015.12.24
史毅龙等.基于“龙芯”的VxWorks系统函数在轨更新研究.《电子设计工程》.2015,(第21期),
张海青.电能计量管理信息系统的设计与开发.《中国新技术新产品》.2017,(第24期),

审查员 罗思异

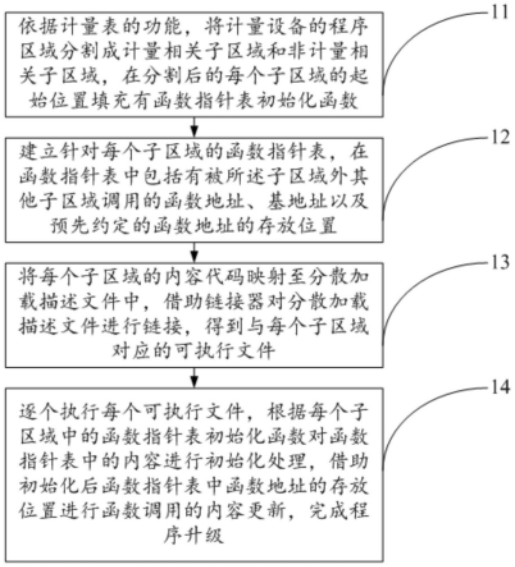
权利要求书1页 说明书7页 附图3页

(54)发明名称

用于减少计量设备程序升级量的控制方法

(57)摘要

本发明实施例提出了用于减少计量设备程序升级量的控制方法,包括将计量设备的程序区域分割成计量相关子区域和非计量相关子区域,在分割后的每个子区域填充有函数指针表初始化函数;建立针对每个子区域的函数指针表;逐个执行每个可执行文件,根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理,借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新。由于在升级过程中仅需确保函数指针表正常即可,因此除了对各子区域中的函数指针表初始化”函数外,被其它区域调用的所有函数不需要进行地址固定的操作,避免了因为函数地址固定且要为这些函数预留一定的存储空间而带来的空间浪费的问题。



CN 109933358 B

1.用于减少计量设备程序升级量的控制方法,其特征在于,所述控制方法包括:

依据计量表的功能,将计量设备的程序区域分割成计量相关子区域和非计量相关子区域,在分割后的每个子区域的起始位置填充有函数指针表初始化函数;

建立针对每个子区域的函数指针表,在函数指针表中包括有被所述子区域外其他子区域调用的函数地址、基地址以及预先约定的函数地址的存放位置;

将每个子区域的内容代码映射至分散加载描述文件中,借助链接器对分散加载描述文件进行链接,得到与每个子区域对应的可执行文件;

逐个执行每个可执行文件,根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理,根据初始化处理后的函数指针表中的基地址与每个函数地址的空间大小、函数表索引号关联的方式进行不同子区域间的函数调用关系更新,完成程序升级。

2.根据权利要求1所述的用于减少计量设备程序升级量的控制方法,其特征在于,所述依据计量表的功能,将计量设备的程序区域分割成计量相关子区域和非计量相关子区域,包括:

根据计量表中关于计量部分的功能定义,将计量设备的程序区域分割得到负责计量的计量相关子区域,以及计量相关子区域以外的非计量相关子区域。

3.根据权利要求1所述的用于减少计量设备程序升级量的控制方法,其特征在于,所述将计量设备的程序区域进行分割,在分割后的每个子区域的起始位置填充有函数指针表初始化函数,包括:

将计量设备的程序区域进行分割,得到包括main函数的关键子区域以及不包括main函数的普通子区域;

在关键子区域和普通子区域的起始位置设有用于对每个子区域对应的函数指针表进行初始化处理的函数指针表初始化函数。

4.根据权利要求3所述的用于减少计量设备程序升级量的控制方法,其特征在于,所述逐个执行每个可执行文件,根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理,借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新,完成程序升级,包括:

执行与每个子区域对应的可执行文件;

根据可执行文件中的函数指针表初始化函数对与当前子区域对应的函数指针表进行初始化处理;

如果需要程序更新时,根据初始化处理后的函数指针表中的基地址与每个函数地址的空间大小、函数表索引号关联的方式进行不同子区域件的函数调用关系更新;

在更新完成后,结束程序升级。

5.根据权利要求1所述的用于减少计量设备程序升级量的控制方法,其特征在于,在每个子区域中包含有功能不同的区域模块。

用于减少计量设备程序升级量的控制方法

技术领域

[0001] 本发明属于程序升级领域，尤其涉及用于减少计量设备程序升级量的控制方法。

背景技术

[0002] 目前，为减少程序升级量所使用的现有技术主要是借助于链接器工具的定位机制来实现的。即通过分散加载描述文件将用户的应用程序分割成多个区域，并将它们分配到不同的地址进行存储和运行。当程序被分割成多个区域后，每个区域中都会存在一些被其它区域调用的函数，为了保证在某个区域的程序发生变更后，其中的被其它区域调用的函数的地址不发生变化(即能被其它区域的程序正常调用)，需要事先将这些被其它区域调用的函数的地址在分散加载描述文件中固定起来。

[0003] 对于需求变更频繁的计量设备，需要为每个区域中的被其它区域调用的函数预留一定的存储空间。然而，当被其它区域调用的函数的数量多时，对于存储空间有限的计量设备，为这些函数都预留一定的存储空间将不可小视，甚至会影响其他区域的布局。由于被其它区域调用的函数存在数量多的情况，造成分散加载描述文件变得庞大，会给该文件的维护、管理带来一定困难。

发明内容

[0004] 为了解决现有技术中存在的缺点和不足，本发明提出了用于减少计量设备程序升级量的控制方法，在程序更新时借助函数指针表仅对发生变化的部分进行更新，从而减少了程序升级量。

[0005] 本实施例提出的用于减少计量设备程序升级量的控制方法，所述控制方法包括：

[0006] 依据计量表的功能，将计量设备的程序区域分割成计量相关子区域和非计量相关子区域，在分割后的每个子区域的起始位置填充有函数指针表初始化函数；

[0007] 建立针对每个子区域的函数指针表，在函数指针表中包括有被所述子区域外其他子区域调用的函数地址、基地址以及预先约定的函数地址的存放位置；

[0008] 将每个子区域的内容代码映射至分散加载描述文件中，借助链接器对分散加载描述文件进行链接，得到与每个子区域对应的可执行文件；

[0009] 逐个执行每个可执行文件，根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理，借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新，完成程序升级。

[0010] 可选的，所述依据计量表记载的功能定义，将计量设备的程序区域分割成计量相关子区域和非计量相关子区域，包括：

[0011] 根据计量表中关于计量部分的功能定义，将计量设备的程序区域分割得到负责计量功能的计量相关子区域，以及计量相关子区域之外的非计量相关子区域。

[0012] 可选的，所述将计量设备的程序区域进行分割，在分割后的每个子区域的起始位置填充有函数指针表初始化函数，包括：

[0013] 将计量设备的程序区域进行分割，得到包括main函数的关键子区域以及不包括main函数的普通子区域；

[0014] 在关键子区域和普通子区域的起始位置设有用于对每个子区域对应的函数指针表进行初始化处理的函数指针表初始化函数。

[0015] 可选的，所述逐个执行每个可执行文件，根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理，借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新，完成程序升级，包括：

[0016] 执行与每个子区域对应的可执行文件；

[0017] 根据可执行文件中的函数指针表初始化函数对与当前子区域对应的函数指针表进行初始化处理；

[0018] 如果需要程序更新时，根据初始化处理后的函数指针表中的基地址与每个函数地址的空间大小、函数表索引号关联的方式进行不同子区域件的函数调用关系更新；

[0019] 在更新完成后，结束程序升级。

[0020] 可选的，在每个子区域中包含有功能不同的区域模块。

[0021] 本发明提供的技术方案带来的有益效果是：

[0022] 1、将计量设备的程序区域按功能不同进行区域分割，并且针对分割后的每个区域建立函数指针表，这样在进行程序升级时，可以根据升级功能不同仅将对应区域的函数指针表中的调用代码进行更新。相对于现有技术无需将全部内容进行更新，从而有效降低了升级量。同时在升级时借助了分散加载描述文件的升级方式，可以避免需要对地址固定的函数需要进行空间预留导致空间浪费的问题。

[0023] 2、基于预定义将程序区域划分为计量区域和非计量功能区域，将负责计量的部分进行独立处理，从而在升级时仅需要对该部分进行升级，可实现部分区域城区的单独升级，同时提升级效率，避免对未进功能改变的程序进行重复升级，减少升级量。

[0024] 3、由于main函数用于在整个计量设备的程序初始化操作，因此在进行区域划分是，基于是否包含main函数这一特征将程序区域进行分割，有利于与执行其他功能的代码进行分割，从而节省不同区域升级时的数据量。

[0025] 4、在具体的升级过程中，在函数指针表进行初始化处理后，可以根据函数指针表中存储的基地址等信息完成不同区域间的函数调用。这样能够避免因函数地址发生变化或函数本身状态变化后影响已有的参数调用，保证升级后程序的正常运行。

附图说明

[0026] 为了更清楚地说明本发明的技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0027] 图1为本实施例提出的用于减少计量设备程序升级量的控制方法的流程示意图；

[0028] 图2为本实施例提出的程序区域进行分割的结果示意图；

[0029] 图3为本实施例提出的分散加载描述文件的流程示意图。

具体实施方式

[0030] 为使本发明的结构和优点更加清楚，下面将结合附图对本发明的结构作进一步地描述。

[0031] 实施例一

[0032] 本实施例提出的用于减少计量设备程序升级量的控制方法，如图1所示，所述控制方法包括：

[0033] 11、依据计量表的功能，将计量设备的程序区域分割成计量相关子区域和非计量相关子区域，在分割后的每个子区域的起始位置填充有函数指针表初始化函数；

[0034] 12、建立针对每个子区域的函数指针表，在函数指针表中包括有被所述子区域外其他子区域调用的函数地址、基地址以及预先约定的函数地址的存放位置；

[0035] 13、将每个子区域的内容代码映射至分散加载描述文件中，借助链接器对分散加载描述文件进行链接，得到与每个子区域对应的可执行文件；

[0036] 14、逐个执行每个可执行文件，根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理，借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新，完成程序升级。

[0037] 在实施中，本实施例提出的用于计量设备程序的控制方法，在现有的分散加载描述文件+函数壳的基础上，增加了对设备程序分块并建立对应每个分块(子区域)的函数指针表的处理方式。由于在函数指针表中存有每个子区域中的函数调用方式，因此在需要对设备程序进行升级时，仅需要根据函数指针表中记载的方式进行函数调用即可。即使设备程序在更新时发生了函数地址变化、函数增加或函数删除的变化，只需根据函数指针表中记载的调用方式进行调取即可保证设备程序的正常运行。

[0038] 在采用记录函数调用方式的函数指针表后，在设备程序需要进行更新时，仅需要对发生变化的部分代码进行更新即可。相对于现有技术中需要对程序进行全面更新的方式，能够有效节省程序升级量，进而降低了设备在升级过程中的能耗，尤其针对采用电池供电的计量设备，可以延长电池的使用时间。

[0039] 另外，由于在升级过程中仅需确保函数指针表正常即可，因此除了对各子区域中的函数指针表初始化”函数外，被其它区域调用的所有函数不需要进行地址固定的操作(原因是函数指针表中的内容是在每次系统初始化过程中动态被更新的)，从而避免了因为函数地址固定且要为这些函数预留一定的存储空间而带来的空间浪费的问题。

[0040] 可选的，步骤11的具体内容包括：

[0041] 111、将计量设备的程序区域进行分割，得到包括main函数的关键子区域以及不包括main函数的普通子区域；

[0042] 112、在关键子区域和普通子区域的起始位置设有用于对每个子区域对应的函数指针表进行初始化处理的函数指针表初始化函数。

[0043] 在实施中，根据实际的存储空间、模块的功能及属性、认证标准等方面将整个计量设备程序分割成多个独立的、分散的区域。如图2所示，计量设备程序被分割成了3个子区域(即对应程序区域1的关键子区域以及对应程序区域2、3的普通子区域)。其中，每个区域包含了一个或多个模块，同时为每个区域分配了一个函数指针表。

[0044] 在图2中可以看出，关键子区域中包含有main模块，该模块中设有用于进行系统初

始化操作main函数。另外在每个子区域中均设有多个不同功能的模块。对应的针对每个子区域均建立函数指针表，在函数指针表中包括对应该子区域中实际的函数调用情况。在程序更新后为了保证计量设备的正常运行，在计量设备重启后，关键子区域中的代码执行到main函数时进行系统初始化操作，接着调用关键子区域中紧邻main函数的函数指针表初始化函数对对应关键子区域的函数指针表进行初始化处理，使得函数指针表在每次系统初始化过程中被动态更新。

[0045] 在上述过程中可以看出，在调用函数指针表初始化函数对函数指针表进行动态更新后，不论计量设备中的函数地址发生何种变化，均可以基于函数指针表中记载的函数调用关系进行正常跨区域调用，保证了设备的正常运行。

[0046] 进一步，为了额保证函数指针表初始化函数的正常调用，需要对其进行地址固定操作，理想的操作为将其固定在每个子区域的起始位置。

[0047] 在执行完步骤12后得到与每个子区域对应的函数指针表，接着执行步骤13所示的分散加载描述文件的步骤，如图3所示，以便得到针对每个子区域的可执行程序。由于分散加载描述文件的步骤已属于成熟技术，此处不再赘述。

[0048] 在执行完步骤13后得到可执行文件后，需要执行步骤14以便完成程序升级步骤。具体的步骤14包括：

[0049] 141、执行与每个子区域对应的可执行文件；

[0050] 142、根据可执行文件中的函数指针表初始化函数对与当前子区域对应的函数指针表进行初始化处理；

[0051] 143、如果需要程序更新时，根据初始化处理后的函数指针表中的基地址与每个函数地址的空间大小、函数表索引号关联的方式进行不同子区域件的函数调用关系更新；

[0052] 144、在更新完成后，结束程序升级。

[0053] 在实施中，在执行针对一个子区域的可执行文件后，触发函数指针表初始化函数，以便对该区域的函数指针表进行初始化处理。在初始化后如果该子区域的内容需要更新，则根据初始化后的函数指针表中的函数调用关系进行跨区域的函数调用。根据步骤12可知，在函数指针表中存有表明调用关系的函数地址、基地址以及预先约定的函数地址的存放位置，因此可通过“函数指针表的基地址+(每个函数地址的空间大小*函数表索引号)”的方式间接调用到另一个区域中的函数。

[0054] 本发明实施例提出了用于减少计量设备程序升级量的控制方法，包括将计量设备的程序区域进行分割，在分割后的每个子区域的起始位置填充有函数指针表初始化函数；建立针对每个子区域的函数指针表，在函数指针表中包括有被所述子区域外其他子区域调用的函数地址、基地址以及预先约定的函数地址的存放位置；将每个子区域的内容代码映射至分散加载描述文件中，借助链接器对分散加载描述文件进行链接，得到与每个子区域对应的可执行文件；逐个执行每个可执行文件，根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理，借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新，完成程序升级。由于在升级过程中仅需确保函数指针表正常即可，因此除了对各子区域中的函数指针表初始化”函数外，被其它区域调用的所有函数不需要进行地址固定的操作，避免了因为函数地址固定且要为这些函数预留一定的存储空间而带来的空间浪费的问题。

[0055] 另外，本方案目前应用于英国智能燃气表项目中，通过使用本方案实现了计量设备程序的“法制计量相关部分(Legally Relevant以下简称LR)”的分离。虽然CPA(英国商业产品安全认证)允许不进行LR分离，但是若不进行LR分离，那么整个软件将视作法制计量相关部分，这样带来的问题是单纯的非法制计量相关部分(Legally non-Relevant以下简称LNR)的变更因为没有进行LR分离认证而需要升级整个计量设备程序，同时还要重新做认证，重新认证又会带来高额的认证费用及额外的认证时间。

[0056] 但是在GBT《计量器具控制软件的通用要求》征求意见CD稿要求计量设备程序要进行“法制计量相关部分”的分离。

[0057] 基于上述要求，基于本实施例中的前述方案，将程序区域中的内容进行了重新划分。表1为该项目经过LR分离后映射到分散加载描述文件中的内容。可看出整个程序被分割成了两个程序区域，即“非法制计量相关部分”的LNR_IROM区域和“法制计量相关部分”的LR_IROM区域。

[0058] 将LR_IROM区域中的‘函数指针表初始化’函数(LrCalledFuncTabInit)放在了该区域中的起始位置。其中LR区域有lr_firmware、lr_measurement、lr_store、lr_ui四个模块。

[0059]

```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LNR_IROM 0x0800D000 0x00070000                    ; “非法制计量相关部分” 区域
{
  APP_LNR_0x0800D000 0x0800D000 0x00070000        ; load address = execution
address
  {
    *.o (RESET, +First)
    *(InRoot$$Sections)
    startup_stm32l476xx.o (+RO)
    .ANY (+RO)                                    ;其它剩余模块
  }

  APP_LNR_IRAM1 0x10003000 0x00002000{  ; RW data
    hal_stack_guard.o (+RW +ZI)
  }

  APP_LNR_IRAM3 0x10005000 0x00003000  {  ; RW data
    lr_measurement.o (+RW +ZI)
  }

  APP_LNR_IRAM2 0x20000000 0x00018000  {  ; RW data
    .ANY (+RW +ZI)
  }
```

```
    }

    LR_IROM 0x08080000 0x00008000                ;“法制计量相关部分” 区域
    {
      APP_LR_0x08080000 0x08080000 0x00008000      ; load address = execution address
      {
        *.o (LrCalledFuncTabInit)                  ;“函数列表初始化”函数在区域的起
始位置
      }
[0060]
      APP_LR +0                                    ;“法制计量相关部分”的所有模块紧随其
后
      {
        lr_firmware.o (+RO)
        lr_store.o (+RO)
        lr_ui.o (+RO)
        lr_measurement.o (+RO)
      }
    }
```

[0061] 表1 分散加载描述文件信息

[0062] 通过表1中的内容可知,能够实现“法制计量相关部分”分离的目的。

[0063] 表2为LR区域的函数指针表的定义及初始化程序,可以看出,LR区域的函数指针表的基地址被固定在了0x20000000位置,并在‘LrCalledFuncTabInit’初始化程序中按照事先约定好的顺序依次完成表内容的填充。由于LNR区域的函数指针表的定义及初始化程序与LR程序类似,这里不再说明。

```
#include "lr_firmware.h"
#include "lr_measurement.h"
#include "lr_store.h"
#include "lr_ui.h"

/*
** Macro definition
*/
[0064]
#define ADDR_OF_LR_FUNC_TAB      (0x20000000u)
#define NUM_OF_LR _CALLED_FUNC (50u)      /*LR 区的函数被
LNR 区调用的数量*/

/*
** LR 区域的函数指针表的定义、初始化
*/
void __attribute__ \
```

```
((at(ADDR_OF_LR_FUNC_TAB))) *LrCalledFuncTab[NUM_OF_LR
_CALLED_FUNC];
void LrCalledFuncTabInit(void)
{
    uint8_t index = 0u;

    /* lr_firmware 模块中被其它区域调用的函数地址*/
    LrCalledFuncTab[index] = &LrFirmwareGetVersion;
    index++;

    /* lr_ui 模块中被其它区域调用的函数地址*/
    LrCalledFuncTab[index] = &LrUiLcdDisplayCumulant;
    index++;

    /* lr_store 模块中被其它区域调用的函数地址*/
    LrCalledFuncTab[index] = &LrStoreMeasurementDataRead;
    index++;
    LrCalledFuncTab[index] = &LrStoreMeasurementConfigDataStore;
    index++;
    LrCalledFuncTab[index] = &LrSotreMeasurementOperationDataStore;
    index++;

    /* lr_measurement 模块中被其它区域调用的函数地址*/
    LrCalledFuncTab[index] = &LrMeasurementInit;
    index++;
    LrCalledFuncTab[index] = &LrMeasurementSelfCheckInit;
    index++;
    LrCalledFuncTab[index] = &LrMeasurementSetTaskId;
    index++;
    LrCalledFuncTab[index] = &LrMeasurementMsgHandle;
    index++;
    /* ... */
}
```

[0065]

[0066] 表2 LR区域的函数指针表的定义及初始化程序

[0067] 上述实施例中的各个序号仅仅为了描述，不代表各部件的组装或使用过程中的先后顺序。

[0068] 以上所述仅为本发明的实施例，并不用以限制本发明，凡在本发明的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

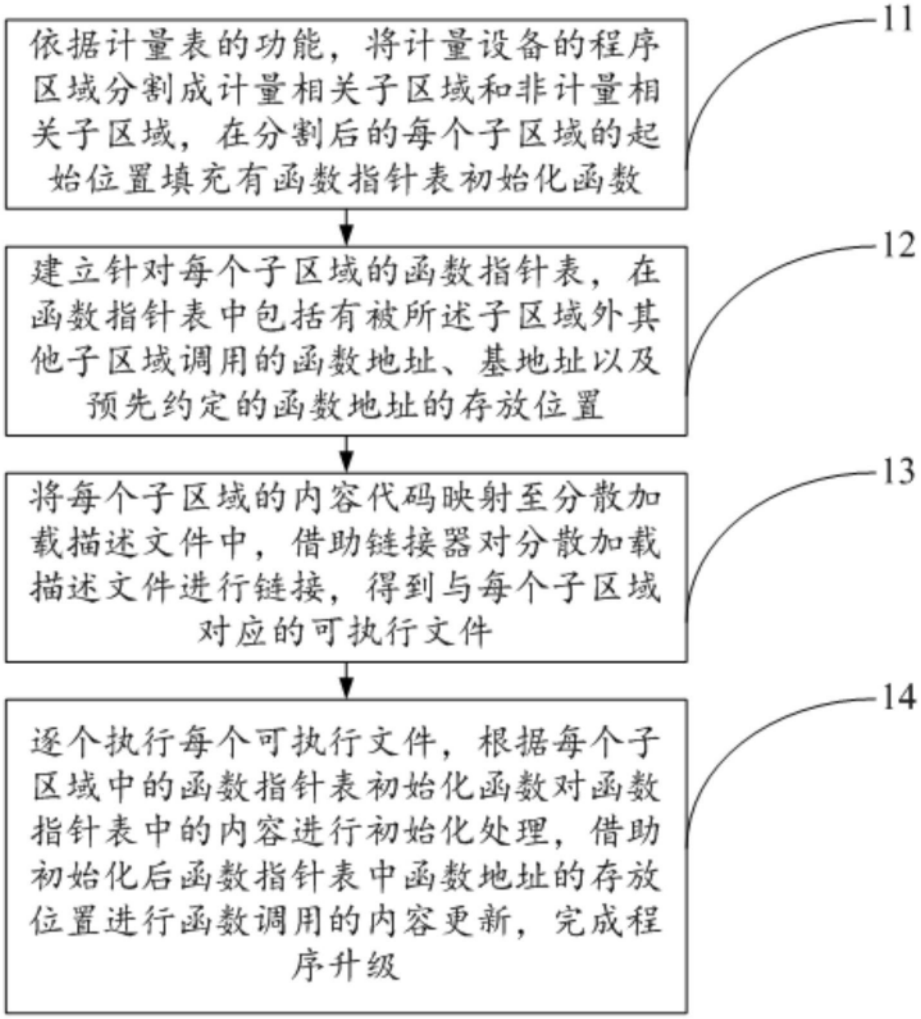

依据计量表的功能，将计量设备的程序区域分割成计量相关子区域和非计量相关子区域，在分割后的每个子区域的起始位置填充有函数指针表初始化函数
11
建立针对每个子区域的函数指针表，在函数指针表中包括有被所述子区域外其他子区域调用的函数地址、基地址以及预先约定的函数地址的存放位置
12
将每个子区域的内容代码映射至分散加载描述文件中，借助链接器对分散加载描述文件进行链接，得到与每个子区域对应的可执行文件
13
逐个执行每个可执行文件，根据每个子区域中的函数指针表初始化函数对函数指针表中的内容进行初始化处理，借助初始化后函数指针表中函数地址的存放位置进行函数调用的内容更新，完成程序升级
14

图1

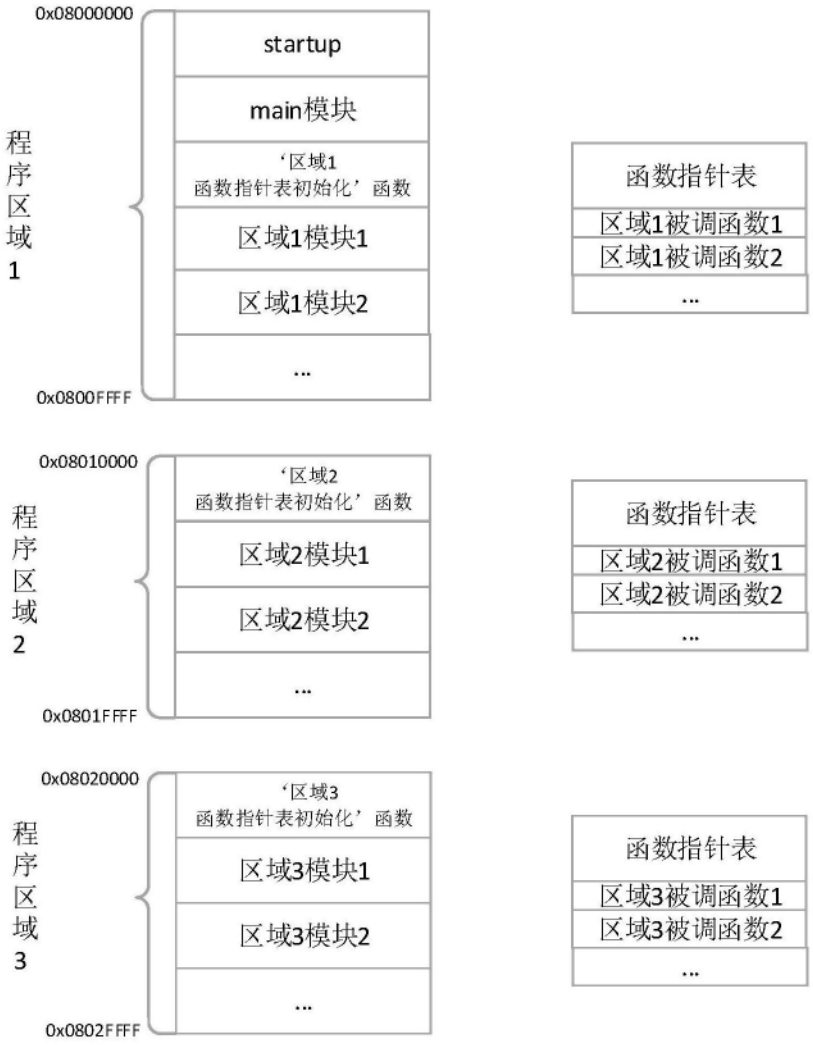

0x08000000
startup
main模块
‘区域1
函数指针表初始化’函数
区域1模块1
区域1模块2
…
0x0800FFFF
程序区域1
函数指针表
区域1被调函数1
区域1被调函数2
…
0x08010000
‘区域2
函数指针表初始化’函数
区域2模块1
区域2模块2
…
0x0801FFFF
程序区域2
函数指针表
区域2被调函数1
区域2被调函数2
…
0x08020000
‘区域3
函数指针表初始化’函数
区域3模块1
区域3模块2
…
0x0802FFFF
程序区域3
函数指针表
区域3被调函数1
区域3被调函数2
…

图2

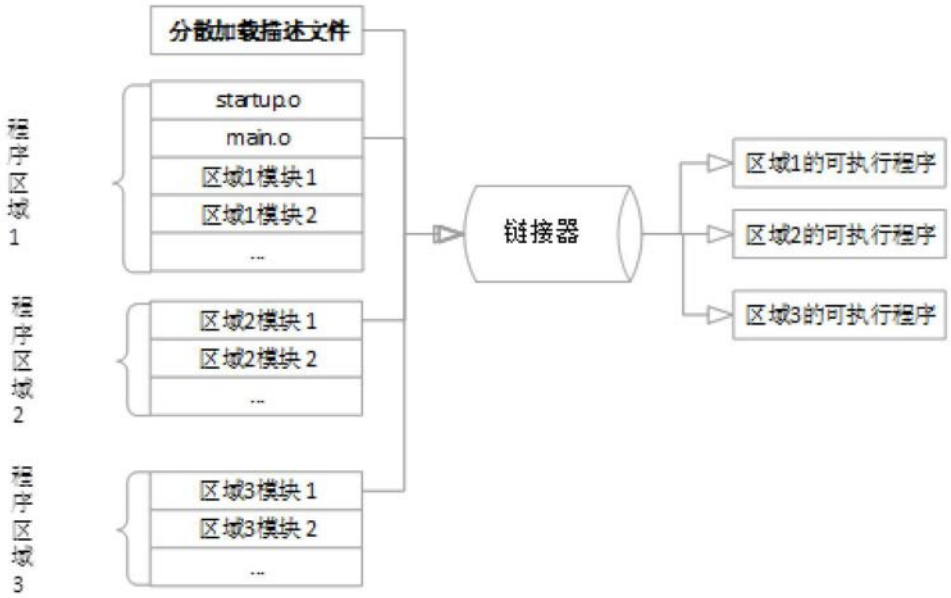

分散加载描述文件
startup.o
main.o
区域1模块1
区域1模块2
...
区域2模块1
区域2模块2
...
区域3模块1
区域3模块2
...
程序区域1
程序区域2
程序区域3
链接器
区域1的可执行程序
区域2的可执行程序
区域3的可执行程序

图3