[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 21/22 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200810086152.2

[43] 公开日 2008 年 8 月 13 日

[11] 公开号 CN 101241534A

[22] 申请日 2003.8.27
[21] 申请号 200810086152.2
分案原申请号 03155346.X
[30] 优先权
[32] 2002. 9. 4 [33] JP [31] 2002 - 258481
[71] 申请人 松下电器产业株式会社
地址 日本大阪府
[72] 发明人 藤原睦 根本祐辅 安井纯一
前田卓治 伊藤孝幸 山田泰司
井上信治

[74] 专利代理机构 中科专利商标代理有限责任公司
代理人 汪惠民

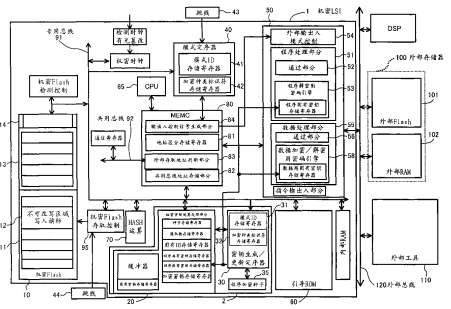
权利要求书 2 页 说明书 14 页 附图 11 页

[54] 发明名称

有加密部分或外部接口的半导体器件及内容再生方法

[57] 摘要

本发明的目的在于：提供一种安全性级别很高的半导体器件。 机密 LSI1，拥有：对程序加密的加密部分 2，及用以在它和外部存储器 100 之间输出入程序、数据的外部接口 50。 在加密部分 2 中，针对由密钥生成/更新定序器 30 判断出的不允许执行的顺序禁止秘密秘钥运算处理 20 的操作。 在外部接口 50 中，程序处理部分 51 和数据处理部分 55 是相互独立的。



1、一种信息处理装置，包括集成电路与存在于所述集成电路的外部的外部存储器，其特征在于：

所述集成电路，包括：

外部接口，在与所述外部存储器之间进行信息的输入与输出，

第一密钥保持部，以不能改写的状态保持第一密钥信息，

第二密钥保持部，以能够改写的状态保持第二密钥信息，

加密解密处理部，利用所述第一密钥信息或者所述第二密钥信息，对经由所述外部接口输入、输出的信息进行加密处理或者解密处理，以及

切换部，根据经由所述外部接口输入、输出的信息，在所述第一密钥信息与所述第二密钥信息之间对所述加密解密处理部进行加密处理或者解密处理所用的密钥进行切换。

2、根据权利要求1所述的信息处理装置，其特征在于：

所述第一密钥保持部所保持的所述第一密钥信息不能被读出到所述集成电路的外部。

3、根据权利要求1所述的信息处理装置，其特征在于：

所述集成电路还包括生成所述第二密钥信息的第二密钥信息生成部。

4、根据权利要求1所述的信息处理装置，其特征在于：

所述集成电路还包括：

随机数取得部，取得随机数信息，

第二密钥信息生成部，以所述随机数信息为基础生成所述第二密钥信息，以及

第二密钥写入部，将已生成的所述第二密钥信息写入所述第二密钥保持部。

5、根据权利要求1所述的信息处理装置，其特征在于：

所述第一密钥保持部在保持所述第一密钥信息以前处于能够改写的状态；

所述集成电路还包括：

第一密钥信息生成部，生成所述第一密钥信息，以及

第一密钥写入部，将已生成的所述第一密钥信息写入所述第一密钥保持部，同时将所述第一密钥保持部设定为不能进行所述第一密钥信息的改写的状态。

6、根据权利要求 1 所述的信息处理装置，其特征在于：

所述第一密钥保持部，包括：

密钥信息保持部，保持所述第一密钥信息，

设定保持部，接收并保持显示是否允许对所述密钥信息保持部进行改写的设定信息，以及

改写控制部，根据所述设定保持部的设定信息在可否对所述密钥信息保持部进行改写之间进行切换；

所述集成电路还包括不可写入设定部，当所述第一密钥信息被写入所述密钥信息保持部时，该不可写入设定部将禁止所述密钥信息保持部被改写的信息作为所述设定信息提供给所述设定保持部。

7、根据权利要求 1 所述的信息处理装置，其特征在于：

所述集成电路还包括：将存储在所述外部存储器中的加密程序解密的程序加密解密部。

8、根据权利要求 7 所述的信息处理装置，其特征在于：

所述程序加密解密部将程序加密后再将该加密后的程序记录到所述外部存储器中。

有加密部分或外部接口的半导体器件及内容再生方法

技术领域

本发明涉及一种提高象用于密钥安装系统的LSI那样的半导体器件的安全性的技术。

背景技术

本案申请人在日本国特愿（专利申请）2001－286881 中，揭示了使密钥安装系统中的密钥的机密性和隐匿性比现有技术下的高的技术。

但因为所述这一技术不是公开了的发明，所述这里没有本来应该叙述的现有技术。

发明内容

本发明的目的，在于：提供一种安全性级别很高的半导体器件，提供一种安全性级别很高的内容再生方法。

为解决上述问题，本发明提供一种半导体器件，包括：执行对程序加密及对程序解密中之至少一个的加密部分，所述加密部分，拥有：能够执行含有对程序进行加密处理及解密处理的多个顺序的加密运算部分及加密控制部分，该加密控制部分，判断是否允许执行所述加密运算部分能够执行的每一个顺序，对被判断为不允许执行的顺序，它就禁止所述加密运算部分的操作。

根据本发明，在加密部分，对加密控制部分判断出加密运算部分可执行的每一个顺序中的不允许执行的顺序，就禁止加密运算部分的操作。换句话说，加密运算部分仅执行由加密控制部分判断出的允许执行的顺序。所以可预先防止顺序的不正当执行，提高安全性。

在本发明所涉及的半导体器件中，所述多个顺序中最好含有密钥的加密处理及解密处理。

在本发明所涉及的半导体器件中，最好是，加密控制部分拥有用以存储模式ID的模式ID存储寄存器，而且，根据所述模式ID存储寄存器中所存储的模式ID的值判断是否允许执行每一个顺序。

最好是，所述加密控制部分拥有：对应于所述每一个顺序而设且用以存储其发行次数的寄存器，所述加密控制部分除了根据所述模式ID的值以外，还根据存储在所述寄存器中的所述每一个顺序的发行次数判断是否允许执行每一个顺序。

再就是，最好是，拥有有不可改写区域的机密存储器，所述不可改写区域中存储着所述模式ID，所述模式ID存储寄存器仅在起动该半导体器件时可写入，而且起动时，写入的是从所述机密存储器的所述不可改写区域读出的所述模式ID。还有，最好是，拥有存储引导程序的引导ROM，将所述模式ID写到所述模式ID存储寄存器中这一操作由存储在所述引导ROM中的引导程序来执行。

再就是，最好是，拥有存储表示该半导体器件是否被第一次起动的安装模式旗标的机密存储器，所述加密控制部分除根据所述模式ID值以外，还根据所述安装模式旗标来判断是否允许执行每一个顺序。

再就是，最好是，本发明所涉及的半导体器件，拥有存储与所述多个顺序中的至少一个相对应的引导程序的引导ROM，所述加密运算部分通过执行存储在所述引导ROM中的引导程序来执行顺序。

再就是，最好是，本发明所涉及的半导体器件拥有控制部件，通过它的控制做到：不能从该半导体器件外部访问所述加密运算部分及加密控制部分所拥有的寄存器。

再就是，最好是，本发明提高一种半导体器件，在该半导体器件和外部存储器之间，拥有用以进行程序、数据的输出入的外部接口，所述外部接口拥有输出入程序的程序处理部分及输出入数据的数据处理部分；所述程序处理部分和数据处理部分相互独立。

根据本发明，在外部接口中，程序处理部分和数据处理部分是独立的。因此，程序遭到不正当执行的风险就被分散了，安全性提高。

再就是，最好是，所述本发明所涉及的半导体器件中的程序处理部分，拥有：将程序原样输出入的通过部分及程序解密用密码引擎。该程序解密

用密码引擎，接收存储在所述外部存储器中的加密了的程序，将它解密为明文程序，再供向该半导体器件内部。

再就是，最好是，所述通过部分，拥有：执行用通过部分和加密用通过部分。在该半导体器件中执行通过所述执行用通过部分所输入的程序，另一方面，通过所述加密用通过部分输入的程序被供向加密部分并被加密。

再就是，最好是，拥有存储表示所述外部存储器中的各个区域和地址之间的对应关系的地址管理信息的地址区分存储寄存器，当访问所述外部存储器读入程序的时候，参考所述地址管理信息，决定使所述加密用通过部分、所述执行用通过部分及所述程序解密用密码引擎中之某一个有效。

再就是，最好是，所述地址区分存储寄存器仅在该半导体器件起动时可以写入。

再就是，最好是，拥有有不可改写区域的机密存储器。所述不可改写区域中存储着所述地址管理信息，起动该半导体器件时，所述地址区分存储寄存器中写入从所述机密存储器的所述不可改写区域读出的所述地址管理信息。

再就是，最好是，拥有具有用以存储模式ID的模式ID存储寄存器的模式定序器，另外还根据存储在所述模式ID存储寄存器中的模式ID的值决定使所述加密用通过部分、所述执行用通过部分及所述程序解密用密码引擎中之某一个有效。

再就是，最好是，所述模式定序器拥有跳线值判断部分，另外还根据由所述跳线值判断部分判断出的跳线值，决定使所述加密用通过部分、所述执行用通过部分及所述程序解密用密码引擎中之某一个有效。

再就是，最好是，在本发明所涉及的半导体器件中，数据处理部分拥有：将数据原样输出入的通过部分及在输出入数据时进行加密或者解密的数据加密解密用密码引擎。

为达到上述目的，本发明提供一种内容再生方法，包括：将存储在外部存储器的不可再生区域中的原来的内容取到LSI中的步骤；在所述LSI中，使用存储在内部存储器中的固有ID生成数据固有密钥的步骤；在所述LSI中，使用所述数据固有密钥对所述原来的内容加密的步骤；将已加密了的内容存储到所述外部存储器的可再生区域中的步骤；将存储在所述

可再生区域中的所述已加密了的内容取到所述 LSI 中，利用所述数据固有密钥将该已加密了的内容解密并再生该已加密了的内容的步骤。

根据本发明，在 LSI 中，用利用存储在内部存储器中的固有 ID 生成的数据固有密钥，对存储在外部存储器的不可再生区域中的原来的内容加密。已加密了的内容，被存储到外部存储器的可再生区域中，再生的时候使用数据固有密钥将该加密了的内容解密。这样以来，因为外部存储器的可再生区域中存储着用从固有 ID 生成的数据固有密钥加密了的内容，故不能由没有相同的数据固有密钥的其他 LSI 来进行再生。结果是，可防止内容被不正当地执行，提高安全性级别。

最好是，在本发明所涉及的内容再生方法下，所述原来的内容为一由数据共有密钥加密的内容，在使用所述数据固有密钥将所述原来的内容加密之前，使用存储在内部存储器中的所述数据共有密钥将所述原来的内容解密。

发明的效果

综上所述，根据本发明，加密运算部分仅执行由加密控制部分判断出为允许执行的顺序。因此，可事先防止顺序的不正当执行。在外部接口，程序处理部分和数据处理部分是独立的。因此，程序遭到不正当执行的风险就分散了。还有，因为在外部存储器的可再生区域，存储着利用由固有 ID 生成的数据固有密钥加密的内容，故不可能由不具有同一个数据固有密钥的其他 LSI 再生这一内容。因此，防止了内容的不正当执行。结果是安全性级别提高了。

附图说明

图 1 为显示本发明的实施例所涉及的作为半导体器件的机密 LSI 的结构的方框图。

图 2 为显示使用了图 1 的机密 LSI 的开发及产品化的整个流程的图。

图 3 为显示引导程序的整个处理流程的流程图。

图 4 为初始值设定处理 SZ1 的流程图。

图 5 为显示图 1 中的机密 LSI1 中的加密部分及其周边的结构的图。

图 6 为显示图 1 中的机密 LSI1 中的共用总线与私用总线的设定方法

的图。

图 7 为显示图 1 中的机密 LSI1 中的外部主接口与其周边的结构的图。

图 8 为显示商品操作模式中外部主接口的操作的图。

图 9 为显示机密存储器的存取控制的图。

图 10 为商品操作模式中的通常引导程序的一个数据流。

图 11 为商品操作模式中的通常引导程序的又一个数据流。

符号说明

1—机密 LSI（半导体器件）；2—加密部分；10—机密存储器；11—不可改写区域；20—秘密密钥运算处理部分（加密运算部分）；30—密钥生成 / 更新定序器（加密控制部分）；31—模式 ID 存储寄存器；33—顺序发行次数存储寄存器；35—存储部分；40—模式定序器；41—模式 ID 存储寄存器；45—跳线值判断部分；50—外部接口；51—程序处理部分；52—通过部分；52a—执行用通过部分；52b—加密用通过部分；53—程序加解密用密码引擎；55—数据处理部分；56—通过部分；58—数据加密 / 解密用密码引擎；60—引导 ROM；81—地址区分存储寄存器；82—共用总线地址存储部分；83—外部存取地址判断部分；100—外部存储器。

具体实施方式

下面，参考附图，说明本发明的实施例。

图 1 为表示本实施例所涉及的作为半导体器件的机密 LSI 的内部结构的方框图。图 1 中的结构是这样的，即机密 LSI1 可通过外部总线 120 和外部存储器 100（闪烁存储器 101 及 RAM102）等连接。而且，可通过施加模式 ID 来设定操作模式。

对本实施例所涉及的主要的结构要素进行简单的说明。

首先，机密 LSI1 包括：含不可改写区域 11 的机密存储器（机密 Flash）10。该不可改写区域 11 中设有不可改写区域写入旗标 12。一旦模式 ID 写到机密存储器 10 中，不可改写区域写入旗标 12 的旗标值就会从“可写入”变成“已经写完”，之后就不能向不可改写区域 11 写入了。需提一下，在本实施例中，机密存储器 10 由闪烁存储器构成，当然并不限于此，只要是非易失性存储器什么都行。

还有，加密部分 2 是对程序进行加密、解密的部分，拥有：作为加密运算部分的秘密密钥运算处理部分 20、作为加密控制部分的密钥生成 / 更新定序器 30、存储程序加密种子的存储部分 35。秘密密钥运算处理部分 20 拥有存储各种密钥等的寄存器，它能够执行包括程序的加密处理或者解密处理的多个顺序（sequences）。密钥生成 / 更新定序器 30 判断是否允许执行秘密密钥运算处理部分 20 可执行的各种顺序，针对已经判断出为不允许执行的顺序，便让秘密密钥运算处理部分 20 停止操作。密钥生成 / 更新定序器 30 拥有模式 ID 存储寄存器 31，根据存储在该模式 ID 存储寄存器 31 中的模式 ID 的值，判断是否允许执行每一个顺序。密钥生成 / 更新定序器 30 还拥有：存储表示密钥或者程序由什么算法、密钥长加密的加密种类标识符的加密种类标识符存储寄存器 32。加密部分 2 的结构和操作的详细情况后述。

模式定序器 40 也拥有模式 ID 存储寄存器 41。该模式定序器 40 根据存储在模式 ID 存储寄存器 41 中的模式 ID 和跳线 43 的值控制外部主接口（I / F）50 的操作，换句话说，是控制通过哪一个主接口来将存储在外部存储器 100 中的程序、数据读进来。由此可控制是否可执行存储在外部存储器 100 中的明文程序。模式定序器 40 还拥有：存储了表示用什么方法将密钥加密的加密种类标识符的加密种类标识符存储寄存器 42。

外部主接口 50，在模式定序器 40 的控制下，通过程序处理部分 51 所拥有的通过部分 52 和程序解密用密码引擎 53、数据处理部分 55 所拥有的通过部分 56 及数据加密 / 解密用密码引擎 58 中之任一个，在它和外部存储器 100 之间进行程序、数据的输出入。

这里，除后述的管理模式及应用程序开发模式以外，通过通过部分 52 输入的程序不会在机密 LSI1 内部执行。换句话说，通过部分 52，是一在明文程序的加密、或者是用其他的密钥对已经加密了的程序再次加密时有效的部分。机密 LSI1 的结构是这样的，除了后述的管理模式及应用程序开发模式以外，不让操作移向通过通过部分 52 输入的程序。因此，即使例如已经成为商品的机密 LSI1 通过通过部分 52 取进了明文程序，也不能执行该明文程序。

引导 ROM60 存储控制机密 LSI1 的启动操作的引导程序。HASH 运

算部分 70，为验证读到机密 LSI1 中的程序的正当性而计算 HASH 值。

还有，在外部存储器 100 中，程序存储在闪烁存储器 101 中；数据（内容）存储在 RAM102 中。外部工具 110 中存储了一开始启动机密 LSI1 时存储在机密存储器 10 中的各种初始值。该初始值的种类随着所设定的操作模式的不同而不同。

图 2 为显示使用了图 1 中的机密 LSI1 的开发及产品化的整个流程的图。如图 2 所示，机密 LSI1 在管理模式（模式 ID: 00）、密钥生成模式（模式 ID: 01）、开发模式（模式 ID: 10）及商品操作模式（模式 ID: 11）这 4 种操作模式下操作。

首先，被设定为管理模式的机密 LSI1 作为管理者用 LSI 操作。在管理者用 LSI 中，开发密钥生成程序（PA1），而且，使用任意的密钥生成密钥对该密钥生成程序加密（PA2）。

被设定为密钥生成模式的机密 LSI1 作为密钥生成用 LSI 操作，在密钥生成用 LSI 中，安装在管理者用 LSI 中生成的、加密的密钥生成程序（PB1）。执行该密钥生成程序以后，就生成了各种密钥（PB2）。

被设定为开发模式的机密 LSI1 作为开发用 LSI 操作，在开发用 LSI 中，开发在实际的产品中执行的应用程序（PC1）。而且，使用程序共有密钥对该应用程序加密（PC2）。

被设定为商品操作模式的机密 LSI1 作为实际的商品 LSI 操作。在商品 LSI 中，安装了在开发用 LSI 中生成的由程序共有密钥加密的应用程序，在其内部，用程序固有密钥将所安装的应用程序变换成加密了的的应用程序（PD1）。在通常的商品操作下执行由程序固有密钥加密了的应用程序。需提一下，在开发用 LSI 中也可为调试应用程序（PC4）来执行该变换处理（PC3）。

机密 LSI1，通过执行存储在引导 ROM60 中的引导程序而进行以下操作。

图 3 为显示引导程序的整个处理过程的流程图。一给机密 LSI1 通上电以后，就由 CPU65 来执行存储在引导 ROM60 中的引导程序。如图 3 所示，首先，将每一个硬件初始化（SZ0）。然后，从外部工具 110 读入各种各样的初始值，并将该读入的初始值设定在机密存储器 10 中（SZ1）。

图 4 为初始值设定处理 SZ1 的流程图。首先，在跳线 44，判断机密存储器 10 是否安装在 LSI 内（SZ11）。接着，判断不可改写区域写入旗标 12 是否为“已写完”（SZ12），因为当为“已写完”（SZ12 为“是”）时，初始值就已经设定在机密存储器 10 中，故结束处理 SZ1。当不可改写区域写入旗标 12 为“可写入”（SZ12 为“否”）时，就将初始值写到机密存储器 10 中。不仅将模式 ID 写到机密存储器 10 的不可改写区域 11 中，还将加密的程序固有密钥、地址管理信息、数据固有密钥写到机密存储器 10 的不可改写区域 11 中（SZ13、SZ16～SZ18）。需提一下，在一开始的判断结果为机密存储器 10 在 LSI 的外部的时候（SZ14 为“否”），就将模式 ID 写在表示商品操作模式的值上（SZ15）。这样以来，机密存储器 10 在 LSI 包外那样的产品，就只可在商品操作模式下操作。

接下来，将不可改写区域写入旗标 12 设定为“已写完”（SZ19）。这样以来，以后的不可改写区域 11 就不能再改写了。而且，还将加密种类标识符及安装模式旗标写到通常区域 13、14 中（SZ1A）。而且，当模式 ID 显示管理模式以外的模式的时候（SZ1B 为“否”），除了将加密了的种类标识符及安装模式旗标写到通常区域 13、14 中以外，还将已加密的共有密钥 / 密钥生成密钥写到通常区域 13、14 中（SZ1C）。

之后，回到图 3，执行前处理 SZ2。这里，设定在机密存储器 10 的不可改写区域 11 中的模式 ID，被设定在密钥生成 / 更新定序器 30 的模式 ID 存储寄存器 31 及模式定序器 40 的模式 ID 存储寄存器 41 中；设定在机密存储器 10 的第 1 通常区域 13 中的加密种类标识符被设定在密钥生成 / 更新定序器 30 的加密种类标识符存储寄存器 32 及模式定序器 40 的加密种类标识符存储寄存器 42 中；机密存储器 10 的不可改写区域 11 中所存储的地址管理信息被设定在 MEMC80 的地址区分存储寄存器 81 中。到这里为止的操作，和图 2 中的初始值设定阶段 PA0、PB0、PC0、PD0 相对应。

之后，根据模式 ID 的值来进行每一个模式下的操作（SZ3）。

当模式 ID 为“00”时，机密 LSI1 成为管理模式，根据跳线 43 的值（SA0）执行明文程序执行处理 SA1 或者是程序加密处理 SA2。在密钥生成程序开发阶段 PA1，进行明文程序执行处理 SA1，生成密钥生成程序。

该密钥生成程序存储在外部存储器100中。在密钥生成程序加密阶段PA2，由任意的密钥生成密钥对密钥生成程序加密。

当模式ID为“01”时，机密LSI1成为密钥生成模式，根据安装模式旗标的值（SB0）来执行密钥生成器制造处理SB1或者是密钥管理/发行处理SB2。在密钥生成器制造阶段PB1，执行密钥生成器制造处理SB1，用程序固有密钥对由任意的密钥生成密钥加密的密钥生成程序再次加密。在密钥管理/发行阶段PB2，执行由程序固有密钥加密了的密钥生成程序，即可执行密钥管理/发行处理SB2，而生成密钥。

当模式ID为“10”时，机密LSI1成为开发模式，根据跳线43的值（SC0）来执行程序加密处理SC1、明文程序执行处理SC2、程序安装处理SC3或者是加密程序执行处理SC4。在应用程序开发阶段PC1，执行明文程序执行处理SC2，开发出应用程序。所开发的应用程序存储在外部存储器100中。在应用程序加密阶段PC2，执行程序加密处理SC1。在应用程序安装阶段PC3，执行程序安装处理SC3；在应用程序调试阶段PC4，执行加密程序执行处理SC4。SC3、SC4这些处理和商品操作模式中的各个处理SD1、SD2一样。

当模式ID为“11”时，机密LSI1成为商品操作模式，根据安装模式旗标的值（SD0）来执行程序安装处理SD1或者通常引导处理SD2。在商品安装阶段PD1，执行程序安装处理SD1。在商品操作阶段PD2，执行通常引导处理SD2。

图5为显示加密部分2及其周边部分的结构的图。如图5所示，密钥生成/更新定序器30，除具有模式ID存储寄存器31及加密种类标识符存储寄存器32外，还拥有对应于利用秘密运算处理部分20执行的每一个顺序而设、用于存储其发行次数的寄存器33，以及参考存储器31、33，判断是否可以执行各个顺序（是否可以执行引导ROM60中的每一个程序及外部程序），而控制秘密密钥运算处理部分20的操作的控制部分34。在机密LSI1中，每一个顺序每发行一次，与之对应的存储器33就加1。

程序加密种子35，为在对密钥解密时或者生成密钥时所用的种子，包括共有密钥用和固有密钥用种子。

在上述商品操作模式、开发模式下，由控制部分34施加制约，做到：

将存储在机密存储器 10 中的值设定在加密部分 2 中的每一个寄存器中的顺序（机密 Flash 取得器 / 程序（loader））、生成密钥及对密钥解密的顺序（密钥定序器）分别只能发行一次。例如，若起动机密 LSI 时，一旦由引导程序将存储在机密存储器中的模式 ID 存储到模式 ID 存储寄存器 31 中，就不能再对模式 ID 进行改写了。还有，若在起动机密 LSI 时，对共有密钥和固有密钥解密，而将它们存储到秘密秘钥运算处理部分 20 内部的寄存器中，就不能再生成密钥，也不能对密钥解密了。因此，即使外部存储器 100 中安装了密钥生成程序，也不能生成密钥。曾经解密的固有密钥存储在外部接口 50 内的固有密钥存储寄存器中，加密了的程序用该固有密钥执行。使用存储在秘密秘钥运算处理部分 20 内部的寄存器中的共有密钥、固有密钥更新程序。

需提一下，在上述密钥生成模式、管理模式中，因为解除了对密钥定序器的限制，故能够生成密钥。

这里，可设置对应于程序加密种子而设的、存储其使用次数的程序加密种子使用次数存储寄存器，来代替顺序发行次数存储寄存器 33。因为在生成密钥及对密钥解密的时候使用程序加密种子，故例如只要用模式 ID 限制其使用次数，即使计算程序加密种子的使用次数，也能限制密钥的生成及对密钥的解密。

还有，程序加密种子并不一定非要包括共有密钥用及固有密钥用种子。

图 6 为显示共用总线及私用总线的设定方法的图。这里，“私用总线”指的是，不能从外部存取（外部存取）的总线；外部接口 50 物理上也并非一定要独立。换句话说，被设定为接在私用总线 91 上的寄存器等，不能通过外部存取进行读出和写入。

地址分别施加给了机密 LSI1 内部的寄存器等，共用总线地址存储部分 82 存储地址中接在共用总线 92 上的寄存器等的地址（在图 6 中，为“0X00000”～“0X10000”）。当有外部存取时，外部存取地址判断部分 83 就参考共用总线地址存储部分 82 判断是否要访问共用总线 92，如果是这样，就接收它。另一方面，因外部存取不是对共用总线 92 的存取的时候，是对私用总线 91 的存取，故拒绝存取。

需提一下，在为来自 CPU65 的存取（内部存取）的时候，不进行这

样的判断，接收内部存取。

图 7 为显示外部接口 50 及其周边的结构的图。在图 7 中，地址区分存储寄存器 81，存储表示外部存储器 100 中的各个区域和地址的对应关系的地址管理信息。这里，外部存储器 100 分为：第 1 区域（设定范围内的程序）、第 2 区域（设定范围外的程序）、第 3 区域（设定区域内的数据）及第 4 区域（设定范围外的数据）这四个区域，存储各自的地址。

比较器 85，参考存储在地址区分存储寄存器 81 中的地址管理信息，判断要输入输出的信息的地址属于上述第 1～第 4 区域中的哪一个区域，并将该判断结果送到输出入控制信号生成部分 84。

输出入控制信号生成部分 84，根据从模式定序器 40 输出的模式 ID 及跳线判断结果、比较器 85 的输出等，判断让外部接口 50 所拥有的哪一个接口有效，并将该判断结果作为输出入控制信号送到外部输出入模式控制部分 54。外部输出入模式控制部分 54 根据所接收的输出入控制信号使某一个接口有效。需提一下，当模式 ID 显示商品操作模式时，一定不让执行通过部分 52 有效。这样以来，存储在外部存储器 100 中的明文程序就受到了限制，而不能执行。

在调试管理模式及开发模式的时候，通过程序处理部分 51 的执行用通过部分 52b 读入存储在第 1 区域的程序；在除了密钥生成模式、商品操作模式或者开发模式调试以外的其他时候，通过程序解码用密码引擎 53 读入存储在第 1 区域的程序。这些程序可执行。另一方面，通过程序处理部分 51 的加密用通过部分 52a 读入存储在第 2 区域的程序，供给到加密部分 2 加密或者是再加密。这些程序不可执行。

通过数据处理部分 55 的数据加密解密用密码引擎 58 读入存储在第 3 区域的数据；通过数据处理部分 55 的通过部分 56 读入存储在第 4 区域的数据。

通过加密用通过部分 52a 取入的程序，在加密部分 2 的秘密秘钥运算处理 20 中被加密或者再加密，之后又通过加密用通过部分 52a 被读到外部存储器 100 的第 1 区域中。这样以来，以后就成为可执行的程序。

需提一下，在地址区分存储寄存器 81 及模式 ID 存储寄存器 41 的数据是通过私用总线设定的。换句话说，数据是通过来自内部的存取设定的。

还有，该数据是设定在机密 LSI1 的重新设定之后，且仅可执行一次。

图 8 为显示外部接口 50 的操作的图。假设是一个商品操作模式。如图 8 所示，安装前，由共有密钥加密了的应用程序存储在外部存储器 100 的第 2 区域（设定范围外）中，因此这种状态是不能执行的。换句话说，存储在第 2 区域的由共有密钥加密的应用程序安装时通过加密用通过部分 52a 取到机密 LSI1 中，该应用程序由共有密钥解密后，又由固有密钥再次加密，再次通过加密用通过部分 52a 存储到外部存储器 100 的第 1 区域（设定范围内）。于是，存储在该第 1 区域中由固有密钥加密的应用程序通过程序解密用密码引擎 53 被取到机密 LSI1 内部，并得以在机密 LSI1 内部执行。

需提一下，在开发模式执行以下操作。首先，调试时，将想要执行的程序事先写到第 1 区域（设定区域内）中准备好。于是，即使是明文程序，也能通过执行用通过部分 52b 取入并得以执行。加密时，将想加密的程序事先写到第 2 区域（设定范围外）中准备好。于是，若起动机密 LSI1，就执行加密的顺序，由共有密钥加密并存储到外部存储器 100 中。在安装调试程序时，将要再次加密的程序事先写到第 2 区域（设定范围外）中准备好。而且，在调试已加密的程序时，将已调试的加密程序事先写到第 1 区域（设定范围内）中准备好。这样就解密、执行了。

图 9 为显示机密存储器 10 的存取控制的图。如图 9 所示，存取控制部分 95 拥有：存储不可改写区域 11 的地址的寄存器 96、存储不可改写区域写入旗标 12 的地址的寄存器 97、可写入／不可写入判断部分 98。这样构成寄存器 96、97，数据一旦写到寄存器 96、97 中，就可通过旗标管理等禁止再次写入。

存取控制如下所述。从 CPU65 到机密存储器 10 的存取一定要通过存取控制部分 95 来执行。在指令为“读”的时候，不管存取地的地址是不可改写区域还是通常区域的地址，机密存储器 10 中的数据都被输出到私用总线 91 中。另一方面，当指令为“写”的时候，可写入／不可写入判断部分 98 参考存储地的地址、存储在寄存器 96 中的地址以及不可改写区域写入旗标 12 的值，判断是否写入。具体而言，判断如下所述。

（旗标“已写完”且不可写入区域）···　不可写入

（旗标“已写完”且通常区域） ··· 可写入

（旗标“未写入”且不可写入区域）··· 可写入

（旗标“未写入”且通常区域） ··· 可写入

需提一下，也为机密存储器 10 准备了“区域消去”、“芯片消去”等指令。在不可改写区域 11 旗标 12 为“已写完”时，通常区域接收“区域消去”，而不可写入区域却不接收。不接收“芯片消去”。

还有，在再生内容（数据）的时候，采用以下方法能提高安全性。

数据一开始放在外部 RAM102 的第 4 区域（设定范围外），在数据被放在第 4 区域的时候，数据或是处于由数据共有密钥（和程序共有密钥不同）加密的状态或是处于明文状态。因此就有被其他的 LSI 不正当利用的可能性，而存在安全性问题。

为解决这一问题，对于那些特别想防止被不正当利用的图像、音乐等内容而言，是这样来制造再生内容的程序的，即只能再生存储在外部 RAM102 的第 3 区域（设定范围内）内的内容。在将放在第 3 区域内的数据取到机密 LSI1 时，该数据在数据加密解密用密码引擎 58 中被解密。因为进行这一解密时所使用的数据固有密钥由固有 ID 和随机数组成，所以该数据固有密钥不仅随机密 LSI1 的不同而不同，而且每起动一次该数据固有密钥也不同。因此，数据不容易被不正当利用，安全性也得到提高。需提一下，因为再生内容的程序也由固有密钥加密，故很难被窜改。

图 10 及图 11 为商品操作模式下的通常引导处理的数据流程图。在图 10 中，首先，将存储在机密存储器 10 的不可改写区域 11 中且加密了的程序固有密钥 Enc（程序固有密钥，MK0）、Enc（MK0，CK）设定在秘密密钥运算处理部分 20 的加密密钥存储寄存器中。用所安装的程序加密种子对该已加密了的程序固有密钥解密而得到程序固有密钥。所得到的程序固有密钥设定在外部主接口 50 的程序解密用密码引擎 53 的程序固有密钥存储寄存器中。之后，将存储在机密存储器 10 的不可改写区域 11 中的数据固有 ID 设定在秘密密钥运算处理部分 20 的固有 ID 存储寄存器中。由 CPU65 产生随机数，并将该随机数设定在秘密密钥运算处理部分 20 的随机数存储寄存器中。由秘密密钥运算处理部分 20 从数据固有 ID 和随机数生成数据固有密钥。已生成的数据固有密钥设定在外部主接口 50 的

数据加密解密用密码引擎 58 的数据固有密钥存储寄存器中。

之后，在图 11 中，通过外部主接口 50 所拥有的程序处理部分 51 的程序解密用密码引擎 53，对存储在外部存储器 100 中且由程序固有密钥加密了的应用程序 Enc(应用程序,程序固有密钥)解密并将它取到 HASH 运算部分 70 中，计算 HASH 值。接着，对该计算出的 HASH 值和存储在机密存储器 10 的通常区域 13 中的 HASH 值进行比较，检查应用程序是否被窜改。当 HASH 值一致时，处理将移到存储在外部存储器 100 的应用程序 Enc(应用程序,程序固有密钥),执行应用。需提一下，当 HASH 值不一致时，就推测是有不正当行为，而执行不正当存取控制处理。

由 CPU65 执行应用程序。换句话说,因为由机密 LSI1 内部的 CPU65 作为主体（master）进行存取控制，故外部存取地址判断部分 83 就和以后的操作即内部存取无关。通过应用程序,由数据共有密钥加密的内容(原来的内容)被从外部 RAM102 的第 4 区域(不可再生区域)取到机密 LSI1 中。利用已写到机密存储器 10 中的数据共有密钥在秘密秘钥运算处理部分 20 中将所取入的内容解密。之后，再通过外部接口 50 的数据处理部分 55 中的数据加密解密用密码引擎 58 利用数据固有密钥对所取入的内容加密，并写到外部 RAM102 的第 3 区域（可再生区域）。之后，由该数据固有密钥加密了的内容便可再生，再生时，通过外部接口 50 的数据处理部分 55 中的数据加密解密用密码引擎 58 由数据固有密钥解密。

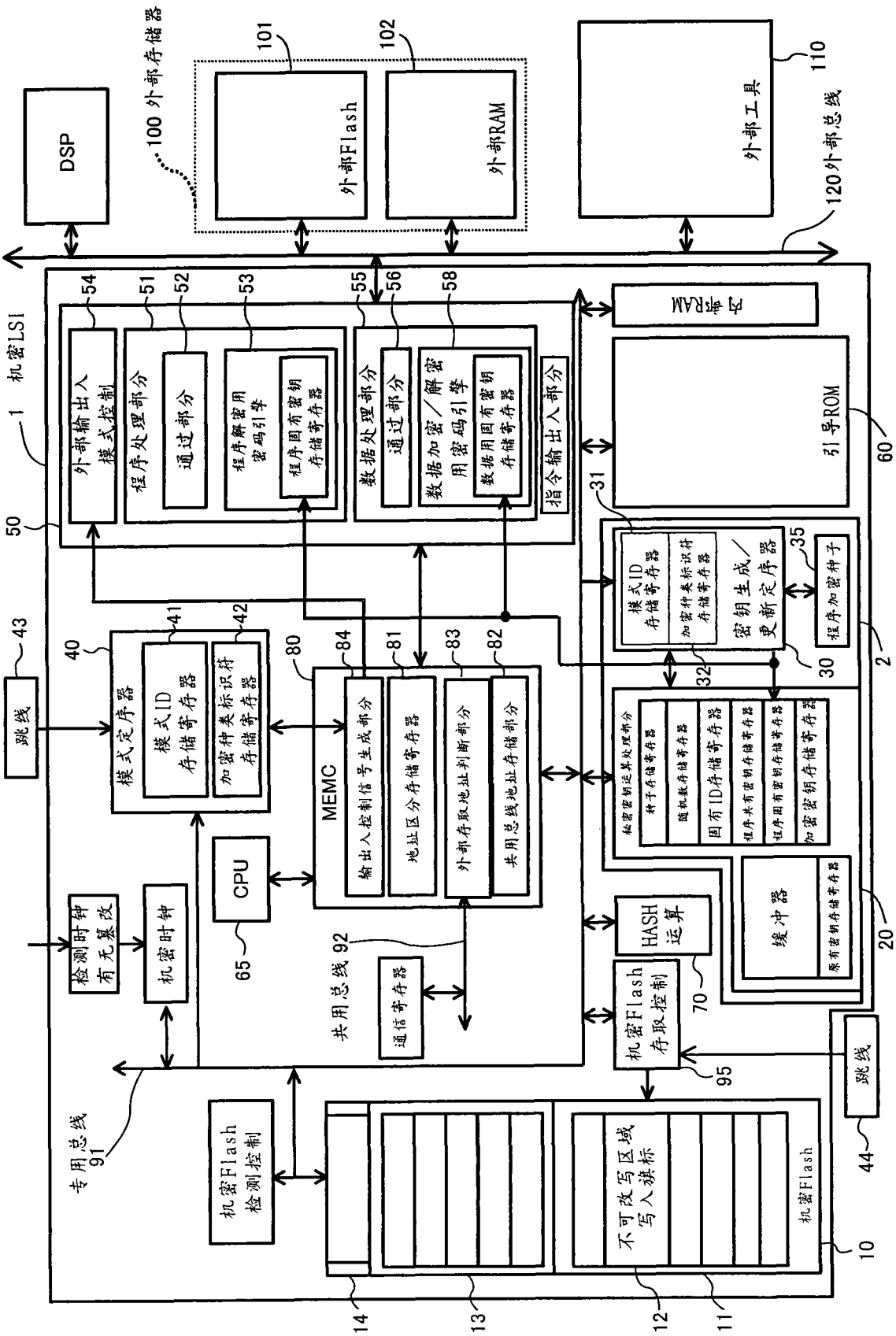

DSP
100 外部存储器
101
外部Flash
102
外部RAM
外部工具
110
120 外部总线
机密LSI
1
外部输出入模式控制
程序处理部分
通过部分
程序解密用密码引擎
程序固有密钥存储寄存器
数据处理部分
通过部分
数据加密／解密用密码引擎
数据用固有密钥存储寄存器
指令输出入部分
50
54
51
52
53
55
56
58
安全
引导ROM
60
31
模式ID存储寄存器
加密种类标识符存储寄存器
密钥生成／更新定序器
35
程序加密种子
30
2
32
跳线
43
40
模式定序器
模式ID存储寄存器
加密种类标识符存储寄存器
41
42
80
MEMC
输出入控制信号生成部分
地址区分存储寄存器
外部存取地址判断部分
共用总线地址存储寄存器
84
81
83
82
检测时钟有无篡改
机密时钟
CPU
65
共用总线 92
通信寄存器
HASH运算
70
缓冲器
原有密钥存储寄存器
20
机密Flash存取控制
95
跳线
44
专用总线
91
机密Flash检测控制
不可改写区域写入旗标
机密Flash
10
14
13
12
11

图 1

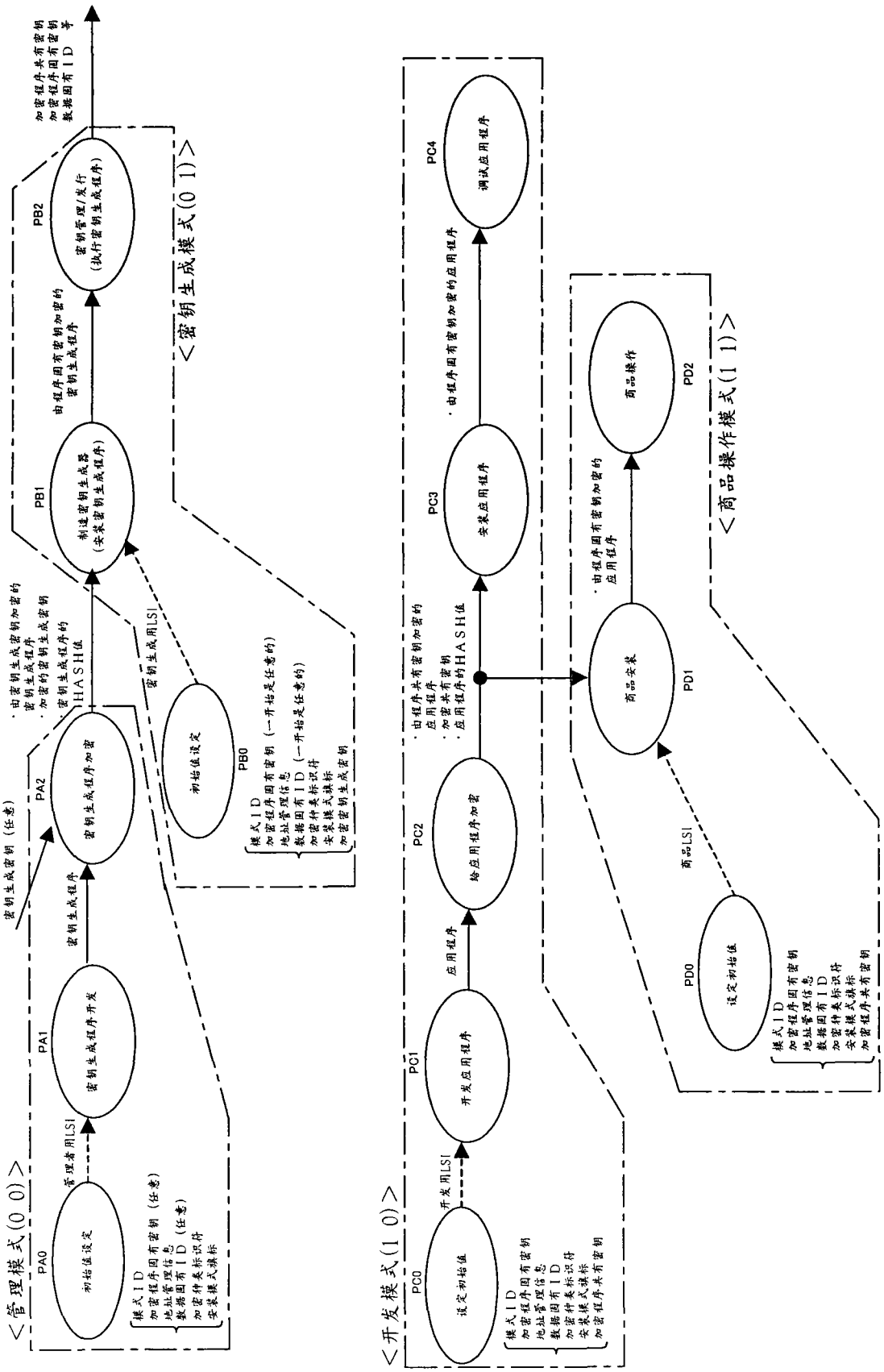

<管理模式(0 0)>
初始值设定
PA0
管理者用LSI
密钥生成程序开发
PA1
密钥生成程序
密钥生成程序加密
PA2
密钥生成用LSI
·由密钥生成密钥加密的密钥生成程序
·加密的密钥生成程序的HASH值
制造密钥生成器(安装密钥生成程序)
PB1
由程序固有密钥加密的密钥生成程序
密钥管理/发行(执行密钥生成程序)
PB2
加密程序共有密钥
加密程序固有ID等
<密钥生成模式(0 1)>
初始值设定
PB0
模式ID
加密程序固有密钥(一开始是任意的)
地址管理信息
数据固有ID(一开始是任意的)
加密种类标识符
安装模式旗标
加密密钥生成密钥
密钥生成用LSI
密钥生成密钥(任意)
模式ID
加密程序固有密钥(任意)
地址管理信息
数据固有ID(任意)
加密种类标识符
安装模式旗标
<开发模式(1 0)>
设定初始值
PC0
开发用LSI
开发应用程序
PC1
应用程序
给应用程序加密
PC2
·由程序共有密钥加密的应用程序
·加密共有密钥
·应用程序的HASH值
安装应用程序
PC3
·由程序固有密钥加密的应用程序
调试应用程序
PC4
模式ID
加密程序固有密钥
地址管理信息
数据固有ID
加密种类标识符
安装模式旗标
加密程序共有密钥
<商品操作模式(1 1)>
设定初始值
PD0
商品LSI
商品安装
PD1
·由程序固有密钥加密的应用程序
商品操作
PD2
模式ID
加密程序固有密钥
地址管理信息
数据固有ID
加密种类标识符
安装模式旗标
加密程序共有密钥

图 2

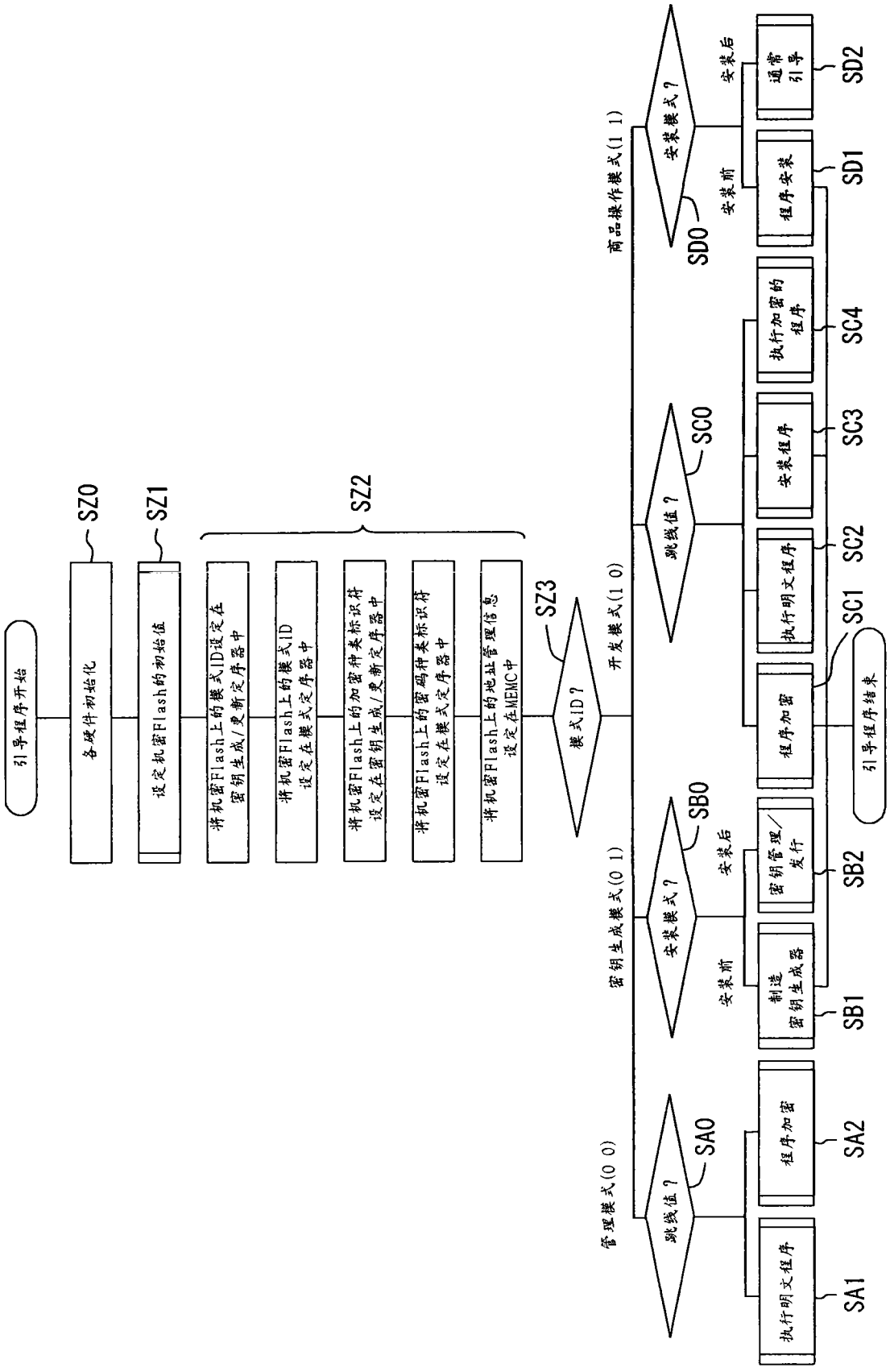

引导程序开始
各硬件初始化
SZ0
设定机密Flash的初始值
SZ1
将机密Flash上的模式ID设定在密钥生成/更新定序器中
将机密Flash上的模式ID设定在模式定序器中
将机密Flash上的加密种类标识符设定在密钥生成/更新定序器中
将机密Flash上的密码种类标识符设定在模式定序器中
将机密Flash上的地址管理信息设定在MEMC中
SZ2
模式ID？
SZ3
管理模式(0 0)
跳线值？
SA0
执行明文程序
SA1
程序加密
SA2
密钥生成模式(0 1)
安装模式？
SB0
安装前
制造密钥生成器
SB1
安装后
密钥管理/发行
SB2
开发模式(1 0)
跳线值？
SC0
程序加密
SC1
执行明文程序
SC2
安装程序
SC3
执行加密的程序
SC4
引导程序结束
商品操作模式(1 1)
安装模式？
SD0
安装前
程序安装
SD1
安装后
通常引导
SD2

图 3

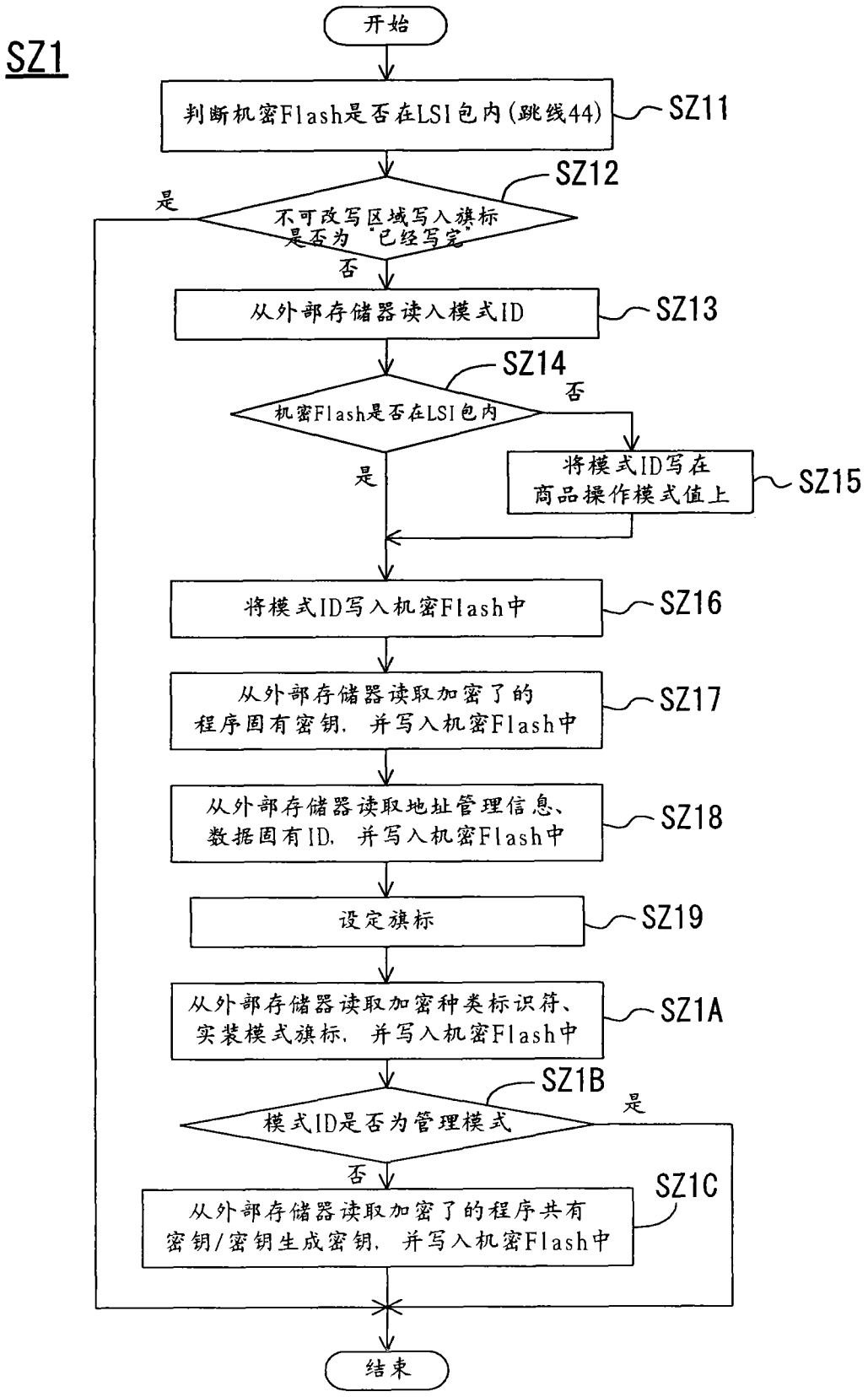

SZ1
开始
判断机密Flash是否在LSI包内(跳线44)
SZ11
SZ12
是
不可改写区域写入旗标
是否为"已经写完"
否
从外部存储器读入模式ID
SZ13
SZ14
否
机密Flash是否在LSI包内
是
将模式ID写在
商品操作模式值上
SZ15
将模式ID写入机密Flash中
SZ16
从外部存储器读取加密了的
程序固有密钥，并写入机密Flash中
SZ17
从外部存储器读取地址管理信息、
数据固有ID，并写入机密Flash中
SZ18
设定旗标
SZ19
从外部存储器读取加密种类标识符、
实装模式旗标，并写入机密Flash中
SZ1A
SZ1B
是
模式ID是否为管理模式
否
SZ1C
从外部存储器读取加密了的程序共有
密钥/密钥生成密钥，并写入机密Flash中
结束

图 4

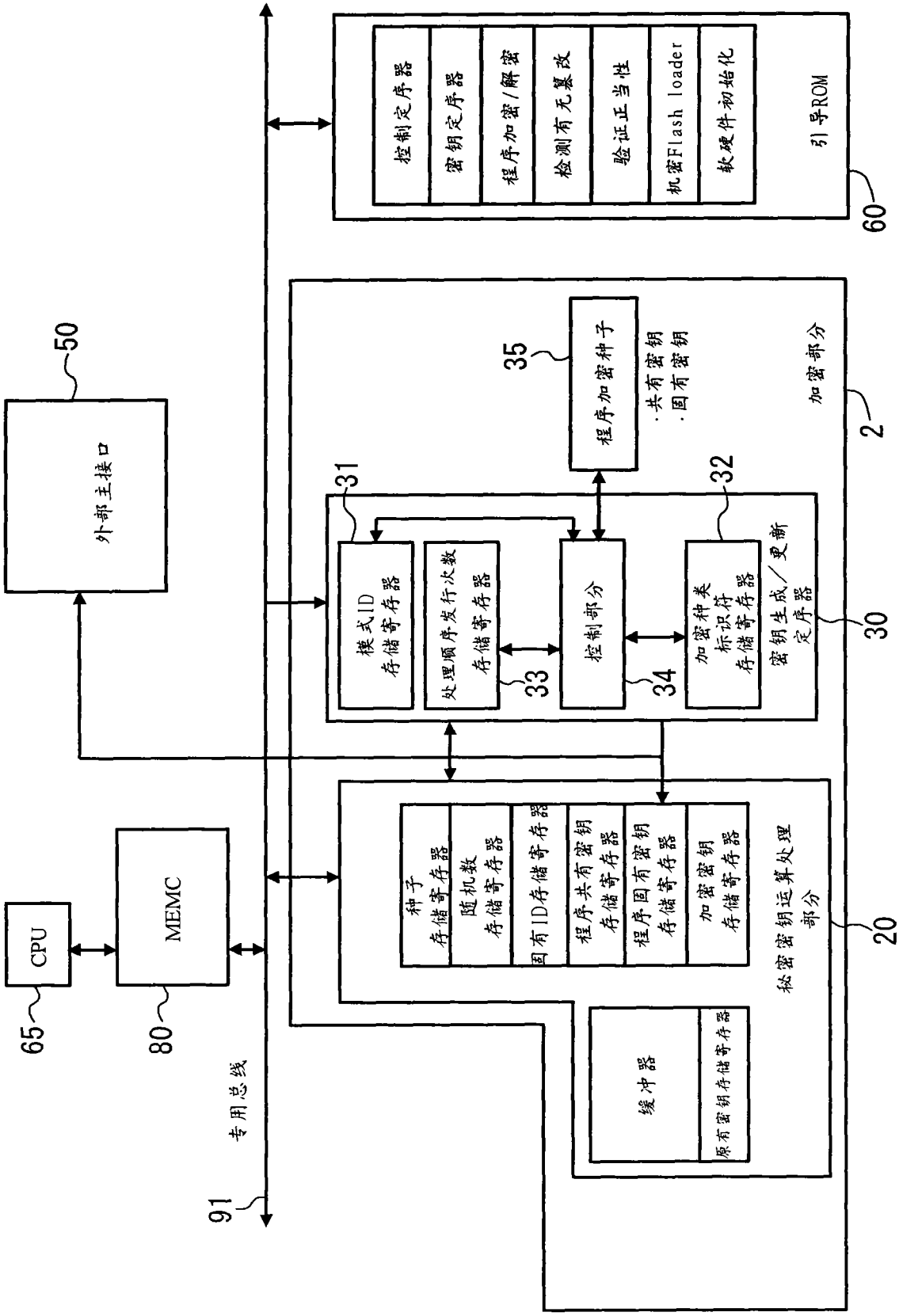

引导ROM
控制定序器
密钥定序器
程序加密/解密
检测有无篡改
验证正当性
机密Flash loader
软硬件初始化
60
加密部分
2
程序加密种子
35
·共有密钥
·固有密钥
模式ID
存储寄存器
31
处理顺序发行次数
存储寄存器
33
控制部分
34
加密种类
标识符
存储寄存器
32
密钥生成/更新
定序器
30
外部主接口
50
种子
存储寄存器
随机数
存储寄存器
固有ID存储寄存器
程序共有密钥
存储寄存器
程序固有密钥
存储寄存器
加密密钥
存储寄存器
秘密密钥运算处理
部分
20
缓冲器
原有密钥存储寄存器
CPU
65
MEMC
80
专用总线
91

图 5

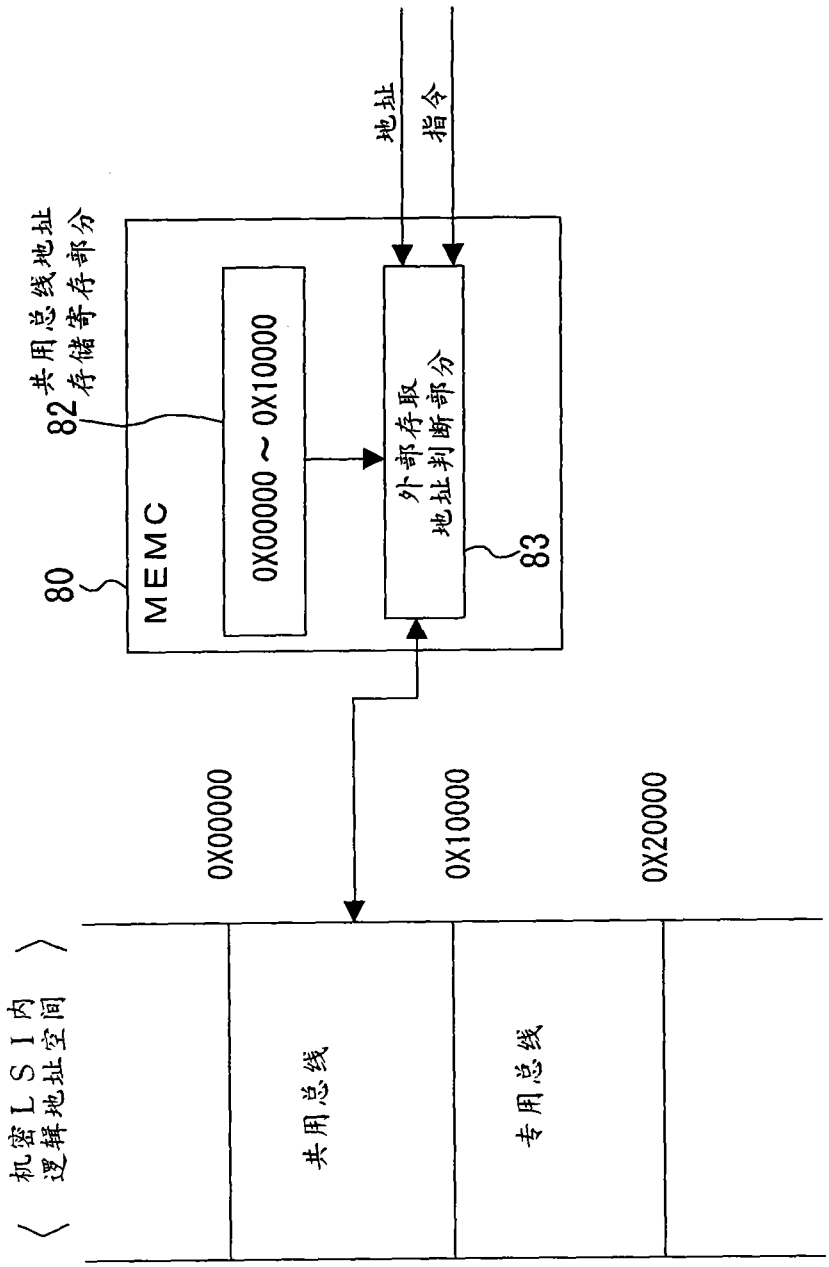

〈 机密LSI内 逻辑地址空间 〉
0X00000
0X10000
0X20000
共用总线
专用总线
80
MEMC
82 共用总线地址 存储寄存部分
0X00000～0X10000
外部存取 地址判断部分
83
地址
指令

图 6

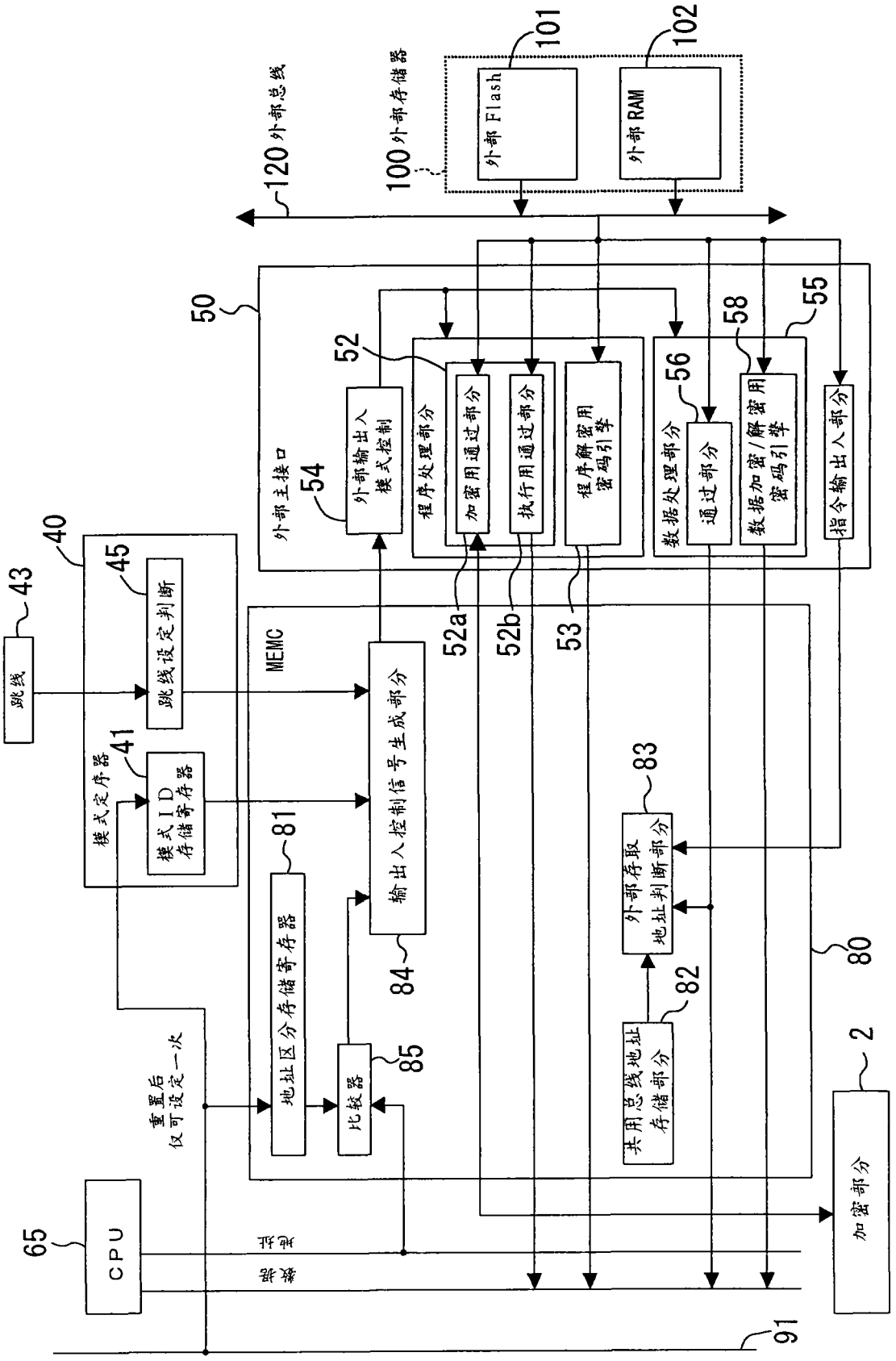

120 外部总线
100 外部存储器
101
外部 Flash
102
外部 RAM
50
外部主接口
52
外部输出入模式控制
54
程序处理部分
加密用通过部分
执行用通过部分
程序解密用密码引擎
数据处理部分
56
通过部分
58
数据加密/解密用密码引擎
55
指令输出入部分
52a
52b
53
40
模式定序器
45
跳线设定判断
43
跳线
41
模式ID存储寄存器
MEMC
81
地址区分存储寄存器
比较器
85
输出入控制信号生成部分
84
83
外部存取地址判断部分
82
共用总线地址存储部分
80
2
加密部分
重置后仅可设定一次
65
CPU
数据
地址
91

图 7

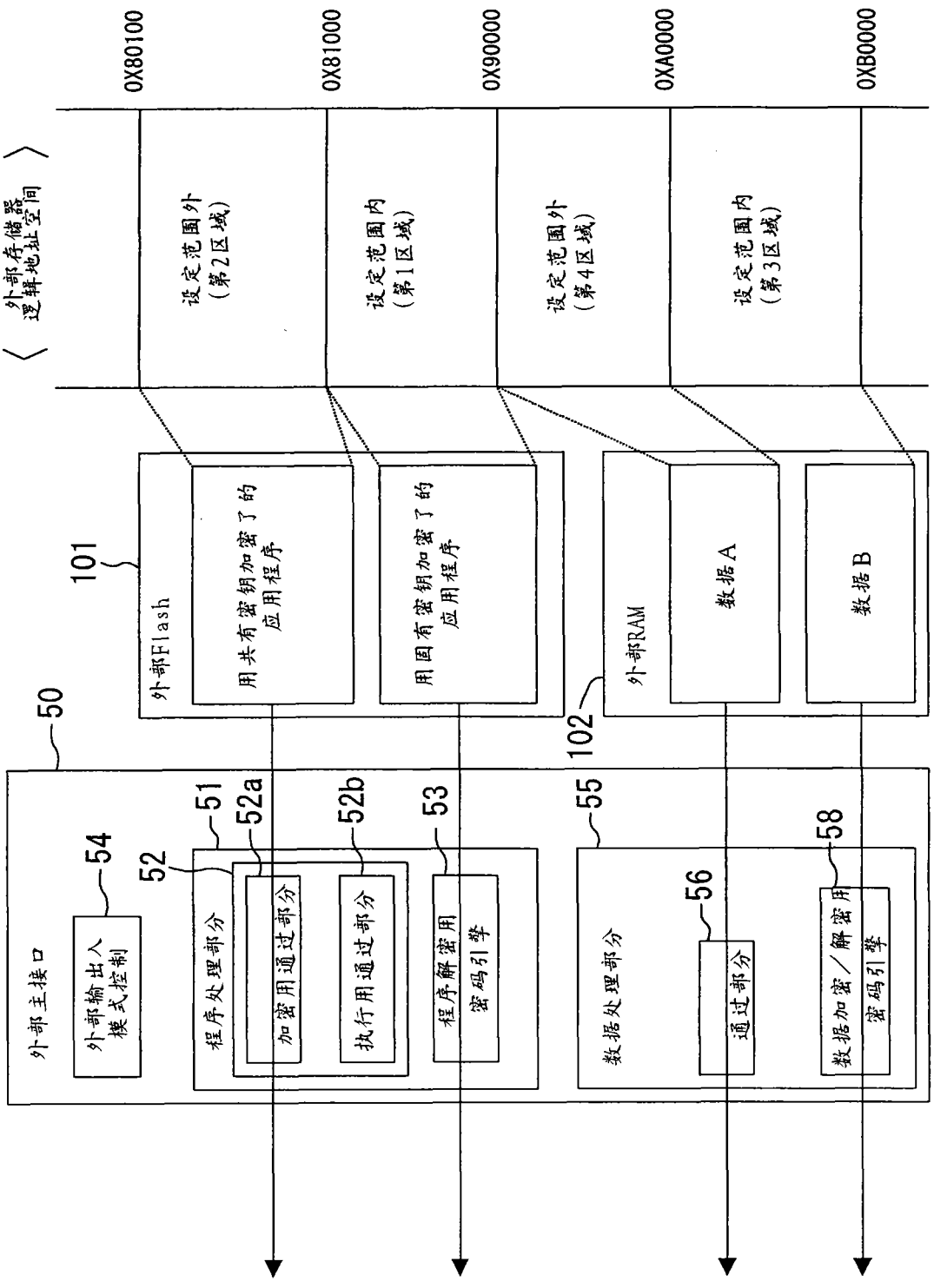

外部存储器
逻辑地址空间
0X80100
0X81000
0X90000
0XA0000
0XB0000
设定范围外
(第2区域)
设定范围内
(第1区域)
设定范围外
(第4区域)
设定范围内
(第3区域)
101
外部Flash
用共有密钥加密了的
应用程序
用固有密钥加密了的
应用程序
102
外部RAM
数据A
数据B
50
外部主接口
54
外部输出入
模式控制
52
程序处理部分
51
52a
加密用通过部分
52b
执行用通过部分
53
程序解密用
密码引擎
55
数据处理部分
56
通过部分
58
数据加密/解密用
密码引擎

图 8

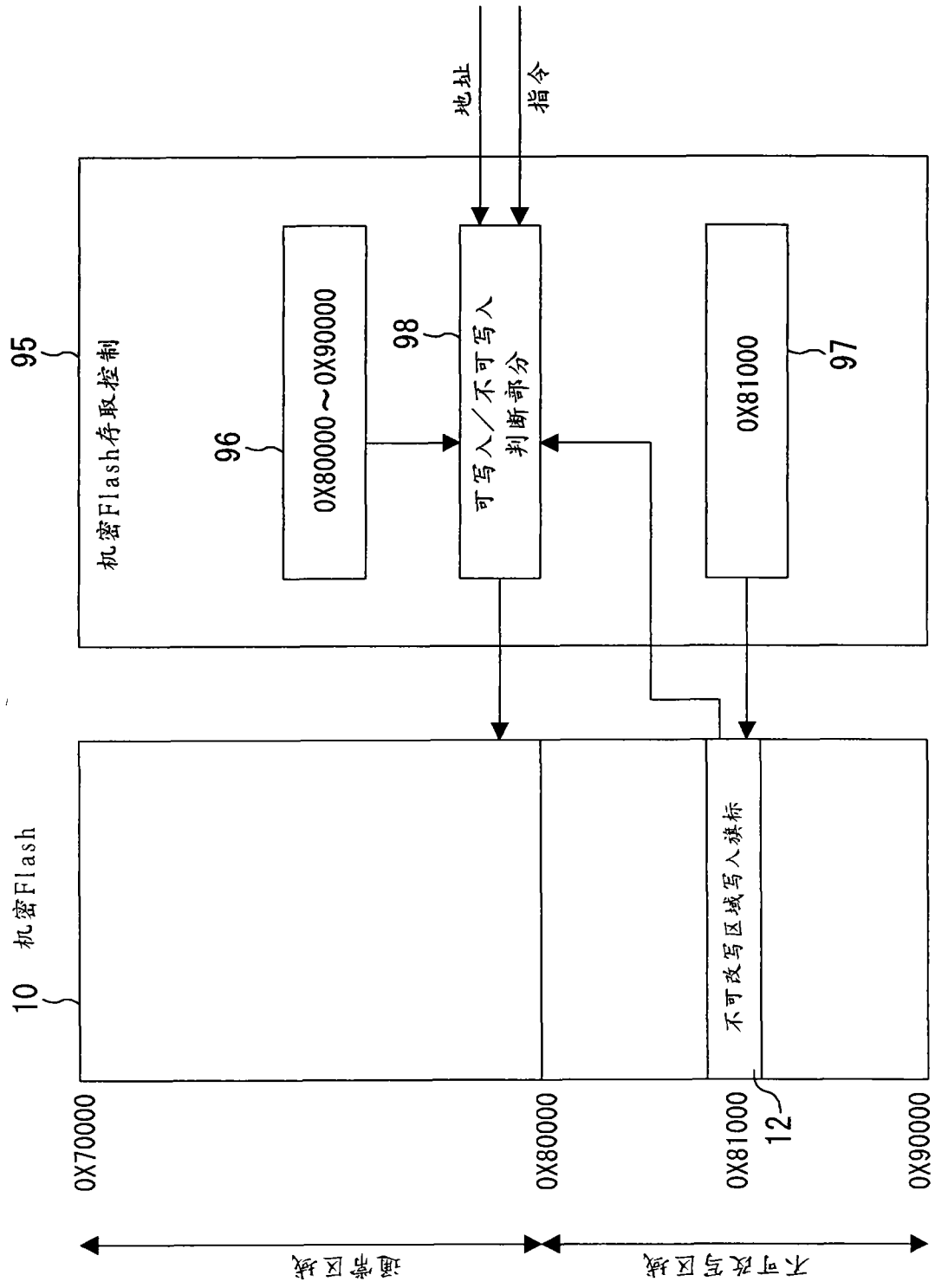

地址
指令
95
机密Flash存取控制
96
0X80000～0X90000
98
可写入／不可写入
判断部分
0X81000
97
10
机密Flash
不可改写区域写入旗标
12
0X70000
0X80000
0X81000
0X90000
通常区域
不可改写区域

图 9

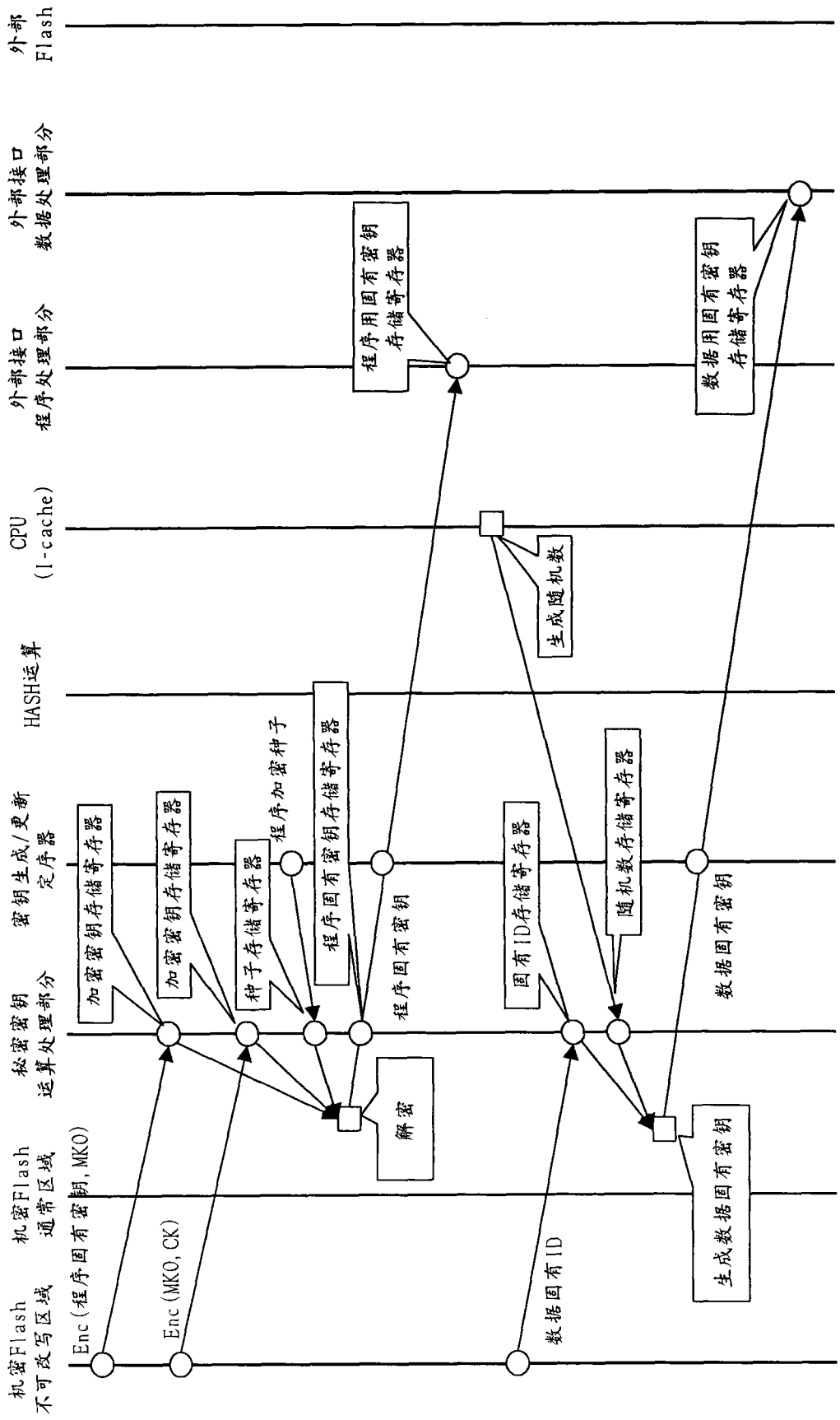

机密Flash
不可改写区域
机密Flash
通常区域
秘密密钥
运算处理部分
密钥生成/更新
定序器
HASH运算
CPU
(I-cache)
外部接口
程序处理部分
外部接口
数据处理部分
外部
Flash
Enc(程序固有密钥,MK0)
Enc(MK0,CK)
加密密钥存储寄存器
加密密钥存储寄存器
种子存储寄存器
程序加密种子
程序固有密钥存储寄存器
解密
程序固有密钥
固有ID存储寄存器
随机数存储寄存器
生成随机数
数据固有ID
生成数据固有密钥
数据固有密钥
程序用固有密钥
存储寄存器
数据用固有密钥
存储寄存器

图 10

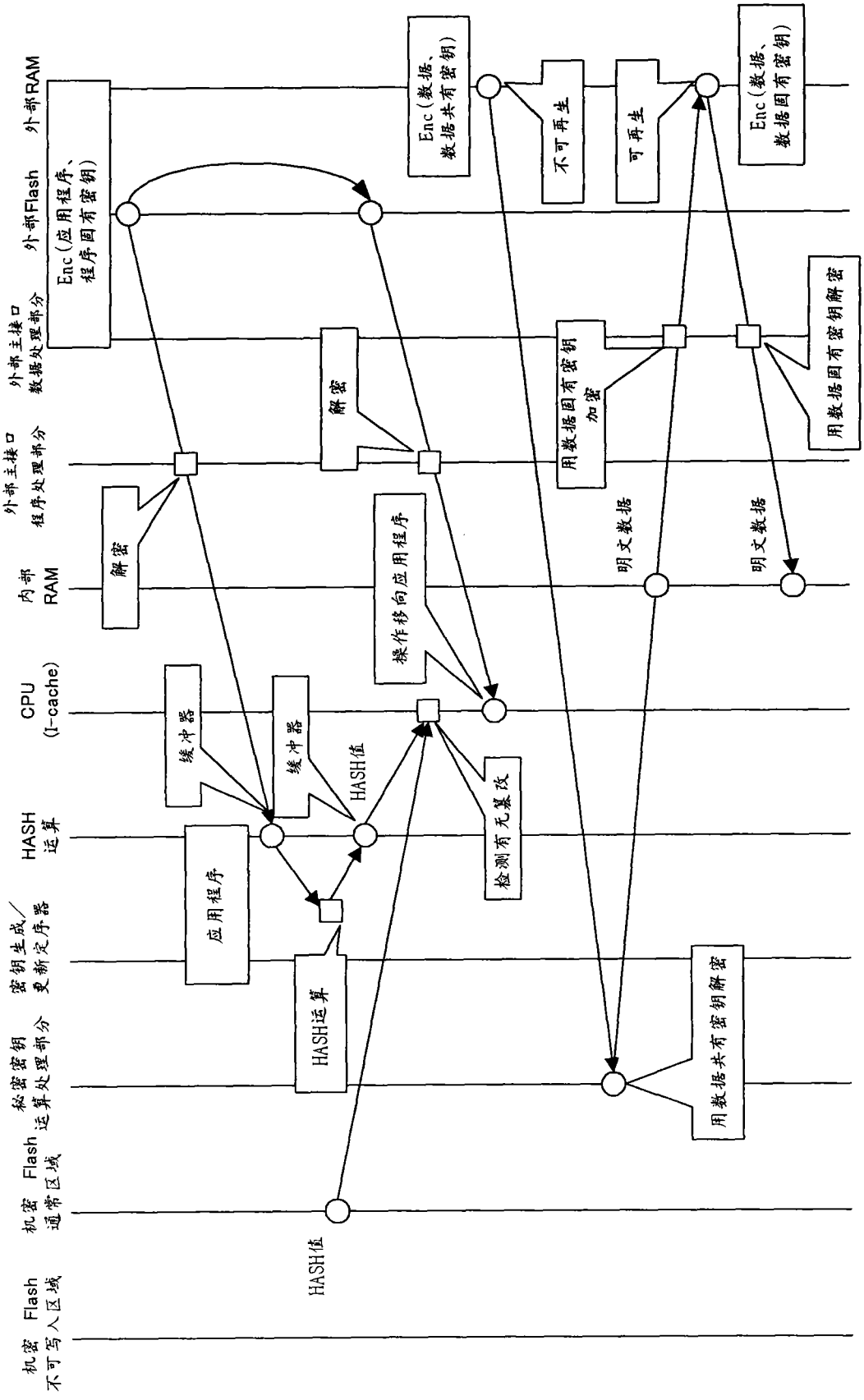

机密 不可写入区域 Flash
机密 通常区域 Flash
秘密密钥 运算处理部分
密钥生成/ 更新定序器
HASH 运算
CPU (I-cache)
内部 RAM
外部主接口 程序处理部分
外部主接口 数据处理部分
外部Flash
外部RAM
Enc(应用程序、程序固有密钥)
解密
缓冲器
缓冲器
应用程序
HASH运算
HASH值
HASH值
操作移向应用程序
检测有无篡改
Enc(数据、数据共有密钥)
不可再生
用数据固有密钥加密
可再生
Enc(数据、数据固有密钥)
用数据固有密钥解密
明文数据
明文数据
用数据共有密钥解密

图 11