(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 102414627 A
(43)申请公布日 2012.04.11

(21)申请号 201080017897.9

(22)申请日 2010.02.17

(30)优先权数据
102009011679.6 2009.02.23 DE

(85)PCT申请进入国家阶段日
2011.10.21

(86)PCT申请的申请数据
PCT/EP2010/000991 2010.02.17

(87)PCT申请的公布数据
WO2010/094466 DE 2010.08.26

(71)申请人 皮尔茨公司
地址 德国奥斯特菲尔登

(72)发明人 彼得·莫斯曼 马蒂亚斯·罗伊施
赫贝特·沃尔特
安德烈亚斯·I·赫克尔

(74)专利代理机构 北京集佳知识产权代理有限公司 11227
代理人 王萍 陈炜

(51)Int.Cl.
G05B 9/03(2006.01)
G05B 19/042(2006.01)
G05B 23/02(2006.01)

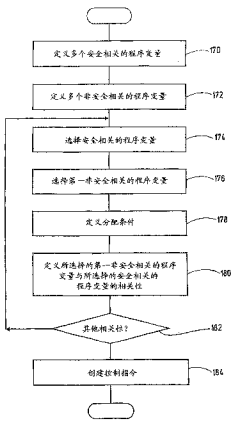
权利要求书 2 页 说明书 14 页 附图 3 页

(54)发明名称

用于创建安全控制装置的应用程序的方法和装置

(57)摘要

本发明涉及一种用于为安全控制装置(20)创建应用程序的方法，该安全控制装置构建为控制具有多个传感器(26)和多个执行器(24)的自动化设备(22)，其中应用程序包括：第一程序部分(74)，在该第一程序部分中故障安全地处理安全相关的程序变量；并且包括至少一个第二程序部分(78)，在该第二程序部分中处理非安全相关的程序变量，其中针对非安全相关的程序变量在第二程序部分(78)内无需故障安全的处理，该方法具有如下步骤：定义多个安全相关的程序变量(46)；定义多个非安全相关的程序变量(48)；从多个安全相关的程序变量(46)中选择安全相关的程序变量(50)；从多个非安全相关的程序变量(48)中选择第一非安全相关的程序变量(52)，其中第一非安全相关的程序变量(52)在实施应用程序时反复地被分配瞬时值；定义至少一个分配条件(54)，其在实施应用程序时被处理；定义相关性(56)，其将所选择的第一非安全相关的程序变量(52)与所选择的安全相关的程序变量(50)相关，其中所选择的第一非安全相关的程序变量(52)的瞬时值在应用程序实施时根据分配条件(54)被分配给所选择的安全相关的程序变量(50)。此外本发明还涉及相应的装置以及相应的计算机程序。



CN 102414627 A

1. 一种用于为安全控制装置 (20) 创建应用程序的方法，该安全控制装置构建为控制具有多个传感器 (26) 和多个执行器 (24) 的自动化设备 (22)，其中应用程序包括：第一程序部分 (74)，在该第一程序部分中故障安全地处理安全相关的程序变量；并且包括第二程序部分 (78)，在该第二程序部分中处理非安全相关的程序变量，其中针对非安全相关的程序变量在第二程序部分 (78) 内无需故障安全的处理，该方法具有如下步骤：

- 定义多个安全相关的程序变量 (46)，
- 定义多个非安全相关的程序变量 (48)，
- 从所述多个安全相关的程序变量 (46) 中选择安全相关的程序变量 (50)，
- 从所述多个非安全相关的程序变量 (48) 中选择第一非安全相关的程序变量 (52)，其中第一非安全相关的程序变量 (52) 在实施应用程序时反复地被分配瞬时值，
- 定义至少一个分配条件 (54)，其在实施应用程序时被处理，
- 定义相关性 (56)，其将所选择的第一非安全相关的程序变量 (52) 与所选择的安全相关的程序变量 (50) 相关，其中所选择的第一非安全相关的程序变量 (52) 的瞬时值在应用程序实施时根据分配条件 (54) 被分配给所选择的安全相关的程序变量 (50)。

2. 根据权利要求 1 所述的方法，其特征在于，分配条件 (54) 代表询问，借助该询问确定所选择的第一非安全相关的程序变量 (52) 是否满足对安全相关的程序变量所要求的可靠性要求。

3. 根据上述权利要求之一所述的方法，其特征在于，似然性询问定义为分配条件 (54)，借助该似然性询问检验所选择的第一非安全相关的程序变量 (52) 和所选择的另外的程序变量是否彼此相一致。

4. 根据上述权利要求之一所述的方法，其特征在于，似然性询问定义为分配条件 (54)，借助似然性询问确定所选择第一非安全相关的程序变量 (52) 是否满足比较标准，其中该比较标准代表所选择的第一非安全相关的程序变量 (52) 的典型特性。

5. 根据上述权利要求之一所述的方法，其特征在于，应用程序包括多个变换指令，其中变换指令代表相关性 (56) 和分配条件 (54)，其中变换指令的至少一部分包含在第一程序部分 (74) 中。

6. 根据上述权利要求之一所述的方法，其特征在于，代表相关性 (56) 的第一代码部分 (200) 和代表分配条件 (54) 的第二代码部分 (202) 组合成程序模块 (76,220)。

7. 根据上述权利要求之一所述的方法，其特征在于，应用程序通过使用在计算机 (12) 上运行的计算机程序 (16) 来创建，其中计算机程序 (16) 包括显示模块 (60)，其中显示模块 (60) 实现在创建应用程序期间使应用程序的源代码 (62) 借助多个代表所述源代码的图形源代码符号 (196) 显示在连接到计算机 (12) 上的显示单元 (14) 上，其中图形源代码符号 (196) 以基本显示形式实施，其中多个图形源代码符号 (196) 包括多个图形变换代码符号 (198)，其代表在源代码 (62) 中包含的变换代码，其中变换代码包括：第一代码部分 (200)，其代表相关性 (56)；并且包括第二代码部分 (202)，其代表分配条件 (54)，其中计算机程序 (16) 还包括识别模块 (68)，借助识别模块识别两个代码部分 (200,202) 的至少其中之一，其中在存在所识别的代码部分时显示模块 (60) 实现以相对于基本显示形式修改过的显示形式显示至少一个图形变换代码符号 (204)，其包含在多个图形变换代码符号 (198) 中。

8. 根据权利要求 7 所述的方法，其特征在于，借助识别模块 (68) 识别第一代码部分

(200)。

9. 根据上述权利要求之一所述的方法，其特征在于，应用程序通过使用在计算机 (12) 上运行的计算机程序 (16) 来创建，其中应用程序的创建通过从多个预先定义的程序模块中选择多个程序模块来进行，其中计算机程序 (16) 包括显示模块 (60)，显示模块 (60) 实现将多个图形程序模块符号 (212) 显示在连接到计算机 (12) 上的显示单元 (14) 上，其中多个图形程序模块符号 (212) 包括 ：第一数目的图形程序模块符号 (214)，其代表预先定义的程序模块 ；和第二数目的图形程序模块符号 (216)，其代表多个所选择的程序模块，其中至少第一数目的图形程序模块符号 (214) 以基本显示形式实施，其中所述多个预先定义的程序模块包括在创建应用程序期间所创建的程序模块 (220)，其代表相关性 (56) 和分配条件 (54)，其中所创建的程序模块 (220) 通过至少一个修改过的图形程序模块符号 (222) 代表，其中显示模块 (60) 实现以相对于基本显示形式修改过的显示形式显示修改过的图形程序模块符号 (222)。

10. 根据上述权利要求之一所述的方法，其特征在于，在实施应用程序时基于所选择的第一非安全相关的程序变量 (52) 产生复制的非安全相关的程序变量。

11. 根据权利要求 10 所述的方法，其特征在于，不仅针对所选择的第一非安全相关的程序变量 (52) 而且针对复制的非安全相关的程序变量处理分配条件 (54)。

12. 根据上述权利要求之一所述的方法，其特征在于，所选择的第一非安全相关的程序变量 (52) 是程序输入变量，其中分配给程序输入变量的瞬时值代表传感器信号 (102) 的值，该传感器信号通过非故障安全地构建的传感器 (104) 产生。

13. 一种用于为安全控制装置 (20) 创建应用程序的装置，安全控制装置 (20) 构建为控制具有多个传感器 (26) 和多个执行器 (24) 的自动化设备 (22)，其中应用程序包括 ：第一程序部分 (74)，在该第一程序部分中故障安全地处理安全相关的程序变量 ；并且包括第二程序部分 (78)，在该第二程序部分中处理非安全相关的程序变量，其中针对非安全相关的程序变量在第二程序部分 (78) 内无需故障安全的处理，

该装置具有如下单元 (12,14,16) ：这些单元用于定义多个安全相关的程序变量 (46) 和从所述多个安全相关的程序变量 (46) 中选择安全相关的程序变量 (50)，用于定义多个非安全相关的程序变量 (48) 和用于从多个非安全相关的程序变量 (48) 中选择第一非安全相关的程序变量 (52)，其中第一非安全相关的程序变量 (52) 在应用程序实施时反复地被分配瞬时值，用于定义至少一个分配条件 (54)，其在应用程序实施时被处理，并且用于定义相关性 (56)，其将所选择的第一非安全相关的程序变量 (52) 与所选择的安全相关的程序变量 (50) 相关，其中所选择的第一非安全相关的程序变量 (52) 的瞬时值在应用程序实施时根据分配条件 (54) 被分配给所选择的安全相关的程序变量 (50)。

14. 一种计算机程序，其具有程序代码单元，用于在该计算机程序 (16) 在计算机 (12) 上实施时执行根据权利要求 1 至 12 之一所述的方法。

用于创建安全控制装置的应用程序的方法和装置

[0001] 本发明涉及一种用于创建安全控制装置的应用程序的方法和装置，其构建为控制具有多个传感器和多个执行器的自动化设备，其中应用程序包括：第一程序部分，在该第一程序部分中故障安全地处理安全相关的程序变量；以及至少一个第二程序部分，在所述至少一个第二程序部分中处理非安全相关的程序变量，其中对于在第二程序部分内的非安全相关的程序变量并不需要安全相关的处理。

[0002] 在本发明的意义下的安全控制装置是如下设备或装置：其接收由传感器提供的输入信号并且由此通过逻辑链接并且可能通过另外的信号处理步骤或数据处理步骤产生输出信号。输出信号然后可以输送给执行器，执行器根据输入信号在受控的设备中引起动作或者反应。

[0003] 这种安全控制装置的一个优选的应用领域是在机械安全领域中监控应急关断按键、双手控制装置、安全门或者光栅。这种传感器用于例如对机器进行防护，其中在工作中从该机器中产生对于人或者材料物品的危险。在打开安全门或者在操作应急关断按键时，分别产生信号，其作为输入信号输送给安全控制装置。响应于此，安全控制装置于是例如借助执行器将机器的产生危险的部分关断。

[0004] 不同于“正常”控制装置，对安全控制装置典型的是，当在安全控制装置中或者与其相连的设备中出现故障时，安全控制设备本身于是始终保证引起危险的设备或者机器的安全状态。因此，在安全控制装置中对于自己的故障安全性提出了极高的要求，这导致在开发和制造时的巨大开销。

[0005] 通常，安全控制装置在其使用之前需要通过主管的监管部门的特别许可，例如在德国通过专业协会或者 **TÜV** 许可。在此，安全控制装置必须遵守预先给定的安全标准，其例如记录在欧洲标准 EN 954-1 或类似的标准例如标准 IEC 61508 或者标准 EN ISO 13849-1 中。因此，在下文中安全控制装置理解为如下设备或者装置：其至少满足所述欧洲标准 EN 954-1 的安全类型 3 或其安全完整性等级 (SIL) 至少达到根据所述标准 IEC61508 的第二级。

[0006] 可编程的安全控制装置为用户提供了如下可能性：逻辑链接以及必要时其他的信号或者数据处理步骤借助软件、所提及的应用程序单独地确定其需求。由此，得到与以前的解决方案相比大的灵活性，在以前的解决方案中，逻辑链接通过在不同的安全模块之间的限定的布线来产生。应用程序例如可以借助市面上可获得的个人计算机 (PC) 并且通过使用相应地设计的软件程序来创建。

[0007] 在根据现有技术的设备中，通常使用两种可编程的控制装置。一种安全控制装置用于解决安全任务，以及一种标准控制装置用于解决标准任务。个别也可以使用共同的控制装置，借助其解决所有标准任务和安全任务。在两种实施形式中，安全任务的解决通过安全相关的程序变量的故障安全的处理来实现。为此，通过安全传感器（其为故障安全地构建的传感器）来检测安全相关的量并且借助安全相关的控制输入信号输送给安全控制装置或者共同的控制装置。在该控制装置中通过使用安全相关的程序变量确定安全相关的控制输出信号的值。借助这些控制输出信号来激励安全执行器（其为故障安全地构建的执行

器）用于实施安全相关的动作。标准任务的解决通过处理非安全相关的程序变量来实现，对于这些程序变量无需故障安全的处理。为此，通过标准传感器检测非安全相关的量，其也称作过程相关的量。借助非安全相关的控制输入信号将这些量输送给标准控制装置或共同的控制装置。在该控制装置中通过使用非安全相关的程序变量确定非安全相关的控制输出信号的值。借助这些控制输出信号激励标准执行器，其于是实施非安全相关的动作。

[0008] 在两种实现形式中，对于安全任务的解决需要使用因故障安全地构建而昂贵的传感器。为此不能使用因非故障安全地构建而廉价的标准传感器也不能使用非安全相关的程序变量的值。

[0009] 不仅使用可编程的安全控制装置而且使用共同的控制装置来解决所有标准和安全任务会有助于提高在创建应用程序时的灵活性并且因此提高了在实现安全控制装置时的灵活性。然而其在成本（用于创建应用程序的成本和对实现安全控制装置所需的部件的成本）方面还未最优。

[0010] 因此，本发明的任务是改进开头所述的类型的方法和装置，以便进一步减小在创建具有安全相关的功能的应用程序时的成本并且因此进一步减小实现安全控制装置时的成本，并且提高了灵活性以便因此能够实现对安全控制装置进行更为简单、更为快速并且更为一目了然的编程。

[0011] 该任务通过开头所述的方式的方法来解决，其中实施以下步骤：

[0012] - 定义多个安全相关的程序变量，

[0013] - 定义多个非安全相关的程序变量，

[0014] - 从多个安全相关的程序变量中选择安全相关的程序变量，

[0015] - 从多个非安全相关的程序变量中选择第一非安全相关的程序变量，其中第一非安全相关的程序变量在实施应用程序时反复地被分配瞬时值，

[0016] - 定义至少一个分配条件，其在实施应用程序时被处理，

[0017] - 定义相关性，其将所选择的第一非安全相关的程序变量与所选择的安全相关的程序变量相关，其中所选择的第一非安全相关的程序变量的瞬时值在应用程序实施时根据分配条件被分配给所选择的安全相关的程序变量。

[0018] 此外，该任务通过开头所述类型的装置来解决，其具有如下单元：这些单元用于定义多个安全相关的程序变量和从多个安全相关的程序变量中选择安全相关的程序变量，用于定义多个非安全相关的程序变量和用于从多个非安全相关的程序变量中选择第一非安全相关的程序变量，其中第一非安全相关的程序变量在应用程序实施时被反复地分配瞬时值，用于定义至少一个分配条件，其在应用程序实施时被处理，并且用于定义相关性，其将所选择的第一非安全相关的程序变量与所选择的安全相关的程序变量相关，其中所选择的第一非安全相关的程序变量的瞬时值在应用程序实施时根据分配条件被分配给所选择的安全相关的程序变量。

[0019] 新的方法和新的装置所基于的思想是，在应用程序实施时根据分配条件为所选择的安全相关的程序变量分配所选择的第一非安全相关的程序变量的瞬时值，并且因此一般而言，将非安全相关的程序变量转换成安全相关的程序变量。由此，提高了在创建应用程序时的灵活性。为了解决安全任务，除了安全相关的程序变量和通过其来代表的安全相关的量之外现在也提供了非安全相关的程序变量以及通过其来代表的非安全相关的量。因此，

针对各安全任务提高了编程技术上可实现的解决方案的数目。

[0020] 由应用程序的创建者可任意定义所需的分配条件也有利于提高灵活性。因此，不同的非安全相关的程序变量和通过其来代表的物理量可以用于转换。

[0021] 为所选择的安全相关的程序变量分配所选择的第一非安全相关的程序变量的瞬时值引起：对于提供安全相关的程序变量的瞬时值无需故障安全地构建的传感器。代表安全相关的程序变量的物理量因此也可以通过使用非故障安全的传感器来检测。这有助于减小成本。

[0022] 对于将非安全相关的程序变量转换成安全相关的程序变量基本上仅定义分配条件和相关性。这能够实现清楚、简单并且因此快速地对安全控制装置编程，这也有助于提高故障安全性。

[0023] 因此上述任务被完全解决。

[0024] 在本发明的另一扩展方案中，分配条件代表询问，借助该询问确定所选择的第一非安全相关的程序变量是否满足对安全相关的程序变量所需要的可靠性要求。

[0025] 该措施保证了，尽管非安全相关的程序变量的瞬时值递送给安全相关的程序变量但安全控制装置可以故障安全地运行。为了解决安全任务，除了安全相关的程序变量和由其代表的物理量之外因此也可以使用非安全相关的程序变量和由其代表的量。此外，为了提供故障安全性，只有在满足分配条件时才为所选择的安全相关的程序变量分配所选择的第一非安全相关的程序变量的值。

[0026] 关于技术系统，可靠性是在限定的概率边界和时间段内连续正确产生议定功率方面的可靠度。可靠性因此是置信度，其由于低故障概率、高正确性概率或低失效率或故障率而显得有根据。

[0027] 对于安全相关的程序变量所需的可靠性要求首先定义了其瞬时值的正确性概率。可靠性要求例如可以规定：安全相关的程序变量的瞬时值至多带有小到可忽略的偏差地与实际存在的物理量的值相一致。通常，为了满足可靠性要求而使用故障安全地构建的传感器。因此，在检测物理量时可以识别故障并且可以遵守所需的正确性概率。所需要的可靠性要求满足与否有利地通过分析所选择的第一非安全相关的程序变量的瞬时值来确定。

[0028] 在本发明的另一扩展方案中，作为分配条件定义了似然性询问，借助其检验所选择的第一非安全相关的程序变量和所选择的另一程序变量是否彼此一致。

[0029] 该措施能够实现高灵活性，因为另一程序变量的任意选择为定义似然性询问开创了大的余地。此外，似然性询问可以通过使用两个程序变量而没有大开销地实现。

[0030] 通过检验两个程序变量是否彼此相一致来确定，两个程序变量是否彼此符合。换言之，确定两个程序变量是否彼此协调。该检验基于如下假设：在理想情况下不仅所选择的第一非安全相关的程序变量而且所选择的其他程序变量代表同一物理量，例如速度或距离。然而针对两个程序变量代表不同但通过物理规律相关的物理量的情况，也可以执行该校验。然而在此情况下，针对两个程序变量之一需要在考虑物理规律的情况下换算瞬时值。

[0031] 有利地，该校验通过两个程序变量的瞬时值的分析来进行。

[0032] 在分析瞬时值时检验两个程序变量的瞬时值是否满足所定义的标准。为此，原则上可以考虑许多目的不同的标准。一方面可以使用针对瞬时值本身的标准。在此情况下检验瞬时值本身是否显现一致的特性。例如可以检验由两个瞬时值形成的差是否小于所定义

阈值或由两个瞬时值形成的商是否在定义的区间中，该区间设置在值 1 左右。另一方面，可以使用针对瞬时值的时间特性的标准。因此，检验瞬时值是否显现一致的时间特性。在此情况下，针对两个程序变量确定时间导数，有利地近似以差分商的形式确定。在该标准的情况下，也可以如前面所描述的那样相应地分析差或商。使用针对时间特性的标准具有如下优点：在非常早的时刻可以确定两个程序变量之间的偏差，远在瞬时值中的相当大的偏差本身能够被觉察之前。有利地，两个程序变量的用于形成差或商的瞬时值分别成对地在确定的时刻在与该时刻相关的小的时间区间之前或内。

[0033] 有利地，多个标准彼此组合。由此，可以创建非常可靠的似然性询问，这使安全性高。

[0034] 优选地，所选择的其他程序变量是非安全相关的程序变量。在该情况下，安全相关的程序变量仅通过使用非安全相关的程序变量来生成，这使灵活性非常大。

[0035] 在前面所述的措施的另一扩展方案中，不仅所选择的第一非安全相关的程序变量而且所选择的另外的非安全相关的程序变量分别是如下程序变量，其中所选择的第一非安全相关的程序变量被分配瞬时值，该瞬时值代表通过第一传感器确定的第一传感器信号的值，并且其中所选择的另外的非安全相关的程序变量被分配如下瞬时值，其代表通过第二传感器确定的第二传感器信号的值。因此，可以以简单的方式和方法通过使用两个非故障安全地构建的传感器将非安全相关的程序变量变换成安全相关的程序变量。

[0036] 在前面所述的措施的另一扩展方案中，两个传感器有利地多样实施。借助两个传感器因此检测同一物理量，而借助不同的测量原理。例如，物理量可以借助其中一个传感器基于电压地检测而借助另一传感器基于电流地检测。该措施保证了非常高程度的安全性。

[0037] 通过合适地选择或组合两个传感器还可以补偿外部影响譬如温度漂移等等。

[0038] 可替选地，作为所选择的另外的程序变量也可以使用安全相关的程序变量，这有助于灵活性的进一步提高。

[0039] 在本发明的另一扩展方案中，似然性询问定义为分配条件，借助该似然性循环检验所选择的第一非安全相关的程序变量是否满足比较标准，其中比较标准代表所选择的第一非安全相关的程序变量的典型特性。

[0040] 该措施具有如下优点：确定是否满足可靠性要求可以仅借助所选择的第一非安全相关的程序变量来实施。另外的程序变量并不需要。该措施能够实现大灵活性和简单实施。针对所选择的第一非安全相关的程序变量为程序输入变量的情况，因此对于将非安全相关的程序变量转换成安全相关的程序变量无需另外的传感器。这能够实现成本非常低廉的转换。

[0041] 总之，可以考虑不同的比较标准。一方面，可以使用针对瞬时值本身的比较标准。例如，涉及所选择的第一非安全相关的程序变量的瞬时值如所期望地必须在其内的区间。该区间可以通过通常期望的最大值和通常期望的最小值来预先给定。另一方面，可以使用针对瞬时值的时间特性的比较标准。例如，涉及对于增加的瞬时值所确定的时间梯度如所期望地必须在其内的区间。其相应地可以用于降低的瞬时值。

[0042] 有利地，比较标准由多个单询问组成。这样，比较标准例如可以不仅包含对瞬时值的预给定而且包含对瞬时值的时间特性的预给定。因此，可以检验瞬时值是否从第一值水平开始在限定的时间段内具有第二值水平并且在该时间段之后存在的瞬时值是否在限定

的区间中。相应内容对于减小的瞬时值也是可能的。比较标准也可以针对首先增加并且随后减小的瞬时值来定义。

[0043] 在本发明的另一扩展方案中，应用程序包括多个变换指令，其中变换指令代表相关性和分配条件，其中变换指令的至少一部分包含在第一存储部分中。

[0044] 变换指令因此包含在其中故障安全地处理安全相关的程序变量并且包含用于激励安全执行器的安全控制装置指令的程序部分中。因此在将非安全相关的程序变量转换成安全相关的程序变量时也提供了安全性。这意味着，由转换引起的安全相关的程序变量是故障安全的，而非安全相关的程序变量始终是非故障安全的。有利地，所有变换指令包含在第一程序部分中。

[0045] 在本发明的另一扩展方案中，代表相关性的第一代码部分和代表分配条件的第二代码部分组合成程序模块。

[0046] 该措施涉及应用程序的源代码。因此，源代码中的两个代码部分组合成一块。因此，在源代码中存在专用的变换位置，在该变换位置中组合变换指令。不仅相关性而且分配条件对于故降安全性是重要的并且因此在监管部门要实施的许可程序的范围中一同被检验。在源代码内的块形成简化了许可程序。优选地，程序模块包含在第一程序部分中。

[0047] 专用的变换位置使应用程序的创建者能够在没有大开销的情况下提供对非安全相关的程序变量在何处用于生成安全相关的程序变量或者哪个安全相关的程序变量归因于非安全相关的程序变量的概况。如果在应用程序中设置多个转换，即多个非安全相关的程序变量分别被转换成安全相关的程序变量，则应用程序包含多个独立程序模块。针对每个转换程序存在专用的转换位置。

[0048] 另一优点在于，组合成程序模块能够实现封装。封装导致程序模块在函数的意义下通过相应地调用而可以任意频繁地再使用。

[0049] 两个代码部分组合成程序模块有利地自动通过计算机程序来进行，通过该计算机程序的使用创建了应用程序。

[0050] 在本发明的另一扩展方案中，应用程序通过使用在计算机上运行的计算机程序来创建，其中计算机程序包括显示模块，其中显示模块实现在创建应用程序期间借助多个代表源代码的图形源代码符号将应用程序的源代码显示在连接到计算机的显示单元上，其中图形源代码模块以基本显示形式实施，其中多个图像源代码符号包括多个图形变换代码符号，其代表在源代码中包含的变换代码，其中变换代码包括第一代码部分，该第一代码部分代表相关性；和第二代码部分，其代表分配条件，其中计算机程序还包括识别模块，借助其识别两个代码部分的至少一个，其中在存在所识别的代码部分时显示模块实现以相对于基本显示形式修改过的显示形式显示包含在多个图形变换代码符号中的至少一个图形变换代码符号。

[0051] 修改过的图形变换符号的显示实现了在显示应用程序中的标记。因此，应用程序的创建者可以毫无问题地定位在应用程序中包含的代表变换代码的范围，即专用的变换位置。应用程序的由非安全相关的程序变量转换成安全相关的程序变量引起的范围因此可以直接识别。创建者因此可以在需要时可以对其进行关键的检验，这在许可程序方面也是有利的。

[0052] 修改过的图形变换符号的显示并且因此在应用程序的显示中的标记自动通过以

下计算机程序来进行，通过使用该计算机程序创建应用程序。应用程序的创建者不必进行附加处理，这些处理超出了通常对于创建应用程序而要进行的处理。

[0053] 有利地，代表第一代码部分的图形变换符号以修改过的显示形式显示。补充地，代表第二代码部分的图形变换符号也可以以修改过的显示形式显示。

[0054] 可考虑多个扩展方案，借助多个图形符号显示应用程序的源代码。在第一扩展方案中，应用程序借助文本输入来创建。应用程序因此例如以结构化的文本形式存在。图形源代码符号在此情况下对应于各字母、数字和必要时特殊符号。在该扩展方案中可以考虑的是，以修改过的方式显示代表第一代码部分的字母中的至少一些。这例如可以通过如下方式实现，字母以相对于基本显示形式改变的色彩来显示。可替选地或补充地，这些字母也可以以相对于基本显示形式改变的字体突出方式例如斜体或加粗来显示。也可以考虑的是，字母例如缩排或者包含第一字母的行前置相应的记号。这样例如可以前置星形或特殊文本譬如“出现安全相关的转换代码”。标记的前置通过添加另外的行来实现。补充地，相应的记号也可以添加在包含最后字母的行之后。

[0055] 在一个优选的扩展方案中，借助识别模块来识别第一代码部分。

[0056] 第一代码部分的识别以及与此联系的代码部分的标记具有如下优点：参与转换的所选择的第一非安全相关的程序变量和所选择的安全相关的程序变量在应用程序的显示中可以被快速地确定。应用程序的创建者因此可以全面地获得信息。其立刻识别哪个程序变量是转换的基础并且哪个程序变量通过转换生成。

[0057] 有利地，第一代码部分于是被识别，使得在相关的代码部分中非安全相关的程序变量被读取并且在此所读取的瞬时值分配给安全相关的程序变量。

[0058] 通过第一代码部分的识别而已知参与转换的程序变量本身。这能够在应用程序的创建者的全面支持的意义下实现标记应用程序的附加范围。优选地，也可以标记代表分配条件的范围。这实现了在显示应用程序中的更为显眼的标记。

[0059] 在一个特别有利的扩展方案中两个代码部分被识别。在该情况下，至少一个图形变换代码符号在存在两个代码部分时以相对于基本显示形式修改过的显示形式来显示。此外，也适于以修改过的显示形式显示所有变换代码符号。

[0060] 关于应用程序的创建以及伴随其的应用程序的显示，可以考虑另外的扩展方案。在此情况下，应用程序通过使用图形符号以所谓的功能块图方式创建。在该扩展方案中，使用了不同类型的图形符号，例如矩形（其代表所选择的功能块）、字母以及占位符。在该扩展方案中，可用的并且因此可选择的程序模块存储在数据库中。针对程序模块的每个在工具栏中设置小的占位符。如果这样的占位符借助拖放功能（Drag&Drop Function）来选择并且放置在图形表面的为此设置的区域中，则在图形表面的相应位置上显示功能块，该功能块代表所选择的程序模块。在数据块中也可以容纳在创建应用程序期间所创建的程序模块。例如代表相关性和分配条件的程序模块。

[0061] 因此，在本发明的另一扩展方案中应用程序通过使用在计算机上运行的计算机程序来创建，其中应用程序的创建通过从多个预先定义的程序模块中选择多个程序模块来进行，其中计算机程序包括显示模块，其实现将多个图形程序模块符号显示在连接到计算机上的显示单元上，其中多个图形程序模块符号包括第一数目的图形程序模块符号，其代表预先定义的程序模块；和第二数目的图形程序模块符号，其代表多个所选择的程序模块，其

中至少第一数目的图形程序模块符号以基本显示形式实施，其中多个预先定义的程序模块包括在创建应用程序期间创建的程序模块，该程序模块代表相关性和分配条件，其中所创建的程序模型通过至少一个修改过的图形程序模块符号来代表，其中显示模块实现修改过的图形程序模块符号以相对于基本显示形式修改过的显示形式显示。

[0062] 通过该措施，向应用程序的创建者可靠地指明应用程序的如下范围，该范围归因于转换的范围并且与应用程序的代表实际安全任务的范围并行地在故障安全性方面是重要的。其中涉及转换的范围要被特别地观察，因为这里在第一程序部分中使用了程序变量，其归因于非安全相关的程序变量。就修改过的图形程序模块符号的范围而言，则可考虑的是，仅修改通过其所代表的如下程序模块的占位符，该程序模块代表相关性和分配条件。补充地，所有在应用程序中创建的代表该程序模块的功能块同样可以被修改。有利地，在该扩展方案中第一代码部分也被识别。

[0063] 在所创建的程序模块被识别使得其不仅包括第一代码部分而且包括第二代码部分时，有利地使用修改过的图形程序模块符号。通过使用修改过的图形程序模块符号，使应用程序的创建者注意到潜在的变换部位。如果所有在应用程序中创建的代表所述的程序模块的功能块被修改，则这实现具体的变换位置的标记。在借助文本输入创建应用程序时，标记也使具体的变换位置突出。

[0064] 在本发明的另一扩展方案中，在实施应用程序时基于所选择的第一非安全相关的程序变量产生复制的非安全相关的程序变量。

[0065] 该扩展方案具有如下优点：为对于安全相关的程序变量在安全控制装置中的故障安全的处理必须存在的两个处理通道的每个提供了独立的非安全相关的程序变量，其分别独立地被转换成安全相关的程序变量。

[0066] 复制的非安全相关的程序变量有利地通过如下方式产生，使得所选择的第一非安全相关的程序变量的瞬时值被读入到两个独立的存储区域中，其中一个针对非安全相关的程序变量设置而一个针对复制的非安全相关的程序变量设置。

[0067] 有利地，复制的非安全相关的程序变量自动地通过计算机程序产生，该计算机程序用于创建应用程序。应用程序的创建者除了通常对于应用程序的创建者要进行的处理之外不需附加的处理。

[0068] 复制的非安全相关的程序变量的自动产生有利地如下运行：借助显示模块创建应用程序的源代码。在存在所识别的代码部分时，识别模块产生复制代码，其被添加到源代码中。基于该复制代码于是在实施应用程序时自动地基于所选择的第一非安全相关的程序变量产生复制的非安全相关的程序变量。但也可以考虑如下实施形式，其中不需要复制代码的明确创建。在该情况下，在安全控制装置上运行的固件中存储有例程，其自动地实施复制或在基于相关性要在第一程序部分中处理非安全相关的程序变量时存在对此所需的指令。

[0069] 在前面所述的措施的另一扩展方案中，不仅针对所选择的第一非安全相关的程序变量而且针对复制的非安全相关的程序变量处理分配条件。

[0070] 因此确定如下的顺序：在第一步骤中产生复制的非安全相关的程序变量。在第二步骤中于是针对两个程序变量检验是否分别满足对于安全相关的程序变量所需要的可靠性要求。通过该处理顺序可以识别在复制时出现的故障，这有助于故障安全性的提高。

[0071] 在本发明的另一扩展方案中，所选择的第一非安全相关的程序变量是程序输入变

量，其中分配给程序输入变量的瞬时值代表通过非故降安全地构建的传感器产生的传感器信号的值。

[0072] 该措施具有如下优点：安全任务可以通过使用成本低廉、非故障安全地构建的传感器来解决。故障安全地构建的昂贵传感器的使用并不一定是必需的。这能够实现安全控制装置的成本低廉的实现。

[0073] 优选地也通过使用非故障安全地构建的传感器来提供所选择的另外的程序变量的瞬时值。

[0074] 优选地，非故障安全地构建的传感器是借助其检测的物理量的传感器，其在由安全控制装置要解决的标准任务的范围中变得必需。这具有如下优点：除了对于解决标准任务变得必需的传感器之外不必使用另外的传感器。安全控制装置因此可以成本低廉地实现。

[0075] 如已经在前面所介绍的那样，不仅所选择的第一非安全相关的程序变量而且所选择的另外的程序变量分别是程序输入变量。可替选地，这两个程序变量也可以分别是程序中间变量。程序中间变量的瞬时值例如根据程序变量的瞬时值来确定。但两个程序变量也可以是程序输出变量。程序输出变量的瞬时值例如根据程序变量的瞬时值或程序中间变量的瞬时值来确定并且代表控制输出信号的值，借助其来激励执行器。所选择的第一非安全相关的程序变量和所选择的另外的程序变量不一定必须是相同类型的程序变量。也可以存在可考虑的组合之一。

[0076] 应理解的是，前面所述的并且在下面还要阐述的特征不仅可以以分别所说明的组合而且也可以以其他组合或单独地使用，而未离开本发明的范围。

[0077] 有利地，所选择的安全相关的程序变量可以用作中间量，根据其例如确定了安全相关的控制输出信号。

[0078] 在前面和下文中的实施中术语应用程序理解为，应用程序包括并且因此代表源代码以及机器代码。或换言之，应用程序通过源代码和机器代码来显示。

[0079] 此外应提及的是，术语非安全相关的和过程相关的在上文和在下文中同义地使用。

[0080] 本发明的实施例在附图中示出并且在下面的描述中进一步阐述。其中：

[0081] 图 1 示出了结合安全控制装置的新的装置的示意图，其中要为该安全控制装置创建应用程序，

[0082] 图 2 示出了用于阐述新方法的简化流程图；

[0083] 图 3 示出了用于创建应用程序的第一图形界面的简化视图；以及

[0084] 图 4 示出了用于创建应用程序的第二图形界面的简化视图

[0085] 在图 1 中根据本发明的装置在整体上用附图标记 10 表示。

[0086] 装置 10 包含带有显示单元 14 的传统的计算机 12，在计算机上执行计算机程序 16。计算机程序 16 能够实现为安全控制装置创建应用程序。其在专业术语中因此通常也称为编程工具。计算机 12 可以实施为 PC 而显示单元 14 可以实施为监视器。

[0087] 在图 1 中示出了在其整体上以附图标记 18 表示的安全电路，其具有安全控制装置 20，其构建为控制在其整体上用附图标记 22 表示的自动化设备。该自动化设备 22 包括多个执行器 24 和多个传感器 26。示例性地示出了在设备 22 中包含的负载 28，其例如是机器

人。

[0088] 安全控制装置 20 双通道冗余地构建，以便实现用于控制安全关键的过程的所需的故障安全性。替代双通道结构，图 1 中示出了两个彼此分离的处理器即第一处理器 30 和第二处理器 32。两个处理器 30、32 通过双向通信接口 34 彼此连接，以便可以彼此控制并且交换数据。优选地，安全控制装置 20 的两个通道和两个处理器 30、32 是不同的，即彼此不同地构建，以便在很大程度上排除系统性故障。

[0089] 借助附图标记 36 表示输入 / 输出单元，其与两个处理器 30、32 的每个连接。输入 / 输出单元 36 从多个传感器 26 接收多个控制输入信号 38 并且将这些控制输入信号以匹配的数据格式转发给两个处理器 30、32 的每个。在假设控制输入信号是模拟信号的情况下，控制输入信号的模拟值通过 A/D 转换被转换成数字值，其于是作为瞬时值分配给程序输入变量。此外，输入 / 输出单元 36 根据处理器 30、32 产生多个控制输出信号 40，借助其来激励多个执行器 24。在假设控制输出信号是模拟信号的情况下，为此借助 D/A 转换将程序输出变量的以数字方式存在的瞬时值转换为模拟值，其于是为控制输出信号的值。

[0090] 用附图标记 42 表示程序存储器，在该程序存储器中以机器代码的形式存储应用程序。应用程序和因此机器代码借助装置 10 来创建。如果程序存储器 42 构建为芯片卡，则其在没有直接连接到计算机 12 的情况下也能够实现简单地交换机器代码与因此应用程序。可替选地，程序存储器 42 也可以构建为在安全控制装置 20 中固定地安装的存储器例如 EEPROM。

[0091] 计算机程序 16 在显示单元 14 上提供了用户界面 44。用户界面 44 能够为编程者或创建者实现应用程序的创建。编程员通过如下方式创建了应用程序 :其通过连接到计算机 12 的输入单元将输入输送给计算机 12。输入单元例如可以是键盘或鼠标。根据用于创建应用程序的编程语言，输入方案不同。如果例如将编程语言结构化文本用作编程语言，则应用程序通过文本输入来创建。而如果功能块图用作编程语言，则编程员通过如下方式创建应用程序 :编程员通过使用鼠标选择预先定义的程序模块，其例如在用户界面 44 上借助图形符号来代表。

[0092] 与所使用的编程语言无关地根据该新的方法需要由编程员进行的不同的输入。这些输入包括多个安全相关的程序变量 46 的定义、多个非安全相关的程序变量 48 的定义、从多个安全相关的程序变量 46 中选择安全相关的程序变量 50、从多个非安全相关的程序变量 48 中选择第一非安全相关的程序变量 52、分配条件 54 的定义、相关性 56 的定义和控制指令 58 的创建。相关性 56 在编程技术上对应于分配，借助分配将所选择第一非安全相关的程序变量 52 分配给所选择的安全相关的程序变量 50。在实施应用程序时由此将非安全相关的程序变量的瞬时值分配给安全相关的程序变量。相关性具有映射的特征。

[0093] 计算机程序 16 包括显示模块 60。借助显示模块 60 检测和分析由编程员通过输入单元进行的输入。一方面，显示单元 60 产生代表相应的输入的源代码 62，其存储在源代码存储器 64 中。关于完整的源代码（其在创建者进行其输入之后存在），源代码 62 是源代码范围。另一方面，显示模块 60 实现图形符号 66 以基本显示形式指示或显示。图形符号 66 代表由编程员进行的输入并且因此也代表由显示模块 60 产生的源代码 62。因此，可能的是，在显示单元 14 上显示应用程序的源代码或者应用程序。

[0094] 计算机程序 16 还包括识别模块 68，在识别模块中分析源代码 62。所有由显示模块

60 产生的源代码包含变换代码，其包括第一代码部分和第二代码部分。第一代码部分代表相关性 56 而第二代码部分代表分配条件 54。如果现在由识别模块 68 识别第一代码部分，则一方面为显示模块 60 输送识别通知 70。基于识别通知 70，显示模块 60 实现以相对于基本显示形式修改过的显示形式显示图形符号 66 的至少一个。另一方面，识别模块 68 产生复制代码 72。复制代码 72 被输送给源代码存储器 64 并且因此添加到源代码中。在此，可以考虑不同的扩展方案。复制代码 72 可以是独立的程序部分，其与第一程序部分 74 和第二程序部分 78 无关地存在。但复制代码 72 也可以是第一程序部分 74 的部分或第二程序部分 78 的部分。

[0095] 在编程员进行了所有对于其应用程序所需的输入时，源代码完全在源代码存储器 64 中。源代码包括第一程序部分 74，在其中故障安全地处理安全相关的程序变量。第一程序部分 74 又包括程序模块 76，第一代码部分和第二代码部分组合成该程序模块。基于与程序模块 76 结合的功能，程序模块 76 可以称作变换模块。此外，源代码包括第二程序部分 78，在其中处理非安全相关的程序变量，其中对于在第二程序部分 78 内的非安全相关的程序变量并不需要故障安全的处理。第一程序部分 74 包括代表安全指令的源代码，这些安全指令对于要由安全控制装置 20 解决的安全任务是必需的。安全指令不仅包括安全控制装置指令，而且也包括变换指令，其中变换指令代表相关性 56 和分配条件 54。变换指令组合成程序模块 76。组合成程序模块 76 的变换指令相对于实际的安全控制装置指令是独立的。第二程序部分 78 包括代表标准指令的源代码，标准指令对于要由安全控制装置 20 解决的标准任务是必要的。

[0096] 在源代码存储器 64 中存储的完整的源代码 80 借助编译器 82 被翻译成机器代码。优选地，其此外以 CRC（循环冗余校验）来保证。

[0097] 由编译器 82 翻译的机器代码存储在程序存储器 42 中。对于安全相关的程序变量的故障安全的处理，在程序存储器 42 中存储第一机器代码 84 和第二机器代码 86。第一机器代码 84 针对第一处理器 30 来确定而第二机器代码 86 针对第二处理器 32 来确定。第一机器代码 84 包括第一安全代码 88 和标准代码 90。第一安全代码 88 一方面包括由第一处理器 30 在由安全控制装置 20 要解决的安全任务的范围中要处理的安全指令。另一方面，第一安全代码 88 包括由第一处理器 30 在通过相关性 56 和分配条件 54 定义的转换的范围内要处理的安全指令。总之，第一安全代码 88 因此包括安全控制装置指令和变换指令。标准代码 90 包括由第一处理器 30 在由安全控制装置 20 要解决的标准任务的范围中要处理的标准指令。第二机器代码 86 包括第二安全代码 92。根据第一安全代码 88 的实施，第二安全代码 92 包括由第二处理器 32 要处理的安全指令，即安全控制装置指令和变换指令。由编程员创建的应用程序包括完整的源代码 80 和两个机器代码 84、86。

[0098] 根据应用程序的处理程序在第一处理器 30 中一方面处理第一当前安全指令 94 而另一方面处理当前标准指令 96。基本上同时在第二处理器 32 中处理第二当前安全指令 98。不仅第一当前安全指令 94 而且第二当前安全指令 98 可以是安全控制装置指令或变换指令。

[0099] 在当前标准指令 96 的处理的范围中（标准指令是标准控制指令并且因此是非安全相关的控制指令），第一非安全相关的数据 100 在第一处理器 30 与输入 / 输出单元 36 之间交换。在此情况下，通过使用程序输入变量将数据输送给第一处理器 30，其中其瞬时值

代表非安全相关的控制输入信号 102 的值，控制输入信号由非安全相关的传感器 104 产生。非安全相关的传感器 104 是如下传感器，其例如涉及对于驱动调节所需的输入量。在此情况下例如可以涉及转速、角度或速度。非安全相关的传感器 104 非故障安全地构建。输入 / 输出单元 36 通过使用程序输出变量被输送数据，其中其瞬时值代表非安全相关的控制输出信号 106 的值，控制输出信号被输送给非安全相关的执行器 108 用以其激励。非安全相关的执行器 108 例如可以是马达或者调节气缸。非安全相关的程序输出变量的瞬时值根据非安全相关的程序输入变量按照标准指令来确定。在此情况下，必要的是确定中间量，其瞬时值分配给程序中间变量。程序中间变量的瞬时值借助第二非安全相关的数据 110 输送给工作存储器 112 并且存储在那里。

[0100] 如果第一当前安全指令 94 是安全控制指令并且因此是安全相关的控制指令，则在其处理的范围中第一安全相关的数据 114 在第一处理器 30 与输入 / 输出单元 36 之间交换。在此情况下，通过使用安全相关的程序输入变量为第一处理器 30 输送数据，其中其瞬时值代表安全相关的控制输入信号 116 的值，控制输入信号由安全相关的传感器 118 产生。安全相关的传感器 118 例如是应急关断按键、双手控制装置、安全门、转速监控装置或者其他用于记录安全相关的参数的传感器。输入 / 输出单元 36 通过使用安全相关的程序输出变量被输送数据，其中其瞬时值代表安全相关的控制输出信号 120 的值，控制输出信号被输送给安全相关的执行器 122，用于其激励。安全相关的执行器 122 例如是所谓的接触器，其工作接触部设置在电流供给装置 124 与负载 28 之间的连接中。通过安全相关的执行器 122 可以关断负载 28 的电流供给装置 124，由此可能的是，在出现相应的故障时至少使负载 28 转变到安全状态中。安全相关的程序输出变量的瞬时值根据安全相关的程序输入变量按照安全控制指令来确定。在此情况下，可以必要的是，确定安全相关的中间量，其瞬时值分配给安全相关的程序中间变量。安全相关的程序中间变量的瞬时值借助第二安全相关的数据 126 输送给工作存储器 112 并且存储在那里。

[0101] 如果第二当前安全指令 98 是安全控制指令并且因此是安全相关的控制指令，则根据在第一处理器 30 中处理的第一当前安全指令 94 进行。关于第二当前安全指令 98，以相应的方式使用对应于第一安全相关的数据 114 的第三安全相关的数据 128 和对应于第二安全相关的数据 126 的第四安全相关的数据 130。

[0102] 根据前述的实施不仅由第一处理器 30 而且由第二处理器 32 产生安全相关的控制输出信号 120 的值，该前述的实施并不表示，由这两个处理器产生的值同时作为控制输出信号 120 输出。前述的实施应仅反映安全控制装置 20 的关于要解决的安全任务方面冗余的结构。两个处理器 30、32 为此构建为，确定控制输出信号 120 的值。在安全控制装置 20 的无故障的运行期间，仅将由处理器例如第一处理器 30 确定的值作为控制输出信号 120 输出。

[0103] 通过输入 / 输出单元 36 将另外的外围单元 132 连接到安全控制装置 20，外围单元在由安全控制装置 20 要解决的标准任务和安全任务的范围中变得必要。例如，在此情况下可以涉及工作方式选择开关或启用按键。但其也可以是显示单元。

[0104] 通过输入 / 输出单元 36 在安全控制装置 20 与安全相关的传感器 118、安全相关的执行器 122 和（如果需要）另外的外围单元 132 之间交换测试信号 134。借助测试信号 134 可以在安全控制装置 20 中确定，连接到安全控制装置上的单元和部件是否无故障地工作，

这是必要的，因为一旦与安全控制装置 20 连接的装置出现故障，必须保证要控制的设备 22 的安全状态。

[0105] 如果第一当前安全指令 94 和第二当前安全指令 98 是变换指令，则在两个处理器 30、32 中处理相应步骤，其对于所选择的第一非安全相关的程序变量 52 的瞬时值根据分配条件 54 被分配给所选择的安全相关的程序变量 50 是必要的。为此，所选择的第一非安全相关的程序变量 52 的在限定的时间步存在的原始瞬时值 136 被输送给复制单元 138。在复制单元 138 中基于所选择的第一非安全相关的程序变量 52 产生复制的非安全相关的程序变量。为此将原始瞬时值 136 读入两个分离的存储器区域中。出于清楚性的原因，并未示出两个存储器区域。两个存储器区域被独立地读取，使得原始瞬时值 136 被输送给第一检查单元 140 而复制的瞬时值 142 输送给第二检查单元 144。在第二检查单元 144 中因此分析并且因此处理复制的非安全相关的程序变量。

[0106] 两个检查单元 140、144 一起形成似然性单元 146。复制单元 138 和似然性单元 146 可以组合地视为变换单元。在此应提及的是，复制单元 138、检查单元 140、144 和似然性单元 146 全部是功能单元而非在工作存储器 112 内的结构单元。

[0107] 在两个检查单元 140、144 中独立地处理分配条件 54。在第一检查单元 140 中针对所选择的第一非安全相关的程序变量 52 和在第二检查单元 144 中针对用于复制的非安全相关的程序变量。因此，不仅针对所选择的第一非安全相关的程序变量 52 而且针对复制的非安全相关的程序变量确定所述的可靠性要求是否分别满足。

[0108] 针对分配条件 54，可以考虑两个实施形式。在第一实施形式中，似然性询问定义为分配条件，借助其检验所选择的第一非安全相关的程序变量 52 和所选择的另外的程序变量是否彼此相一致。相应地检验所复制的非安全相关的程序变量和所选择的另外的程序变量是否彼此相一致。如果所选择的另外的程序变量是非安全相关的程序变量，则两个检查单元 140、144 被输送在限定的时间步存在的第一瞬时值 148。在此涉及所选择的另外的非安全相关的程序变量的瞬时值。如果在第一检查单元 140 中确定所选择的第一非安全相关的程序变量 52 和所选择的另外的非安全相关的程序变量彼此相一致，则原始瞬时值 136 分配给所选择的安全相关的程序变量 50 并且输送给第一处理器 30。如果在第二检查单元 144 中确定所复制的非安全相关的程序变量和所选择的另外的非安全相关的程序变量彼此相一致，则所复制的瞬时值 142 分配给与所选择的安全相关的程序变量 50 对应的安全相关的程序变量并且输出给第二处理器 32，其中该程序变量在第二安全代码 92 中应用并且在第二处理器 32 中处理。

[0109] 而如果所选择的另外的程序变量是安全相关的程序变量，则第一检查单元 140 通过第五安全相关的数据 150 被输送有在限定的时间步存在的第二瞬时值。在此情况下涉及所选择的另外的安全相关的程序变量的瞬时值。相应地，第二检查单元 144 通过第六安全相关的数据 152 被输送有在限定的时间步存在的第三瞬时值。在此情况下涉及相应的在第二安全代码 92 中应用的安全相关的程序变量的瞬时值。在第一检查单元 140 中于是确定所选择的第一非安全相关的程序变量 52 和所选择的另外的安全相关的程序变量是否彼此相一致。在第二检查单元 144 中于是确定，所选择的第一非安全相关的程序变量 52 和相应的在第二安全代码 92 中使用的安全相关的程序变量是否彼此相一致。原始瞬时值 136 和复制的瞬时值 144 的分配和输出对应于所选择的另外的程序变量是非安全相关的程序变

量的情况来进行。

[0110] 在第二实施形式中，似然性询问被定义为分配条件，借助似然性询问检验所选择的第一非安全相关的程序变量 52 是否满足比较标准，其中比较标准代表所选择的第一非安全相关的程序变量 52 的典型特性。在该情况下，第一检查单元 140 借助第五安全相关的数据 150 被输送比较数据，原始瞬时值 136 与比较数据比较。第二检查单元 144 借助第六安全相关的数据 152 被输送相应的比较数据。如果在第一检查单元 140 中确定所选择的第一非安全相关的程序变量 52 满足比较标准，则原始瞬时值 136 被分配给所选择的安全相关的程序变量 50 并且输出给第一处理器 30。如果在第二检查单元 144 中确定所复制的非安全相关的程序变量是否满足比较标准，则所复制的瞬时值 142 分配给安全相关的程序变量并且输送给第二处理器 32，其中该程序变量在第二安全代码中应用并且对应于所选择的安全相关的程序变量 50。

[0111] 如果为了检查分配条件 54 在两个检查单元 140、144 中分析多个瞬时值或瞬时值的时间上的改变，则在两个检查单元 140、144 中中间存储对于多个相继的时间步存在的瞬时值。

[0112] 在图 1 中描述了第一实施形式，其中不仅第一安全代码 88 而且第二安全代码 92 包含变换指令。这应不具有限制作用。也可以考虑如下实施形式，其中仅仅第一安全代码 88 包含变换指令而第二安全代码 92 并不包含变换指令。也可以考虑如下实施形式，其中在一个安全代码或两个安全代码中仅仅包含代表分配条件 54 的变换指令。就复制代码 72 而言，则同样可以考虑多个实施形式。可考虑的是，仅在第一安全代码 88 中包含相关的指令。但也可以在两个安全代码 88、92 中包含相关的指令。

[0113] 图 2 中示出的流程图示出了在创建根据新的方法的应用程序时的原理方法。

[0114] 根据步骤 170，定义多个安全相关的程序变量 46。在接下来的步骤 172 中限定了多个非安全相关的程序变量 48。在随后的步骤 174 中从多个安全相关的程序变量 46 中选择安全相关的程序变量 50。在随后的步骤 176 中从多个非安全相关的程序变量 48 中选择第一非安全相关的程序变量 52。所选择的第一非安全相关的程序变量 52 在实施应用程序时被重复分配瞬时值。在随后的步骤 178 中，定义分配条件 54，其在实施该应用程序时被处理。在步骤 178 中，针对似然性询问被定义为分配条件的情况附加地选择另一程序变量。在接着的步骤 180 中，定义相关性 56，其将所选择的第一非安全相关的程序变量 52 与所选择的安全相关的程序变量 50 相关。在实施应用程序时，所选择的第一非安全相关的程序变量 52 的瞬时值根据分配条件 54 分配给所选择的安全相关的程序变量 50。如果另外的非安全相关的程序变量要转换成安全相关的程序变量，即进行另一相关（这在后续的步骤 182 中确定），则步骤 174 至 180 被重新实施。而如果不进行另外的转换并且因此相关，则接着步骤 182 之后实施步骤 184。在步骤 184 中创建控制指令。

[0115] 在图 3 中示出了第一图形用户界面 190，其代表第一编程方案。总之，应用程序通过使用在计算机 12 上运行的计算机程序 16 来创建。第一图形用户界面 190 在此使编程者能够进行对创建应用程序所需的文本输入。由计算机 16 包括的显示模块 60 实现在创建应用程序期间至少将源代码 62 或至少以摘出方式将完全的源代码 80 的一部分显示到连接到计算机 12 上的显示单元 14 上。在后续的实施中，简化地考虑源代码的摘出。

[0116] 在图 3 中示出了源代码并且因此应用程序的摘出，这通过恢复点

(Fortsetzungspunkte)192 和多个矩形 194 表示，其中矩形的每个代表源代码并且因此应用程序的程序行。总之，在显示单元 14 上显示的第一图形用户界面 190 包括多个代表源代码的图形源代码符号 196。图形源代码符号在此以基本显示形式实施。多个图形源代码符号 196 包括多个图形变换源代码符号 198。多个图形变换代码符号 198 代表在源代码中包含的变换代码。变换代码又包括 :第一代码部分 200，其代表相关性 56 ;和第二代码部分 202，其代表分配条件 54。借助识别模块 68 识别两个代码部分 200、202 的至少之一。在存在所识别的代码部分的情况下，显示模块 60 实现 :在多个图形变换代码符号 198 中包含的至少一个图形变换代码符号 204 以相对于基本显示形式修改过的显示形式显示。这在图 3 中通过如下方式表示，两个以附图标记 204 表示的矩形设置有阴影。

[0117] 第一代码部分 200 代表变换指令，其代表相关性 56。第二代码部分 202 代表变换指令，其代表分配条件 54。如在图 3 中通过虚线表示的块所表示的那样，两个代码部分 200、202 组合成程序模块 76。

[0118] 通过以相对于基本显示形式修改过的显示形式显示至少一个图形变换符号 204，在源代码内并且因此在应用程序内提供了被检测的变换位置。

[0119] 在图 4 中示出了第二图形用户界面 210，其代表第二编程方案。在该第二编程方案中应用程序也通过使用在计算机 12 上运行的计算机程序 16 来创建。然而在第二编程方案中通过从多个预定义的程序模块选择多个程序模块来进行应用程序的创建。为了能够进行选择，显示模块 60 实现多个图形程序模块符号 212 在第二图形用户界面 210 并且因此在显示单元 14 上的显示。多个图形程序模块符号 212 又包括 :第一数目的图形程序模块符号 214，其代表预先定义的程序模块 ;和第二数目的图形程序模块符号 216，其代表多个所选择的程序模块。预先定义的程序模块的选择例如通过使用鼠标借助拖放功能来进行并且在图 4 中借助两个箭头 218 来表示。至少第一数目的图形程序模块符号 214 以基本显示形式实施。多个预先定义的程序模块包括在创建应用程序期间所创建的程序模块 220，其代表相关性 56 和分配条件 54。所创建的程序模块 220 通过至少一个修改过的图形程序模块符号 222 代表。显示模块 60 实现修改过的图形程序模块符号 222 以相对于基本显示形式修改过的显示形式显示。这在图 4 中通过使用阴影来表示。所创建的程序模块 220 对应于前面所描述的程序模块 76。

[0120] 程序模块的创建在第二编程方案中如下进行 :在特定为此所设置的输入区中编程员规定要创建的程序模块的功能。为此，编程员例如以文本输入形式输入由要创建的程序模块实施的控制指令。在该阶段中，针对要创建的程序模块已应用图像程序模块符号。现在如果识别模块 68 在由编程员输入的控制指令内识别两个代码部分 200、202 之一，则已经应用的图形程序模块符号被修改，由此其以相对于基本显示形式修改过的显示形式显示。

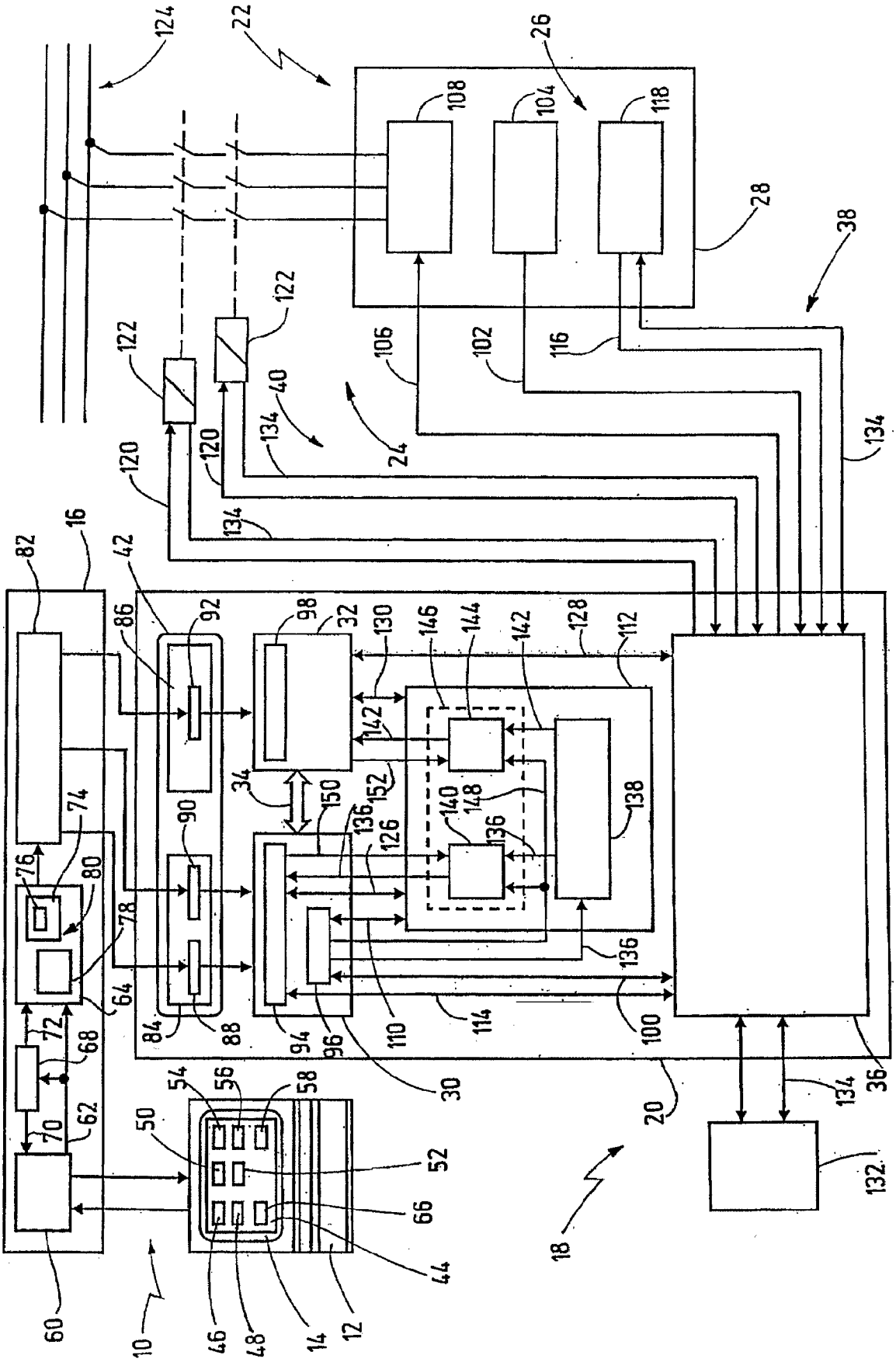

124
22
26
108
104
118
28
38
122
122
106
102
116
40
134
24
120
120
134
134
16
82
42
86
92
98
32
130
146
144
142
128
112
74
90
34
150
152
140
148
138
136
126
136
76
80
78
64
72
68
84
88
94
96
110
114
100
30
20
134
36
54
56
58
50
52
66
62
70
44
60
10
46
48
14
12
18
132

图 1

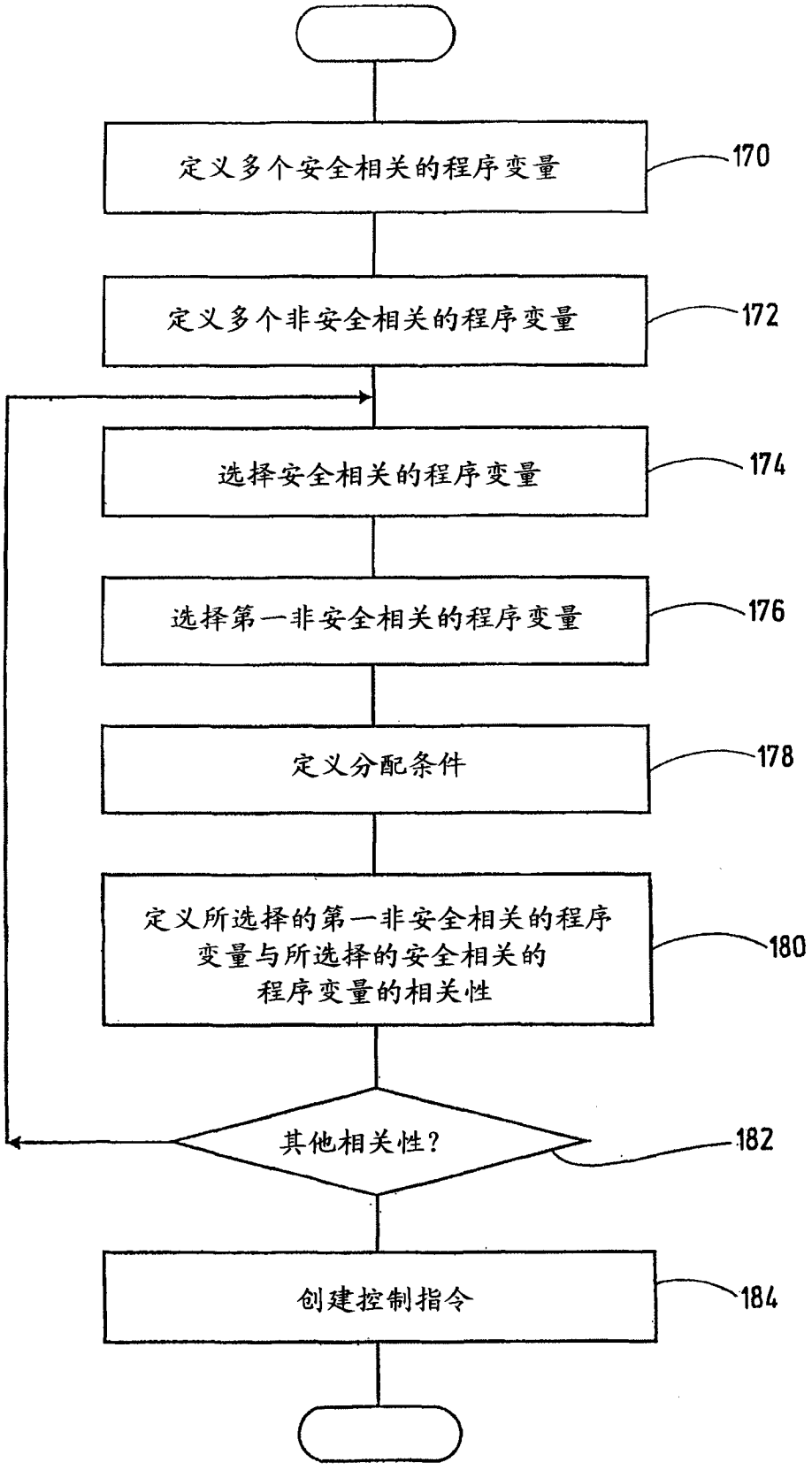

定义多个安全相关的程序变量
170
定义多个非安全相关的程序变量
172
选择安全相关的程序变量
174
选择第一非安全相关的程序变量
176
定义分配条件
178
定义所选择的第一非安全相关的程序变量与所选择的安全相关的程序变量的相关性
180
其他相关性?
182
创建控制指令
184

图 2

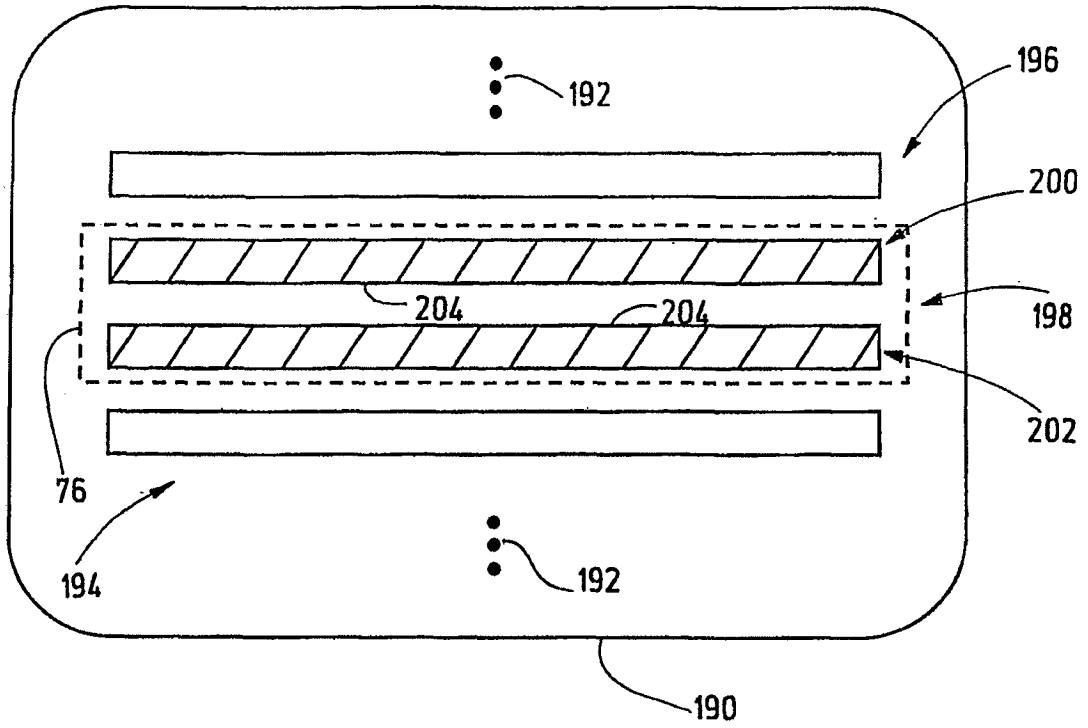

192
196
200
204
204
198
202
76
194
192
190

图 3

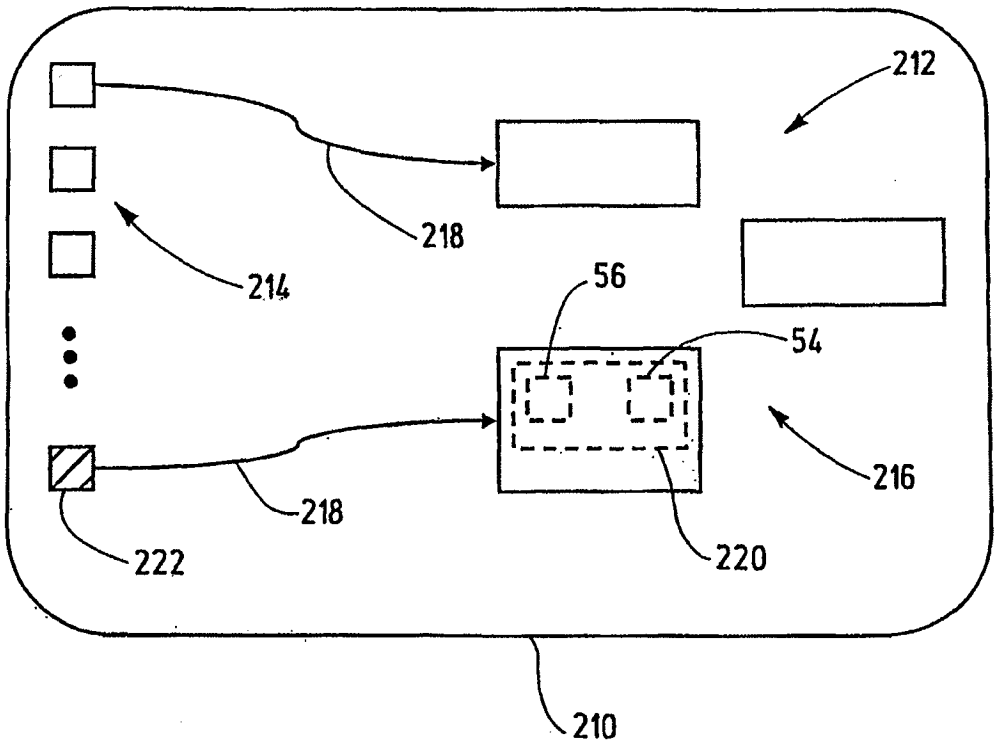

212
218
214
56
54
216
218
222
220
210

图 4