[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/45 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200580036137.1

[43] 公开日 2009 年 5 月 13 日

[11] 公开号 CN 101432696A

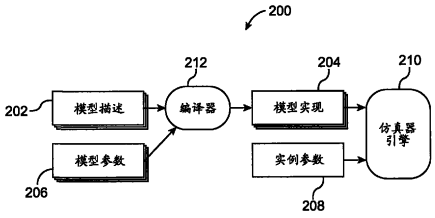
[22] 申请日 2005.10.20
[21] 申请号 200580036137.1
[30] 优先权
[32] 2004.10.20 [33] US [31] 10/965,311
[86] 国际申请 PCT/US2005/037605 2005.10.20
[87] 国际公布 WO2007/044018 英 2007.4.19
[85] 进入国家阶段日期 2007.4.20
[71] 申请人 卡登斯设计系统公司
地址 美国加利福尼亚州
[72] 发明人 肯尼思·S·昆德特
[74] 专利代理机构 北京康信知识产权代理有限责任公司
代理人 余 刚 尚志峰
权利要求书 21 页 说明书 28 页 附图 5 页

[54] 发明名称

模型编译方法

[57] 摘要

提供了用于编译供仿真使用的模型的方法，该方法包括接收模型的描述；以及自动地将描述转换成为在仿真期间所选择的分析定制的模型的实现。



1. 一种编译用在仿真中的模型的方法，所述方法包括:

接收所述模型的描述;

自动地将所述描述转换成仿真期间为选择的分析定制的所述模型的实现。

2. 根据权利要求 1 所述的方法，

其中，所述选择的分析是 AC 分析；以及

其中，为所述 AC 分析定制所述模型的所述实现。

3. 根据权利要求 1 所述的方法，

其中，所述模型表示电子电路的至少一部分;

其中，所述选择的分析是 AC 分析；以及

其中，为所述 AC 分析定制所述模型的所述实现。

4. 根据权利要求 1 所述的方法，

其中，所述选择的分析是 DC 分析；以及

其中，为所述 DC 分析定制所述模型的所述实现。

5. 根据权利要求 1 所述的方法，

其中，所述模型表示电子电路的至少一部分;

其中，所述选择的分析是 DC 分析；以及

其中，为所述 DC 分析定制所述模型的所述实现。

6. 根据权利要求 1 所述的方法，

其中，所述选择的分析是瞬态分析；以及

其中，为所述瞬态分析定制所述模型的所述实现。

7. 根据权利要求 1 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的分析是瞬态分析；以及

其中，为所述瞬态分析定制所述模型的所述实现。

8. 根据权利要求 1 所述的方法，

其中，所述选择的分析是谐波平衡分析；以及

其中，为所述谐波平衡分析定制所述模型的所述实现。

9. 根据权利要求 1 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的分析是谐波平衡分析；以及

其中，为所述谐波平衡分析定制所述模型的中间表示。

10. 根据权利要求 1 所述的方法，

其中，所述选择的分析是从包括蒙特卡洛、失配、参数、和拐角分析的组中选出的分析；以及

其中，为所述分析定制所述模型的所述中间表示。

11. 根据权利要求 1 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的分析是从包括蒙特卡洛、失配、参数、和拐角分析的组中选出的分析；以及

其中，为所述分析定制所述模型的中间表示。

12. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

自动地将所述描述转换成仿真期间为选择的算法使用而定制的所述模型的实现。

13. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行 Samanski 法则；以及

其中，为所述 Samanski 法则定制所述模型的所述实现。

14. 根据权利要求 12 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的算法执行 Samanski 法则；以及

其中，为所述 Samanski 法则定制所述模型的所述实现。

15. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行牛顿法，以及

其中，为所述牛顿法定制所述模型的所述实现。

16. 根据权利要求 12 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的算法执行牛顿法；以及

其中，为所述牛顿法定制所述模型的所述实现。

17. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行松弛法，以及

其中，为所述松弛法定制所述模型的所述实现。

18. 根据权利要求 12 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的算法执行松弛法；以及

其中，为所述松弛法定制所述模型的所述实现。

19. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行 Krylov 法，以及

其中，为所述 Krylov 法定制所述模型的所述实现。

20. 根据权利要求 12 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的算法执行 Krylov 法；以及

其中，为所述 Krylov 法定制所述模型的所述实现。

21. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行数值积分法，以及

其中，为所述数值积分法定制所述模型的所述实现。

22. 根据权利要求 12 所述的方法，

其中，所述选择的算法执行从包括向前和向后欧拉、梯形法则、向后差分公式、和 Runge Kutta 的组中选出的数值积分法；以及

其中，为所述数值积分法定制所述模型的所述实现。

23. 根据权利要求 12 所述的方法，

其中，所述模型表示电子电路的至少一部分；

其中，所述选择的算法执行数值积分法；以及

其中，为所述数值积分法定制所述模型的所述实现。

24. 一种编译用在仿真中的模型的方法，所述方法包括：

接收表示电子电路至少一部分的所述模型的描述；

自动地将所述描述转换为被定制以根据所选划分标准支持划分的所述模型的实现。

25. 根据权利要求 24 所述的方法，

其中，所述电路划分方法标准适合于供基于松弛的定时仿真器使用。

26. 根据权利要求 24 所述的方法，

其中，中间表示包括多个分区。

27. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

自动地将所述描述转换成仿真期间为并行估计定制的所述模型的实现。

28. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

在除去所述模型的所选特征的过程中，自动地将所述描述转换为所述模型的实现。

29. 根据权利要求 28 所述的方法，

其中，选择包括选择支持算法；以及

其中，在自动转换所述描述的所述过程中，将支持算法从所述模型中去除。

30. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

自动地将所述描述转换成仿真期间由所述模型表示的元件可被表示为连接至一个或多个其他元件的方式定制的所述模型的实现。

31. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

自动地将所述描述转换成通过改变所述模型的结构拓扑定制的所述模型的实现。

32. 一种编译用在仿真中的模型的方法，所述方法包括：

接收所述模型的描述；

接收所述描述内的参数的一组参数值；

自动地将所述描述转换为至少包括一些参数值的所述模型的实现。

33. 根据权利要求 32 所述的方法，进一步包括:

预估计依赖于所述一组参数值至少一部分的所述模型的一部分。

34. 根据权利要求 32 所述的方法，进一步包括:

只为没有接收到值的参数分配与所述实现相关的参数存储器。

35. 根据权利要求 32 所述的方法，

其中，所述一组模型参数值表示最坏情况的拐角。

36. 根据权利要求 32 所述的方法，进一步包括:

检查将被仿真的设计，以确定编译掉哪个参数。

37. 根据权利要求 32 所述的方法，进一步包括:

检查将被仿真的设计，以使所有参数都被编译掉。

38. 根据权利要求 32 所述的方法，进一步包括:

检查将被仿真的设计，以确定编译掉哪个参数；以及将保留在所述实现中的参数作为参数。

39. 一种编译方法，包括:

接收将在仿真器中被仿真的模型的描述;

接收所述描述内的参数的一组参数值;

自动地将所述描述转换为实现，以编译掉已经接收到参数值的所述参数中的至少一些参数。

40. 一种编译用在仿真中的模型的计算机执行方法，所述方法包括:

接收将在仿真器中被仿真的所述模型的语言描述的多个版本;

自动地将所述描述的所选版本转换为所述模型的实现。

41. 根据权利要求 40 所述的方法，

其中，所述多个版本中的至少两个提供不同的速度/精度折衷。

42. 根据权利要求 40 所述的方法，

其中，在复合描述中指定所述多个版本；以及

其中，自动转换包括使用编译器指令以将所选版本与另一个版本区分开来。

43. 根据权利要求 40 所述的方法，

其中，在复合描述中指定所述多个版本；以及

其中，自动转换包括使用编译器指令以将所选版本与另一个版本区分开来；以及

其中，在与所述模型描述相关的注释中指定所述编译器指令。

44. 根据权利要求 40 所述的方法，

其中，在复合描述中指定所述多个版本；以及

其中，自动转换包括使用编译器指令以将所选版本与另一个版本区分开来；以及

其中，在不同于包括所述复合描述的一个或多个文件的文件中指定所述编译器指令。

45. 根据权利要求 40 所述的方法，

其中，在符号描述中指定所述多个版本；以及

其中，自动转换包括使用编译器指令以将所选版本与另一个版本区分开来；以及

其中，由用户交互地指定所述编译器指令。

46. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述实现包括以指定仿真器为目标的可执行代码。

47. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述实现表示使用 C 代码或 C++代码的所述模型。

48. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述模型描述包括 HDL 模型描述。

49. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述实现包括所述模型的中间表示。

50. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述实现包括所述模型的中间表示；以及进一步包括：

将所述中间表示转换为以指定仿真器为目标的可执行代码。

51. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述实现包括所述模型的中间表示；以及进一步包括：

将所述中间表示转换为以指定仿真器为目标的可执行代码；以及

接收由具有所述可执行代码的所述指定仿真器使用的一组实例参数值。

52. 根据权利要求 1、12、24、27、28、30、31、32 或 40 所述的方法，

其中，所述模型包括压缩模型。

53. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括：

接收一个或多个定制标准；以及

将元件模型的高级描述转换为根据一个或多个接收标准定制的所述模型的实现。

54. 根据权利要求 53 所述的计算机可读介质，

其中，所述一个或多个定制标准包括一种分析的指示和和一种算法的指示中的一个或多个；以及

其中，转换包括：

如果所述定制标准包括一种分析，则将所述模型的所述高级描述转换成为所指示类型的分析定制的所述模型的实现，以及

如果所述定制标准包括一种算法，则将所述模型的所述高级描述转换成为所指示类型的算法定制的所述模型的实现。

55. 根据权利要求 53 所述的计算机可读介质，

其中，所述一个或多个定制标准包括划分标准的指示和结构拓扑标准的指示中的一个或多个；以及

其中，转换包括：

如果所述定制标准包括划分标准，则将所述模型的所述高级描述转换成为所指示划分标准定制的所述模型的实现，以及

如果所述定制标准包括结构拓扑，则将所述模型的所述高级描述转换成为所指示结构拓扑定制的所述模型的实现。

56. 根据权利要求 53 所述的计算机可读介质，

其中，所述一个或多个定制标准可进一步包括划分标准的指示；以及

其中，转换包括：

如果所述定制标准包括划分标准，则将所述模型的所述高级描述转换成为所指示划分标准定制的所述模型的实现。

57. 根据权利要求 53 所述的计算机可读介质，

其中，所述一个或多个定制标准可进一步包括结构拓扑标准的指示；以及

其中，转换包括：

如果所述定制标准包括结构拓扑标准，则将所述模型的所述高级描述转换成为所指示划分标准定制的所述模型的实现。

58. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括：

接收在仿真期间模型被使用的分析的指示；以及

将所述模型的高级描述转换成为所指示分析定制的所述模型的实现。

59. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于 AC 分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述 AC 分析定制的所述模型的实现。

60. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于 DC 分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述 DC 分析定制的所述模型的实现。

61. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于瞬态分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述瞬态分析定制的所述模型的实现。

62. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于谐波平衡分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述谐波平衡分析定制的所述模型的实现。

63. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于谐波平衡分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述谐波平衡分析定制的所述模型的实现。

64. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于蒙特卡洛分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述蒙特卡洛分析定制的所述模型的实现。

65. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于失配分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述失配分析定制的所述模型的实现。

66. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于参数分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述参数分析定制的所述模型的实现。

67. 根据权利要求 58 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于拐角分析的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述拐角分析定制的所述模型的实现。

68. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括:

接收在仿真期间模型被使用的分析的指示;

如果所指示的分析是 AC 分析，则将所述模型的高级描述转换成为所述 AC 分析定制的所述模型的实现；以及

如果所指示的分析是 DC 分析，则将所述模型的高级描述转换成为所述 DC 分析定制的所述模型的实现。

69. 根据权利要求 68 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是瞬态分析，则将所述模型的所述高级描述转换成为所述瞬态分析定制的所述模型的实现。

70. 根据权利要求 68 或 69 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是谐波平衡分析，则将所述模型的所述高级描述转换成为所述谐波平衡分析定制的所述模型的实现。

71. 根据权利要求 68 或 69 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是蒙特卡洛分析，则将所述模型的所述高级描述转换成为所述蒙特卡洛分析定制的所述模型的实现。

72. 根据权利要求 68 或 69 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是失配分析，则将所述模型的所述高级描述转换成为所述失配分析定制的所述模型的实现。

73. 根据权利要求 68 或 69 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是参数分析，则将所述模型的所述高级描述转换成为所述参数分析定制的所述模型的实现。

74. 根据权利要求 68 或 69 所述的计算机可读介质，

其中，转换包括:

如果所指示的分析是拐角分析，则将所述模型的所述高级描述转换成为所述拐角分析定制的所述模型的实现。

75. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括:

接收在仿真期间模型被使用的算法的指示;

将所述模型的高级描述转换成为所指示的算法定制的所述模型的实现。

76. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于 Samanski 法则的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述 Samanski 法则定制的所述模型的实现。

77. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于牛顿法的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述牛顿法定制的所述模型的实现。

78. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于松弛法的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述松弛法定制的所述模型的实现。

79. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于 Krylov 法的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述 Krylov 法定制的所述模型的实现。

80. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于数值积分法的指示；以及

其中，转换包括将所述模型的所述高级描述转换成为所述数值积分法定制的所述模型的实现。

81. 根据权利要求 75 所述的计算机可读介质，

其中，接收包括接收在仿真期间模型被用于数值积分法的指示，所述数值积分法选自包括向前和向后欧拉、梯形法则、向后差分公式、和 Runge Kutta 的组；以及

其中，转换包括将所述模型的所述高级描述转换成为所述数值积分法定制的所述模型的实现。

82. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括：

接收在仿真期间模型被使用的算法的指示；

如果所指示的算法是 Samanski 法则，则将所述模型的高级描述转换成为所述 Samanski 法则定制的所述模型的实现，以及

如果所指示的算法是牛顿法，则将所述模型的高级描述转换成为所述牛顿法定制的所述模型的实现。

83. 根据权利要求 82 所述的计算机可读介质，

其中，转换包括：

如果所指示的算法是松弛法，则将所述模型的所述高级描述转换成为所述松弛法定制的所述模型的实现。

84. 根据权利要求 82 所述的计算机可读介质，

其中，转换包括:

如果所指示的算法是 Krylov 法，则将所述模型的所述高级描述转换成为所述 Krylov 法定制的所述模型的实现。

85. 根据权利要求 82 所述的计算机可读介质，

其中，转换包括:

如果所指示的算法是积分，则将所述模型的所述高级描述转换成为数值积分定制的所述模型的实现。

86. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括:

接收将在仿真器中被仿真的模型的描述;

接收所述描述内的参数的一组参数值;

自动地将所述描述转换为至少包括一些参数值的所述模型的实现。

87. 根据权利要求 86 所述的计算机可读存储介质，进一步包括用于执行一种方法的可执行指令，所述方法进一步包括:

预估计依赖于所述一组参数值的至少一部分的所述模型的一部分。

88. 根据权利要求 86 所述的计算机可读存储介质，进一步包括用于执行一种方法的可执行指令，所述方法进一步包括:

只为没有接收到值的参数分配与所述实现相关的参数存储器。

89. 一种计算机可读介质，具有用于执行一种方法的可执行指令，所述方法包括:

接收将在仿真器中被仿真的模型的描述;

接收所述描述内的参数的一组参数值;

自动地将所述描述转换为实现，以将接收到参数值的所述参数中的至少一些参数编译掉。

90. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

编译器装置，用于将所述描述转换成为仿真期间选择的分析定制的所述模型的实现。

91. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

编译器装置，用于将所述描述转换成为仿真期间选择的算法使用所定制的所述模型的实现。

92. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置，所述模型表示电子电路的至少一部分；以及

编译器装置，用于将所述描述转换为被定制以根据所选划分标准支持划分的所述模型的实现。

93. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

编译器装置，用于在将所述模型的所选特征去除的过程中将所述描述转换为所述模型的实现。

94. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置;

编译器装置，用于将所述描述转换成为仿真期间由所述模型表示的元件可被表示为连接至一个或多个其他元件的方式定制的所述模型的实现。

95. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

编译器装置，用于将所述描述转换为通过改变所述模型的结构拓扑定制的所述模型的实现。

96. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

用于接收所述描述内的参数的一组参数值的装置;

编译器装置，用于将所述描述转换为至少包括一些参数值的所述模型的实现。

97. 一种计算机处理系统，包括:

用于接收将在仿真器中被仿真的模型的描述的装置；以及

用于接收所述描述内的参数的一组参数值的装置;

编译器装置，用于将所述描述转换为实现，以编译掉已经接收到参数值的所述参数中的至少一些参数。

98. 一种计算机处理系统，包括:

用于接收将在仿真器中仿真的模型的描述的多个版本的装置;

编译器装置，用于将所述描述的所选版本转换为所述模型的实现。

模型编译方法

技术领域

本发明总的来说涉及系统的计算机仿真，更具体地，涉及用在仿真器中的模型。

背景技术

相关技术描述

仿真概述

仿真器是用于基于系统描述来分析系统的计算机软件程序。基本上，仿真器允许用户在系统的仿真操作期间通过执行分析来询问关于系统的问题。仿真器数字地计算系统对仿真刺激的响应。通常，仿真器将各种刺激施加给可执行的系统模型以预测系统将如何响应。仿真使得能够在开始实际实现系统的时间和开支之前理解复杂的系统如何运转。通常，系统描述包括描述系统行为的方程。仿真器在由刺激表示的上下文仿真条件中解决这些方程，以仿真在那些条件下系统的运行。关于仿真器更透彻的论述，参见 Ken Kundert, *The Designer's Guide to SPICE and Spectre*, Kluwer Academic Publishers, 1995。

电路仿真器通常从包含在电路中的元件的数学模型构造电路方程。模型可以为内置的（对于 SPICE 之类的仿真器中半导体模型通常是这种情况），或者它们可以由用户使用某些类型的模型化语

言（例如，Verilog-AMS，其为许多不同的硬件描述语言（HDL）中的一种）来指定。电路仿真器通常构造一组普通的差分方程来描述将被仿真的电路。从电路元件的参数化模型以及网络表中提供的电路连接性信息来构造方程。给定初始条件和刺激，从方程求解出电路的响应。更具体地，为了形成方程，仿真器将单个元件的数学模型与描述元件如何互联的方程结合起来。从 Kirchoff（基尔霍夫）电压和电流定律导出互联方程。KCL 规定在任何时刻流出节点的所有电流总和为零。KVL 规定在任何时刻沿一个环路的所有分支电压的代数和为零。

因此，诸如 SPICE 的电路仿真器进行电路描述并将其进行仿真，以预测其真实的领域行为。电路描述通常包括网络表，其为具有描述元件应该如何彼此连接的附加信息的元件列表。电路描述还包括各种电路元件的参数值。电路描述还包括使其接口数量联系起来的方程的集合形式的单个电路元件的详细模型。此外，电路描述通常包括关于当元件被互联时如何结合接口数量的一般规则。过去，电路仿真器已经提供了这些方面作为内置能力。元件模型通常用诸如 C 的编程语言来编写，并且被编译入仿真器。互联规则基于基尔霍夫定律并被硬编码进仿真器。

虽然该方法已经在电子工业很好地应用了三十多年，但是它在许多方面都存在问题。以下是使用该方法所能引起的问题类型的实例。第一，对于被编译进仿真器的模型，它们对于最终用户来说是黑箱。通常，用户能够理解模型需要大量的文档，并且通常由于模型的复杂性，使得文档与所实施的模型不匹配。第二，用户对模型具有最低限度的控制或者没有控制；他们通常必须接受他们所被给定的。因为模型可能在各种仿真器之间不匹配，所以这对于需要使用来自不同厂家的仿真器的用户是存在问题的。第三，更新或修改模型的过程将会是很费力的并且非常慢。这可能涉及许多不同当事

者，所有当事人都必须同意更新，然后必须行动以执行该更新，并且这些当事者中的许多可能不直接从更新中受益。

出于这些和其他原因，已经趋向于摆脱通用的编程语言（例如C），而是改为使用专用的模型化语言（例如，Verilog-A）来编写模型。为此，当前正加强 Verilog-A 语言以更好地支持压缩建模。期望当与如果以 C 实现相比以 Verilog-A 编写模型时，模型的实现和维护将更容易。另外，用 Verilog-A 编写的模型将更容易理解，这样能够期望最终用户自己将能够增强并维护它们。为了使这得以实现，仿真器不久将会配备有“模型编译器”，其将使用户将用 Verilog-A 编写的模型转换为链接到仿真器中的模型的实现。在许多情况下，Verilog-A 模型将首先被翻译为诸如 C 或 C++的通用编程语言，然后生成的代码被编译和链接到仿真器中。模型编译器已经开始出现。例如，参见 Lemaitre, L., McAndrew, C., Hamm, S., ADMS-Automatic Device Model Synthesizer, IEEE 2002 Custom Integrated Circuit Conference, 2002 年 5 月，p.27-30。基于标准化 Verilog-A 语言的编译器很可能大量地取代基于非标准语言的现有编译器。

更具体地，当实现用于电路仿真器的模型时，通常已经为目标提供了当在仿真器内执行时所需要的时间和内存方面是高效，并且在开始用 C 实现模型时的时间方面是高效的实现。应该理解，尽管在过去 C 编程语言已经被最经常地应用于实现模型，但是诸如 C++和 Fortran 的其他的编程语言也已经被使用。特别关心的是模型的参数。诸如 BSIM3 的现代 MOS 模型可具有数百个参数。存储模型每个实例的这些参数中每一个的各个副本将需要惊人的开支；所以 SPICE 将这些参数划分为实例参数和模型参数。

通常可以在模型的每个实例上指定通常数字很小的实例参数，因此，每个实例将需要存储这些参数。实例参数通常传递几何信息。

剩余的参数（即，模型参数）通常用于描述制造过程；所以将在大量的实例之间共享。因而，通常使用*.model*语句将模型参数指定为不同的组。然后，每个实例都将引用单个*.model*语句。以这种方式，可以在许多实例之间共享*.model*参数的单个副本，这可以减少内存需求。通过仔细地实现模型，对于每组*.model*参数，只涉及*.model*参数以及或许诸如温度的全局环境参数的计算可以被执行一次，并且结果与*.model*参数一起存储。

图 1 是表示使用具有仿真器的元件模型的传统方法 **100** 的说明性流程图。模型描述 **102**（可以用诸如 C 的语言或诸如 Verilog-A 的 HDL 语言进行描述）例如被提供给编译器 **104**（其为将模型描述 **102** 转换为模型实现（implementation）**106** 的计算机程序）。模型实现 **106** 连同分别指定的模型参数 **110** 和实例参数 **112** 被提供给仿真器引擎 **108**。

一般来说，模型实现的这种较早方法的特性是存在对每个模型的描述，并且描述被定制为具有两种参数：实例参数，其对于每个实例是唯一的；以及模型参数，其在多个实例之间共享；并且描述被编译为单个实现，以及在仿真器内将实现与实例参数和模型参数结合起来。

通过以下所选仿真器特性的说明，将会更好地理解仿真的复杂性和元件模型的作用。

仿真期间的分析

这里使用的分析是在给定的一组条件或假设的情况下由仿真器使用以预测系统行为的过程。一种能够通过仿真进行分析的系统是包括多个单个元件的电路。可以采用仿真来执行许多不同类型的分析，例如，瞬态分析、DC 分析、AC 分析、谐波平衡分析、Monte

Carlo（蒙特卡洛）分析、失配分析和参数分析、以及拐角分析，只列举了几个实例。

瞬态分析计算作为时间函数的系统响应。瞬态分析通常包括非线性常差分方程的系统。没有已知的方法直接解决这些方程。例如，用在电路仿真器中的方法是使时间离散，其将问题从非线性差分方程的单个系统的解法转换为解决非线性代数方程的系统序列的解法。以离散时间近似取代时间导数算子以及从一些初始条件开始一次一个时间点地求解所生成的有限差分方程被称作数字积分差分方程。存在四种通常用于电路仿真的积分方法，例如，向前欧拉、向后欧拉、梯形法则和向后差分公式（也被称为 Gear 法）。

DC 分析通常用于计算系统的操作点。一般地，操作点只是一些解轨迹（solution trajectory）的快照。在 DC 分析中，操作点也被假定为平衡点。平衡点是常量操作点。换言之，平衡点是不随时间改变的解。例如，如果刺激仍然在变化，则电路不能达到平衡点，因此 DC 分析中的第一步骤是配置独立资源以使它们恒定。此外，所有波形在平衡点是常量，dv/dt=0 以及 di/dt=0，因此，电容器起到开路的作用，以及电感器起到短路的作用。因此，计算平衡点的基本方法包括将所有独立的刺激源配置为常量；用开路取代所有电容器；并且用短路取代所有电感器。求解描述所生成系统的方程给出了平衡点。方程的系统是非线性的以及代数的（没有时间导数或积分）。Newton（牛顿）法是众所周知的求解代数方程的大非线性系统的方法。牛顿法是一直持续到满足某些停止标准的迭代过程。

AC 分析通常包括用“小”正弦信号驱动诸如电路的系统，并且计算稳态或最终解。由于刺激很小，所以可将电路线性化，并且所有生成的信号为正弦信号。AC 分析是计算不具有与瞬态分析相关的精确性问题或 DC 分析的收敛问题的传递函数的有效方式。

谐波平衡分析在频域中用公式表示电路方程，意味着解被写为傅立叶级数。

错误！不能从编辑域代码中创建对象

根据傅立叶系数 V_k 而不是根据波形 $V(t)$ 给出解。对于线性元件这相对容易实现，在时域模型和频域模型之间存在直接映射，但是对于非线性元件则更麻烦。所以通常通过谐波平衡，在时域中估算非线性元件模型并将结果转换回频域，使得它们能够与来自线性元件的结果结合起来。在瞬变分析中，一次一个点地建立解波形，所以只一次一个点地估计模型，但是通过谐波平衡，可以一次估算整个波形的模型。

蒙特卡洛是元分析，是指其是运行其他分析的分析。在蒙特卡洛分析期间，用户识别系统中变量的集合；并且仿真器为那些变量选择随机值并使用随机选择的变量仿真系统。仿真器继续选择随机值的循环，之后使用所选择的值进行仿真以形成表示系统依赖于为识别变量选择的值的统计“图片”。

除了识别变量用于应该匹配的东西以外，失配分析类似于蒙特卡洛分析。参数分析通常包括识别参数，并且在仿真过程中在值的范围内对其进行扫描。例如，参数可能为温度，可在多个仿真循环过程中从冰冻到沸腾对其进行扫描。

仿真期间的算法

这里使用的算法是由仿真器使用来实现分析的特定程序。用在SPICE 中的算法定义我们应该将什么称作仿真的传统方法。该方法被称作仿真的直接方法。在用于仿真的直接方法中，首先用公式表示描述系统的非线性常差分方程，然后通过诸如梯形规则的多步积

分法转换为差分方程系统。使用 Newton-Raphson（牛顿-拉夫逊）算法求解非线性差分方程，其生成使用稀疏高斯消去法求解的线性方程序列。

显式积分和松弛法是两种可选算法方法的实例。诸如向前欧拉的显式积分法消除了实际求解描述系统的差分方程的大系统的需要。而是从前一时刻的解外推出特定时刻的解。在电路仿真的环境中，假定不存在浮置电容器，并且存在将电路中的每一个节点连接至地的至少一个电容器。通过估计电路方程（不求解它们）来执行外推，以确定进入接地电容器的电流。

松弛方法在系统中采用等待时间（latency），通过将其分成较小的元件并独立求解每段。如果在一个或多个段中的信号是潜伏的，则不必求解它们。在电路仿真的环境中，波形松弛法将该方法又向前发展了一步。电路被分成子电路，但是，随着时间独立求解子电路而不是在单个时间点求解子电路。这使得仿真器可以采用多速行为以及等待时间。多速行为是子电路中的信号正在变化、但在一个子电路中的信号比另一个中的信号变化更慢的情况。在这种情况下，在其信号变化更慢的子电路中，仿真器能够自由选择较大的时间步。

用于仿真的元件建模

诸如 SPIEC 的传统电路仿真器提供了相对有限的内置模型集，模型元件所需要的通常在集成电路上可以得到。存在相对有限的添加新模型的能力。这些更早的仿真器一般只为最终用户提供添加由少量简单公式描述的元件的能力。这些现有的仿真器解译模型，因此，相对执行较慢。不幸的是，这种仿真方法一般不太适于复杂模型或那些大量使用的模型。

Verilog-AMS 是具有能够用于有效描述模型的宽范围的特征的 HDL 实例。实例包括诸如电阻器、电容器、和电感器的基本元件。额外的实例包括半导体元件，例如，二极管、BJT、MOSFET、和变容二极管。公共领域的 Verilog-A 模型实际存在于所有普遍使用的压缩模型，例如 Gummel-Poon、VBIC、Mextram BJT 模型和 MOS3、BSIM3&4 和 EKV MOS 模型。HDL 模型也存在功能块（例如数据转换器、解调/调制器、采样器和滤波器），以及逻辑元件（例如，门电路、锁存器和电阻器）。此外，存在用于多学科元件（例如，传感器、致动器、和换能器）的 HDL 模型。对 Verilog-AMS 更完整的论述参见 Ken Kundert 和 Olaf Zinke, *The Designer's Guide to Verliog-AMS*, Kluwer Academic Publishers, 2004。

向诸如 SPICE 的电路仿真器添加模型的能力显著增加了其范围，并能使其非常强大。通过多种不同的模型，仿真器可用于仿真非常多的不同类型的系统。例如，为激光器和光电二极管添加模型的能力使仿真器适合于仿真光电系统。添加磁性和功率半导体模型的能力使系统适合于仿真功率系统。一般地，适当的模型能够使仿真器适合于仿真例如机械、热力、声学或流体系统。

对于器件技术人员，器件模型通常表示将电荷载流子分布表示为器件结构、工艺参数、温度、和时间函数的一组方程。对于电路设计者，器件模型通常表示“压缩模型”，其抽象掉许多技术细节并行为地模仿 I-V 特性。压缩模型被实现为编程语言中的语句。为了实现任何 SPICE 模型（包括 BSIM），设计者通常应该详细理解仿真器，包括其用于和模型进行通信的调用的序列。程序语句通常适合于感兴趣的仿真器的应用程序接口（API）。参见 Justin E. Harlow, *A Gentle Introduction to Compact Model Compilers*, 2002。

以诸如 Verilog-AMS(Verilog 和 Verilog-A 的超集）的工业标准 HDL 或 VHDl-AMS（VHDL 的超集）编写的模型是更加可以移植

的。此外，以 HDL 源形式编写的模型使用户更易于校正缺陷或增强。结果，例如，对于使用 HDL 而不是诸如 C 或 C++的系统级语言编写模型的兴趣越来越大。

将压缩模型编译到特定目标仿真器

压缩模型编译器将模型语句从 HDL 翻译为中间数据结构。紧接着，中间数据结构被翻译为适合于以仿真器使用的形式。该方法的优点是简化了将模型瞄准具有不同 API 的不同仿真器的过程，这是因为瞄准从中间表示继续进行而不是从模型方程继续进行，参见 Justin E.Harlow, *Supra*。

模型编译器的一个实例是自动器件模型合成器（ADMS），其将输入视为模型的 Verilog-A 描述，并生成直接编译到仿真器中的 C 代码。用前端和后端构造 ADMS，其中，前端将 Verilog-A 编译为中间表示，而后端将中间表示转换为直接编译到目标仿真器中的 C 代码。ADMS 能够为不同的仿真器（例如，Spectre、ADS、McSpice、和 NanoSim）提供不同的后端。因此，可以容易地将相同的模型编译到不同的仿真器中。参见 Laurant Lemaitre 等人, *Supra*。

发明内容

本发明的一个方面为仿真中使用的模型的自动编译。该方法包括接收模型的描述并自动地将描述转换为根据一个或多个所选定制标准而定制的模型的实现。例如，定制标准可包括一个或多个参数值、仿真期间将要运行的分析和/或算法、模型划分、模型结构拓扑或模型连通性、或模型特征去除。

根据本发明一个方面的模型编译自动为特定仿真需求定制模型实现，以通过模型优化来优化仿真性能。实质上，为仿真定制而

不是为仿真器定制模型实现。因此，与现有的压缩模型编译器不同，例如，其生成中间实现，然后将中间实现重新瞄准不同的仿真器，本发明的一个实施例编译适合由仿真器执行的仿真的模型。即，该实现是否是瞄准特定仿真器的可执行形式还是中间形式与本发明的实施无关。

通过以下结合附图的实施例的具体描述，本发明的这些和其他的特征和优点将变得更加显而易见。

附图说明

图 1 是示出将模型描述转换为适合于目标仿真器的实现以及在仿真器内将该实现与实例参数和模型参数结合起来的传统方法的说明性流程图。

图 2 是示出根据本发明实施例的基于一个或多个参数将一个或多个模型描述转换为一个或多个模型实现并将这种模型实现连同其他参数一起提供给仿真器引擎的方法的说明性流程图。

图 3 是根据本发明实施例的过程可运行于其中的说明性计算机系统的示意图；计算机系统包括可以计算机程序和根据本发明实施例的模型实现编码的计算机可读介质或者可与计算机可读介质进行通信。

图 4 是示出基于定制标准将一个或多个模型描述转换为一个或多个实现的方法的说明性流程图，其中，编译器接收模型描述并且还接收定制标准，以及其中，根据该标准，编译器自动地将模型描述转换为定制的模型实现。

图 5 是图画表示面结型二极管模型的模型的说明性电路图。

具体实施方式

概述

本发明提供了用于定制在仿真期间使用的元件模型的新颖方法和装置。给出下列描述使得本领域中的任何技术人员都能够制造并使用本发明。在特定应用及其需求的上下文中描述了本发明的实施例。提供这些具体应用的描述只是作为实例。对本领域的技术人员来说，很容易地想到对优选实施例的各种修改，并且在不背离本发明精神和范围的情况下，可以将此处定义的一般原理应用到其他实施例和应用中。因此，本发明不限于所示出的实施例，而是符合与此处披露的原理和特征一致的最宽范围。

本发明可以通过提供定制的或适合于特定仿真的模型来改善仿真器性能。可以基于诸如较快的执行时间和减小的内存需求来测量或评定或估算仿真器的性能。基本上，如果模型与仿真匹配，则仿真能够以改善的性能更高效地运行。应该理解，本发明包括一种新的方法：为仿真定制模型，而不是象现有压缩编译器那样为仿真器 API 定制模型。

本发明一个实施例的一个方面是基于一个或多个可应用于仿真的参数值自动定制或设计模型。本发明另一实施例的一个方面是基于将被用在仿真中的诸如一个或多个分析或一个或多个算法或一个或多个划分的标准来自动定制或设计模型。本发明又一实施例的一个方面是基于诸如可应用于仿真的一个或多个参数值，或者可应用于仿真的一个或多个可能的分析或一个或多个可能的算法或一个或多个可能的划分的基于仿真的标准，来自动预编译多个不同版本的模型。在仿真时，可以选择一个或多个预编译模型用于仿真。

例如，根据该新方法，可以通过一个或多个描述来表示单个模型，并且可通过编译器将这些描述与多组模型参数结合起来以生成多个实现。此外，即使给出单个描述和单组模型参数，也可以通过编译器生成为特定情况优化的模型的多个实现。特别地，例如，可以编译为特殊环境优化的版本。例如，可以具有一个具有寄生电阻器的 MOS 模型以及没有寄生电阻器的 MOS 模型的版本；具有源极和大量短路的版本和没有源极和大量短路的版本；用于 n 型器件的版本和用于 p 型器件的版本；具有电容器（用于瞬态）的版本和没有电容器的版本（用于 DC）等。

除了创建比传统方法的单个实现更加被优化的多个实现，该新方法还提供了在实例参数和模型参数之间的划分不需要硬编码（hard code）的优势。换言之，模型描述可以只标识其具有一组参数。然后，模型参数为其值在编译时被指定的任何参数，以及实例参数为其值未被指定的参数。

参数的模型定制

图 2 是示出根据本发明实施例的基于一个或多个模型参数 **206** 将一个或多个模型描述 **202** 转换为一个或多个模型实现 **204**，并将这种模型实现 **204** 连同实例参数 **208** 一起提供给仿真器引擎 **210** 的过程的说明性流程图 **200**。基本上，基于模型参数 **206** 来定制或专门化一个或多个模型描述 **202**。具体地，将一个或多个模型描述 **202** 和模型参数 **206** 提供给编译器 **212**，其中，编译器是将模型描述 **202** 转换成为指定模型参数专门化的模型实现 **204** 的计算机程序。也就是说，模型参数 **206** 被编译为一个或多个模型实现 **204**。每个实现 **204** 能够与在仿真器内与提供的实现 **204** 结合的实例参数 **208** 一起被提供给仿真器引擎 **210**。这种新方法能够产生改善的仿真性能，这是因为例如不需要在内存中传送参数（更好的高速缓存性能）；模型被部分地预估算，消除了完全依赖于指定为模型参数的值的任

何项；以及根据用于确定是否应该执行该代码的测试，可以消除通过模型参数的特定选择变得不必要的代码。

图 3 是根据本发明实施例的能够运行新编译器的说明性计算机系统 **300** 的示意图。计算机系统 **300** 包括一个或多个中央处理单元（CPU）**302**、用户接口 **304**、计算机可读存储介质 **306**、系统总线 **308**、以及用于将 CPU、用户接口、存储器和系统总线连接到一起的一个或多个总线接口。计算机系统 **300** 还包括用于与其他装置 **312** 在计算机网络上进行通信的网络接口 **310**。

可以经由总线 **308** 从接口 **304**、存储器 **306**、或其他装置 **312** 将计算机可读模型描述提供给从存储器 **306** 运行在 CPU **302** 上的编译器。类似地，可以将参数值和/或定制标准提供给编译器。编译器基于参数值和/或定制标准将模型描述转换为模型实现。

如此处所使用的，实现是使用诸如 C 的编程语言或使用被优化以更高效运行的机器语言的模型的专门化。应该理解，实现可以为瞄准特定仿真器引擎的可执行形式，或者它可以为适合于重新瞄准若干不同仿真器引擎中的任何一个中间形式。本质上，在将一个或多个模型描述转换为一个或多个模型实现的过程中，例如，编译器还基于诸如模型参数值的定制标准或基于分析或算法或者其组合来自动地专门化或定制模型。如此处所使用的，转换可以包括下列中的一个或多个：将已知值代入模型方程；预估计模型方程（识别和预估计任何只包括已知常量的子表达式等）；执行代码优化（消去公用子表达式，删除再也不会使用的代码等）以最大化执行速度；如果适当的话，通过删除用于该分析不需要的值的任何公式（例如，DC 分析期间的电荷和电容），瞄准用于特定分析的实现；瞄准用于特定算法的实现，例如，当使用 Samanskii 法则时，消除导数计算；如果适当的话，划分模型。

更具体地，根据本发明的实施例，可将模型参数指定为一组确切值，或者为可用于计算来自其他量的参数值的一组公式，或指定为其组合。特别地，例如，这些其他量可以为其他模型参数、实例参数、新用户定义的实例参数、或仿真器变量或参数。依次考虑每一个。首先假设定义四个参数 α、β、λ和 δ 的模型描述。现在假设模型参数说明包括，

$\alpha = 1$

$\beta = 2*\alpha$

$\delta = \lambda + 1$

没有为 λ 指定值或公式。则 α 和 β 变为取常量 1 和 2 的模型参数，并且它们在编译阶段变为常量，意味着它们的值是固定的，并且它们在编译模型中不会被覆盖。参数 λ 变为实例参数，因此其值可进入到编译模型中。最后，δ 被编译掉并被 λ+1 取代。以这种方式，在使用特定组的三个模型参数值的编译后，以四个参数开始的模型描述被转换为只有一个参数的实现参数。

为了更加灵活，当指定模型参数时，可以指定新的实例参数并且使模型参数成为那些参数的函数。例如，假设使用声明新参数 ε 的参数集来编译上述模型，并将模型参数指定为

$\alpha = \varepsilon$

$\beta = 2*\alpha$

$\lambda = \varepsilon/2$

$\delta = \lambda + 1$

结果的实现将具有一个参数 ε，其不在原始模型的参数中，并且将把作为 ε 函数的值给予原始模型描述的四个参数。仿真器性能将会因为减小的内存需求而改进。在这种情况下，只需要存储 ε，而不是存储α、β、和δ中的每一个。

此外，模型参数集可以根据全局仿真器值来指定模型参数值。这种能力可用于标度（scale）参数，如在 SPICE 多重性因子的情况下，或者全局仿真器值可能为在蒙特卡洛分析期间使用的随机变量。作为进一步的实例，考虑上述模型描述，除了使 ε 不是模型参数集的参数而是全局随机变量。除 ε 的值不会由用户直接指定为实例参数而是被视为随机变量 ε 的值以外，所得到的编译模型将非常类似于上一种情况中的模型。

因此，在本发明该实施例的一个方面，编译器将模型参数编译为模型实现。即，编译器将参数看作在仿真期间不会变化的常量；然后其执行代码优化。例如，考虑由下式给出的电阻器模型：

$$i = \frac{w}{lR_{sh}}v$$

其中，w 是宽度，l 是长度，R_{sh} 是构成电阻器材料的薄层电阻。假设在编译时，w、l、和 R_{sh} 的值已知为 1μm、100μm 和 1kΩ。则，可将模型简化为 $i = 10^{-5}v$。在本发明该实施例的另一方面，编译器实际上至少将某些参数划分为被编译为实现的参数的一个子集和未编译为实现的参数的另一个子集。编译器使用诸如可用的参数值或来自用户提示的可用性的标准来确定将哪个参数划分为实现以及留下哪个作为与实现分开的实例参数。

应该注意，提出的 Verilog-A 和 Verilog-A/MS 扩展本质上不区分实例参数和模型参数。例如，参见 paramset: SPICE *.model* 语句

的 Verilog-A/MS 实现，版本 4，2004 年 2 月 10 日；以及提出的用于压缩建模的 Verilog-A 语言扩展，版本 7，2004 年 4 月 2 日。虽然在特定的现有仿真环境中得出了实例参数和模型参数之间的差别，但是即使在缺少这种差别的情况下本发明的原理也同样适用。具体地，即使通过在这些提议的语言扩展的情况下未被指定为实例或模型参数的参数，编译器仍然可以编译掉特定的参数，并且仍然可以划分将从未被编译掉的参数中被编译掉的参数。

基于其他定制标准的定制

图 4 是示出根据本发明实施例的基于定制或设计标准 **406** 将一个或多个模型描述 **402** 转换为一个或多个实现 **404** 的方法的说明性流程图 **400**。然后，可以将该实现 **404** 与实例参数 **408** 一起提供给仿真器引擎 **410**。基本上，例如，编译器 **412** 接收使用诸如 Verilog-A 的语言的模型描述 **402**。编译器 **412** 还接收定制标准 **406**。可在类似于图 2 的计算机系统上运行的编译器 **412** 根据该标准自动地将高级模型描述转换为定制模型实现。本质上，对计算机系统进行编程以将高级模型描述转换为基于专业化标准而专门化的实现。即，编译器根据它接收的一个或多个专业化标准不同地定制实现。

实现 **404** 可以是瞄准特定仿真引擎的可执行实现，或者可以为中间实现。定制标准可涉及将在仿真期间执行的一个或多个分析或算法。定制还可以基于仿真的划分、仿真期间的并行处理、特征去除、支持算法去除、元件连接细节、或将对模型的结构拓扑所作的改变，以列举几个实例。

用于分析的定制

根据本发明的实施例，将由仿真器执行的分析是一个定制标准。分析是用于在给定一组假设或条件的情况下预测电路行为的过

程。典型的仿真器可用于执行多种不同形式的分析（例如，瞬态分析、DC 分析、AC 分析，谐波平衡分析、蒙特卡洛分析、失配分析、和参数分析，只列举几个实例）中的任意一种。通常，可以通过为将要执行的特定分析定制该模型来改善给定模型的仿真器性能。因此，模型的某些部分可能对于特定分析更加重要，而对于其他分析则不太重要。根据本发明的一个方面，可以根据在仿真期间使用的分析来使模型专门化。

对于 DC 分析例如，，通过将模型的高级（例如，HDL）描述转换为只包括模型的静态（电阻）部分的模型的实现，可以改善仿真器性能。可以改善 DC 分析期间的仿真器性能，这是因为传统方法估计不需要的动态（电容或电感）部分然后将其抛弃，或者将其放置在一系列 'if' 语句中，在每次估计模型时都不得不估计每个条件。这通常比模型动态部分的估计稍微省力，但它不一定可以忽略。

基本上，根据本发明的实施例，运行在计算机系统上的编译器接收将为 DC 分析定制模型的指示。编译器自动地将计算机可读介质中的模型描述转换成为 DC 分析而定制的实现。编译器将定制的实现存储在可由仿真器访问的计算机可读介质中。

对于 AC 分析，例如，通过将模型的高级（例如，HDL）描述转换为只包括导数项（电阻、电导、电容、电感）的模型的实现，可以改善仿真器的性能。在一个实施例中，运行在计算机系统上的新编译器将模型转换为只包括导数项的实现。该实现被放置在计算机可读介质中供仿真器使用。因为可以避免诸如电流、电荷、和通量的不需要量的计算，所以可以改善仿真性能。

对于包括瞬态分析的仿真，例如，在仿真期间需要模型的静态（电阻性）和动态（电容性）部分。通过消除确定是否需要模型的

每一段的条件来改善性能。即，在一个实施例中，运行在计算机系统上的编译器将模型转换为不包括这些不必要条件的实现。换言之，传统技术通常为模型编写用于所有分析的“估计”函数。这种现有的代码以相当多的额外花费计算比诸如 AC 和 DC 的简单分析所需要的更多计算。所以经常将条件添加到现有代码中以确定这些简单分析是否正在运行，并且跳过模型的不需要部分。然而，通过现有代码估计这些条件为瞬态分析添加了开支，这是因为所有这些条件都被估计，即使它们没有提供保存（saving）。

对于包括谐波平衡分析的仿真，例如，通过将模型描述转换为在整个波形上估计模型而不是在单一时间点上估计模型的模型的实现，可以改善仿真器性能。在一个实施例中，运行在计算机系统上的新编译器将模型转换为在整个波形上估计模型的实现。相反，在瞬态分析期间，仿真器在一个时间点估计所有模型，然后移动到下一个点。然而，在谐波平衡仿真器中，可以在移动到下一个模型之前在所有点处估计单个模型。通过增加引用的区域性，这样做提供了更好的高速缓存性能。可选地，例如，可以转换高级模型描述以在频域和时域之间进行划分。

对于包括蒙特卡洛分析的仿真，例如，通过将模型描述转换为随机变量被看作实例参数的模型的实现，可以改善仿真器的性能；通常，在传统技术中，随机变量被看作模型参数，所以即使只有随机变量一直在改变，每个实例也都需要模型参数的完整集合。因此，在一个实施例中，运行在计算机系统上的编译器将模型转换为特定模型参数被视为实例参数的实现。

例如，蒙特卡洛分析可用于评定实例参数值中变化的影响。参数值可以以某种统计方式而变化，并且可以运行一系列仿真以执行蒙特卡洛分析来估计这种统计参数值变化的影响。例如，半导体晶

体管的电特性可由于诸如氧化物厚度或诸如钻蚀的实例参数值的变化而变化。

类似地，例如对于包括失配分析的仿真，通过将模型描述转换为被作为实例参数的模型的实现，可以改善仿真器性能。

同样地，例如对于包括参数分析的仿真，通过将模型参数转换为被作为实例参数的模型的实现，可以改善仿真器性能。

最后，例如包括拐角分析的仿真，在其最坏情况下的拐角改变一组参数。在该分析中，可能存在数百个正在变化的参数，单独改变开支会很大，所以只考虑所选的‘拐角情况’，例如最慢的性能或最快的性能。在第一种（较慢）情况下，设置参数值以得到最小速度，在第二种（较快）情况下，设置参数值以得到最大速度。通过每个拐角创建一个实现，可以改善仿真性能。因此，在一个实施例中，运行在计算机系统上的编译器将模型的高级描述转换为每个拐角的实现。

用于算法的定制

根据本发明的一个实施例，在仿真期间执行的算法是另一种定制标准。算法是用于执行分析的特定程序。例如，典型的仿真器可用于执行多种不同算法（例如，Samanski 法则、牛顿法、松弛法、Kyrlov 法、或数值积分法）中的任意一种。存在若干可能的数值积分法，例如，向前或向后欧拉、梯形法则、向后差分公式和 Runge Kutta，列举几个实例。通常，通过为将要执行的特定算法定制该模型，可以改善给定模型的仿真器性能。

对于 Samanski 法则，例如，通过将模型的高级描述（例如，HDL）转换为不包括导数的模型的实现，可以改善仿真器性能。

Samanski 法不使用导数，所以计算它们将会很费劲。在某些情况下，例如，达到模型估计时间的一半或更多可以只用于导数。通常在 DC 和瞬态分析期间采用 Samanski 法。因此，在一个实施例中，新编译器将高级模型描述转换为省去导数的实现，以便为 Samanski 法则定制模型。

对于牛顿法，例如，通过将模型的 HDL 转换为确实包括导数的模型的实现，可以改善仿真器性能。如果一个实现既支持 Samanski 法又支持牛顿法，则或者只计算导数（这对于牛顿是最佳的，但对于 Samanski 不是最佳），或者你放入测试以除去它们，（这对于任何一个都不是最佳的）（该测试是额外的开销）。如果你有两种实现，每种一个，每一个都会是最佳的。牛顿法通常在 DC 和瞬态分析期间被采用。

对于松弛法，例如，通过将模型的高级描述（例如，HDL）转换为被划分的模型的实现，可以改善仿真器性能，每一个分区在不同的时刻被估计。在一个实施例中，运行在新编译器上的编译器将高级模型描述转换为被划分的实现，使得可在不同的时刻对不同的分区进行划分。如果模型跨越两个分区，则每个分区只需要估计该模型的一部分。例如，通常划分 MOSFET，使得其栅极在一个分区中，而其沟道在另一个分区中。可以创建模型的多个实现，使得一个实现只包括栅极模型，而另一个只包括沟道模型。因此，当估计分区时，只需要估计影响该分区的模型的那部分。松弛法通常在瞬态分析期间被采用，特别在关注速度胜于关注精度的大电路中。

对于 Krylov 法，例如，通过将模型的高级描述（例如，HDL）转换为执行其矩阵向量乘积计算部分的模型的实现，可以改善仿真器性能。因此，在一个实施例中，新编译器将高级模型描述转换为执行其自身的矩阵向量乘积计算部分的实现。传统地，通常估计所有模型，然后对整个电路计算矩阵向量乘积。相反，每个模型可以

采取其矩阵向量乘积部分的计算。这可以改善高速缓存性能，意味着仿真将会更快，这是因为由模型贡献的矩阵向量乘积部分已经在高速缓存中，所以避免了将其从主存储器中取出的时间。松弛法通常在诸如谐波平衡和打靶法的 RF 分析的执行期间被采用。

对于数值积分法，例如，通过将模型的高级描述（例如，HDL）转换为执行其与任何时间导数的有限差分近似部分的模型的实现，可以改善仿真器性能。传统地，通常在估计所有模型之后执行那些计算。但是在这种情况下，该实现可承担此项工作，这将很可能改善性能，这是因为在数据仍然在高速缓冲存储器中时它能执行该近似。数值积分法通常在瞬态分析期间被采用。因此，在本发明的一个实施例中，新编译器将模型描述转换为执行其自身近似的实现。

用于划分的定制

划分可用于将设计的大行为描述分解为若干小描述。计算机辅助设计环境中的划分是将目标聚合成组使得可相对于一组设计约束优化给定目标函数的任务。

根据本发明的一个实施例，划分用于仿真的模型是另一个定制标准。对于某些仿真，通过将模型的高级描述（例如，HDL）转换为模型的划分实现，可以改善仿真器性能。通过划分（和松弛一样），模型可以跨越多个分区。因而，当估计特定划分时，只需要估计模型的一部分。基于松弛的定时是使用划分、松弛和近似模型来实现仿真速度增加的仿真器的特定分类的实例。例如，Naasada's HSIM 和 Cadence's UltraSim 使用划分来增加仿真速度。对模型的划分实现进行编译而不是使仿真器进行划分是有好处的。例如，通常，当仿真器划分电路时，元件模型将停止跨越分区。然而，当估计一个分区时，不需要估计处于另一分区中模型的那个部分。划分模型节省了仿真的工作，只对模型需要仿真的那个部分进行仿真。在过去，

通过手工划分模型，但那是低劳动强度和易于出错的工作。因此，在本发明的一个实施例中，运行在计算机系统上的新编译器将高级模型描述转换为划分实现。

用于并行估计的定制

在一些情况下，通过在仿真期间将模型的 HDL 转换为适合于并行估计的模型的实现，可以改善仿真性能。假定仿真器引擎将运行在多个处理器上。因此，在一个实施例中，新编译器划分该实现，使得在单独的处理器上可以同时对分区进行估计，或者提供允许模型实例被同时在单独的处理器上估计的实现。基本上，编译器对模型代码运行依赖性分析，以确定它依赖于什么，旨在将每个独立代码流放在不同的线程中以在独立的处理器上进行估计。

通过特征去除的定制

在一个实施例中，新编译器将描述转换为省略了被指定用作定制标准的特征的实现。特征包括用于 DC 分析的电容器或电感器或模型仿真不需要的同伦参数。例如，如果基于每一个分析进行特征去除，则用于特征分析的定制可被仿真为用于 AC 方法或用于 DC 分析的定制。然而，如果可以为所有分析进行定制。例如，同伦参数是可以简单地在特定仿真器上除去的参数。并且可能碰巧一个仿真器只可以支持很少的分析，因此在一个提供了非常多分析的仿真器上看起来是用于分析的定制在另一个支持非常少分析的仿真器上将看起来是特征去除。

通过支持算法去除的定制

例如，通过支持算法去除的定制包括涉及公差检查或数值积分的算法的去除。一些仿真器使支持算法与模型分离，这会由于减小

的高速缓存效能而降低性能。专用的实现可将这些算法加入到需要它们的那些实现中。相反，一些仿真器将支持算法做成模型的一部分。这增加了高速缓存效能，但是在某些情况下（某些算法、某些分析、等），不需要这些支持算法。在这种情况下，将支持算法做成模型的一部分的事实降低了速度，这是因为在那里它们始终都被估计。可以建立两个专用实现，一个具有支持算法，一个没有。因此，可以适当地使用最佳的一个。因此，在一个实施例中，编译器将模型描述转换为省略了通过定制标准指定的支持算法的实现。

通过连接模型元件方式的定制

例如，可以根据连接模型的方式（例如，终端连接在一起或接地）实现定制。可选地，可以简化模型（例如，可以消去变量），实现更快的执行。因此，在一个实施例中，运行在计算机系统上的编译器将模型转换为具有由定制标准指定的修正和简化的实现。

通过模型结构拓扑变化的定制

例如，可以根据结构拓扑的变化（例如，从模型中去除寄生电阻器或支持节点）实现定制。可以通过抛弃不希望具有显著影响的模型的昂贵部分，实现结构拓扑的变化以获得仿真速度。可能基于检查电路或在最终用户的指导之下作出这样的决定。因此，在一个实施例中，编译器将模型转换为具有通过定制标准指定的结构拓扑的实现。

多个不同的预编译实现的生成

应该理解，可以基于如参照图 2 所描述的不同参数值或基于如参照图 4 所描述的其他定制标准，通过自动预编译多个定制的模型实现来实现本发明的原理。因此，在本发明的一个实施例中，可以

创建多个预编译模型实现，并且用户或仿真引擎（例如，通过编译器指令）可以指定在仿真中使用哪一个预编译模型实现。

HDL 模型描述到不同模型实现的转换实例

以下的说明包含特定模型描述的多个实现的实例。使用面结型二极管模型作为说明性实例。

面结型二极管

图 5 是以图画形式表示面结型二极管模型的说明性电路图。

理想的二极管是允许电流沿一个方向流动而不能沿另一个方向流动的元件。面结型二极管是可以使用半导体工艺制造的近似具有这种行为的元件。它是具有以下特性的非线性电元件，

$$i_j = I_s\left(e^{v/v_T} - 1\right), \tag{1}$$

$$C_j = \frac{C_{j0}}{\sqrt{1 - \frac{v}{\phi}}}. \tag{2}$$

$$C_d = \tau g_d \quad 其中 \quad g_d = \frac{di_j}{dv} = \frac{I_s}{V_T} e^{v/v_T} \approx \frac{i_j}{V_T}, \text{以及} \tag{3}$$

$$C_t = C_j + \frac{dq_c}{dt}. \tag{4}$$

为了实施使用 Verilog-A/MS 的模型，应该根据分支电压和流量将其用公式表示为结构关系。这已经是用于模型电阻性部分（1）的情况，而不是用于电容性部分（4）。为了避免电荷守恒问题，必须根据电荷和电压用公式表示非线性电容器的结构关系。为了实现这点，（4）的电容相对于电压积分以得到电荷，

$$q = \tau_f i_j - 2C_{j0}\phi\sqrt{1-\frac{v}{\phi}}. \quad (5)$$

然后，总的二极管电流由（1）和（5）的结合产生，

$$i_d = i_j + \frac{dq_c}{dt}. \quad (6)$$

实现这些方程的 Verilog-A/MS 模型在列表 1 中给出。

列表 1

*用于面结型二极管的 Verilog-A/MS 模型（在实践中不应该使用该模型，因为它在 v > **SYMBOL HERE** 时失效）。*

```
//面结型二极管
`include "disciplines.vams"
module diode (a, c);
        parameter real is=10f from (0:inf);        //饱和电流(A)
        parameter real tf=0 from [0:inf);          //正向转换时间(s)
        parameter real cjo=0 from [0:inf);         //零偏置压结电容(F)
        parameter real phi=0.7 exclude 0;          //内置结电势(V)
        inout a, c;
        electrical a, c;
        branch (a, c) res, cap;
        real qd;

        analog begin
                I(res) <+ is*(limexp(V(res)/$vt) - 1);
                qd = tf*I(res) - 2*cjo*phi*sqrt(1 - V(cap)/phi);
                I(cap) <+ ddt(qd);
        end
endmodule
```

面结型二极管模型的 DC 分析定制

在 DC 分析中，不需要模型的电容性部分。然而，需要来自模型电阻性部分的电流及其导数。所以，为 DC 分析设计的实现将可能提供

$$i_j = I_s\left(e^{v/V_T} - 1\right) \text{ 以及} \tag{7}$$

$$\frac{di_j}{dv} = \frac{I_s}{V_T}e^{v/V_T} \approx \frac{i_j}{V_T} \text{ 以及} \tag{8}$$

在该 DC 分析定制的实现中，从不对计算 qd 和 I(cap)的方程进行估计，也不估计相关导数，意味着模型的估计大约比其他方法快两倍。

面结型二极管模型的 AC 分析定制

在 AC 分析中，不需要电流和电荷，但是仿真器必须具有电流和电荷相对于电压的导数。所以为 AC 分析设计的实现将可能提供

$$\frac{di_j}{dv} = \frac{I_s}{V_T}e^{v/V_T} \approx \frac{i_j}{V_T} \text{ 以及} \tag{9}$$

$$\frac{dq_c}{dv} = C_j + C_d = \frac{C_{j0}}{\sqrt{1-\frac{v}{\phi}}}. \tag{10}$$

在该 AC 分析定制的实施中，计算导数 dI(res)/dV(res)和 dI(cap)/dV(cap)，而不是 I(res)和 I(cap)本身。这使得模型估计增加大约 2x 的速度。

瞬态分析

在瞬态分析中，仿真器必须具有电流、电荷、以及电流和电荷相对于电压的导数。所以为瞬态分析设计的实现将可能提供

$$i_j = I_s\left(e^{v/v_T} - 1\right), \tag{11}$$

$$q_c = \tau_f i_j - 2C_{j0}\phi\sqrt{1-\frac{v}{\phi}}, \tag{12}$$

$$\frac{di_j}{dv} = \frac{I_s}{V_T}e^{v/v_T} \approx \frac{i_j}{V_T} \text{ 以及} \tag{13}$$

$$\frac{dq_c}{dv} = C_j + C_d = \frac{C_{j0}}{\sqrt{1+\frac{v}{\phi}}} + \tau\frac{di_j}{dv}. \tag{14}$$

在该瞬态分析中，定制的实现，所有项都被估计，并且不存在确定是否应该估计某一项的条件，所以该模型比如果提供条件的情况运行的要快一点。

3.4 Samanski 法

如果将 Samanski 法应用于 DC 或瞬态分析，则不需要导数。所以在瞬态分析中，实现将可能提供

$$i_j = I_s\left(e^{v/v_T} - 1\right) \text{ 以及} \tag{15}$$

$$q_c = \tau_f i_j - 2C_{j0}\phi\sqrt{1\frac{v}{\phi}}. \tag{16}$$

在 DC 分析中，将只提供（16）。

在该 Samanski 算法定制实现中，计算导数 dI(res)/dV(res)和 DI(cap)/dV(cap),而不是 I(res)和 I(cap) 本身。这使得模型估计中增加大约 2x 的速度。

数值积分法

在瞬态分析中数值积分法将模型中的时间导数转换为离散时间近似。向后欧拉法使用以下近似:

$$\frac{dx(t_k)}{dt}=\frac{x(t_k)-x(t_{k-1})}{t_k-t_{k-1}}. \tag{17}$$

为特定数值积分法设计的实现将可能提供以下方程:

$$i_j(v_k)=I_s\left(e^{v_k/v_T}-1\right)+\frac{1}{t_k-t_{k-1}}\left(t_f i_j(v_k)-2C_{j0}\sqrt{1-\frac{v_k}{\phi}}-\left(t_f i_j(v_{k-1})-2C_{j0}\phi\sqrt{1-\frac{v_{k-1}}{\phi}}\right)\right) \tag{18}$$

以及其相对于 v_k 的导数 (19)

在该数值积分算法定制的实现中，ddt()函数在该模型中被扩展，尽管以前它被假定为由仿真器指配。如较早所描述的，这可以通过增加高速缓存的效能而在某种程度上改善速度。

应该理解，根据本发明优选实施例的上述描述和附图只是本发明原理的说明，在不背离本发明范围和精神的情况下，本领域技术人员可以对本发明进行各种修改。

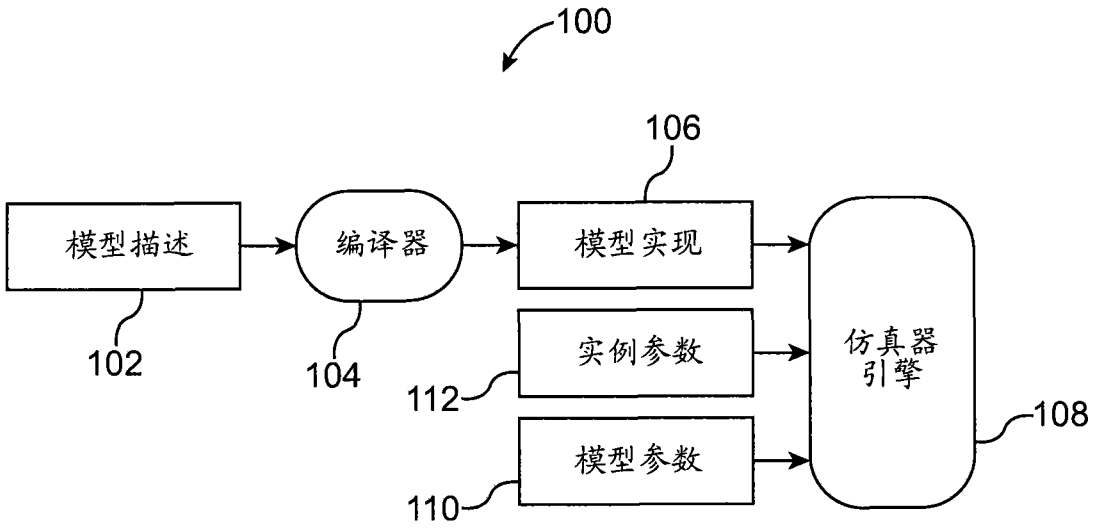


图1
（现有技术）

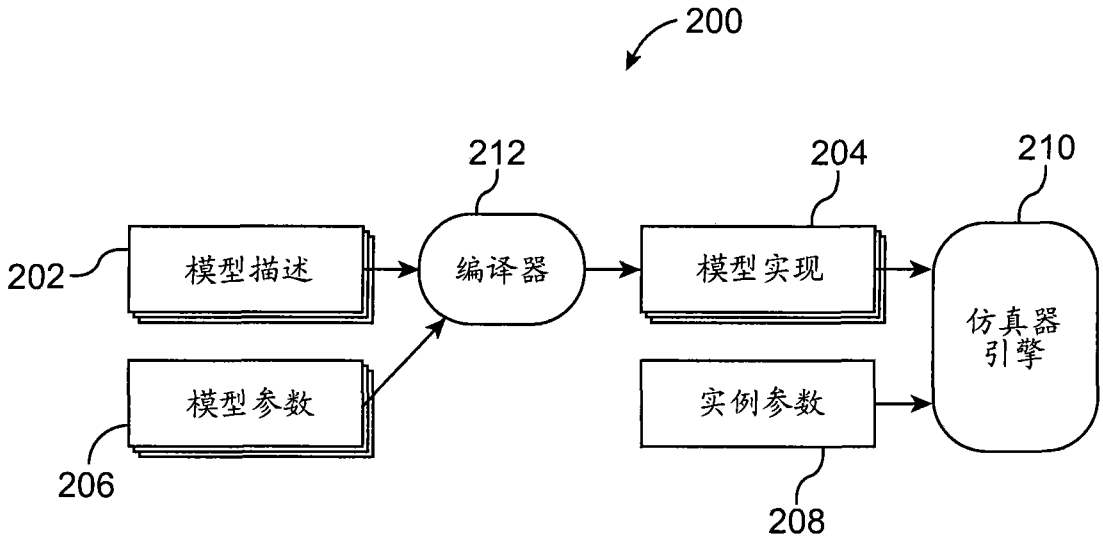


图2

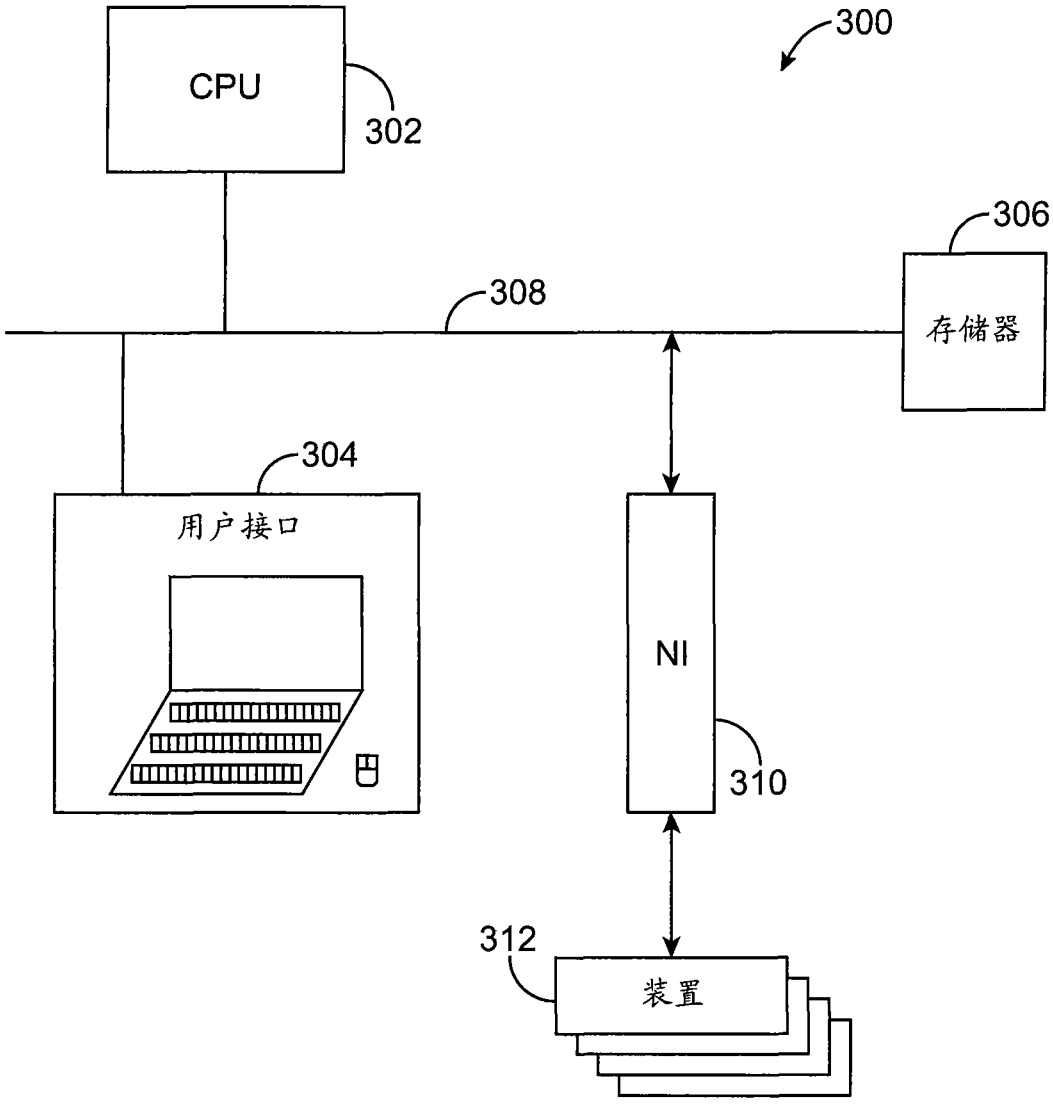


图3

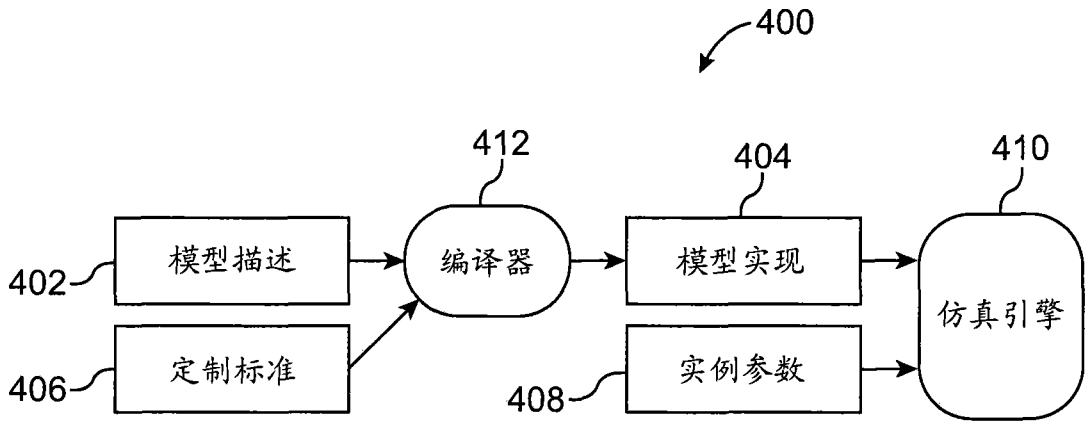


图4

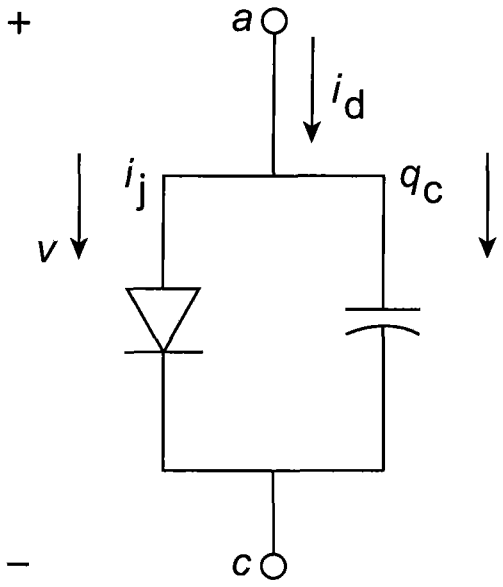


图5