CN43-1258/TP
ISSN 1007-130X
计算机工程与科学
COMPUTER ENGINEERING & SCIENCE
2007 年第 29 卷第 11 期
Vol. 29, No. 11, 2007

文章编号:1007-130X(2007)11-0069-03

粗粒度可重构阵列上的布局布线算法*
Placement and Routing Algorithms for Coarse-Grained Reconfigurable Arrays

左艳辉,窦 勇,徐进辉
ZUO Yan-hui, DOU Yong, XU Jin-hui
(国防科技大学计算机学院,湖南 长沙 410073)
(School of Computer Science, National University of Defense Technology, Changsha 410073, China)

摘 要:开发粗粒度可重构阵列之上的映射工具是把应用算法正确有效地映射到可重构硬件上,并使算法在可重构硬件上正确高效运行的关键之所在。因此,我们设计并实现了映射工具。本文介绍了映射工具的设计和实现过程,并给出了实现中的关键技术——布局。最后,本文还就几个测试程序给出了映射工具的映射结果。测试结果证明,布局算法的结果正确且优化,映射工具的设计合理,功能无误。

Abstract: It is very important to develop a mapping tool for correctly mapping algorithms on hardware and making them execute effectively. So we design and implement a mapping tool. This paper introduces the design of a mapping tool in detail, and gives two key points of implementation: placement and routing. In the end, we show the output of the mapping tool for several example programs. Test results show that the result of placement is correct and optimum, and the design of the mapping tool is reasonable and its functions are correct.

关键词:粗粒度可重构阵列;映射;布局
Key words: coarse-grained reconfigurable array; mapping; placement
中图分类号:TP301 文献标识码:A

1 引言

当前,越来越多的应用领域同时要求计算的高性能和灵活性。由于可重构计算在获得 ASIC 的高性能的同时又保持了很好的软件灵活性,所以可重构计算成为最有优势的计算形式。而粗粒度可重构体系结构则是专门为可重构计算所设计的。

对于粗粒度可重构体系结构而言,只有把应用程序合理有效地映射到可重构的硬件资源上,可重构的硬件资源才能被充分利用,从而提高可重构系统的整体性能。然而,这个映射过程如果采用手工映射,则需要程序员对他所采用的可重构系统的结构有相当的了解,而且需要花费程序员很多时间和精力。所以,为了能更快、更容易地把高级语言所编写的应用程序映射到可重构硬件资源上,就极其需要一种通用的映射工具。它使得一般的应用程序员都能很轻松地利用可重构硬件,更快更容易地得到映射结果。

本文介绍了一种粗粒度可重构阵列上的映射工具。映射工具把由 C 语言程序转化得到的数据流图经过布局布线后转化为可直接配置在可重构硬件上的二进制配置流。映射工具避免了手工映射的复杂性和低效性,使得映射实现自动化。

布局是映射过程中的关键环节,布局的好坏直接决定了映射结果是否高效。因此,本课题对布局算法进行了较深入的研究,提出了对数据流图构造树来生成初始布局和用改进的模拟退火算法来优化布局的思想。

2 粗粒度可重构阵列简介

2.1 粗粒度可重构阵列

可重构阵列是可重构处理器的重要组成部分之一。可重构阵列是由多个处理单元构成的阵列。处理单元 PE 一般有两种类型:用于循环控制与存储访问控制的 MPE 和用于计算任务的 CPE。所有处理单元都和路由器相连,路

* 收稿日期:2007-03-29;修订日期:2007-07-09
基金项目:国家自然科学基金资助项目(60633050)
作者简介:左艳辉(1983-),女,湖南永州人,硕士生,研究方向为高性能体系结构。
通讯地址:410073 湖南省长沙市国防科技大学计算机学院硕士二队;Tel:(0731)4574605;E-mail:zuoyanhui216@sina.com
Address: School of Computer Science, National University of Defense Technology, Changsha, Hunan 410073, P. R. China

由器之间又相互连接构成一定拓扑结构的互连网络。可重构阵列的第一行都是 MPE，MPE 与外部存储器相连，从外部存储器中读出数据或把结果数据存入外部存储器。其它各行都是 CPE，用于对数据进行各种运算操作。每个 PE 都对应一个路由节点。整个阵列是通过网络路由节点把 PE 和网络连接在一起。

数据网络用于传递 PE 之间需要的数据（见图 1），数据在数据网络中是由上至下的流动趋势。对于每个 PE，与其相对应的路由节点相连接，并通过路由节点与其它 PE 相连接。要把 C 语言循环程序映射到可重构阵列上，首先需要把 C 语言循环程序抽象为一个数据流图。

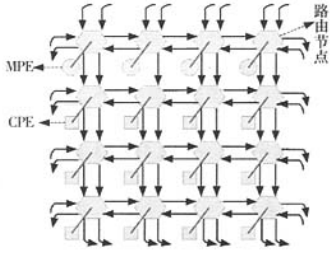

图 1　数据网络的拓扑结构

2.2　数据流图

数据流图是由 C 语言循环程序转化得到的。源程序中的循环控制记录在 MPE 的配置信息中，存储体中数据的读入也由 MPE 来控制。而循环体内的代码则由数据流图来刻画。其中，程序中的输入数据和最后的输出数据用存储节点 MPE 来表示。对数据进行的操作转化为操作节点 CPE，各节点之间的数据流动关系转化为边。入边表示数据要经过此节点的操作或存储，出边表示操作节点操作结果的流出或数据从存储节点流出。将源程序转化为数据流图，如图 2a 所示。

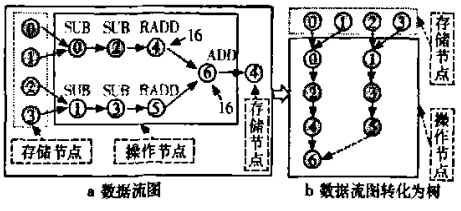

图 2　数据流图及其转化

把数据流图映射到可重构阵列上就是要把数据流图中的存储节点和操作节点分别映射到可重构处理单元阵列上的 MPE 和 CPE 上，并通过数据网络把各 PE 之间的数据流动关系表示出来。这就是布局和布线的工作。

3　映射工具的设计和优化

我们所采用的映射方法是一种同时布局布线方法，即在布局的同时进行布线。只有布线成功了，才能说明布局也成功，才能终止布局。这种同时布局布线的方法可能比单独的布局布线能找到更好的解决问题的方案。布局过程中还融合了模拟退火算法的思想，以便能得到一个相对较优的解。

3.1　布局

3.1.1　布局算法的描述

布局算法以数据流图为输入，输出则是各节点在可重构阵列上的映射结果。

数据流图可以抽象为节点的集合和边的集合。其中，节点的集合为 $V=\{V_0,V_1,V_2,\cdots,V_n\}$，$V_i$ 表示数据流图中的一个存储节点 MPE 或操作节点 CPE。节点间的数据传递关系用有向边来表示，边的集合 $E=\{E_0,E_1,E_2,\cdots,E_m\}$，其中 E_i 是连接第 V_{i-1} 号节点到第 V_i 号节点的边。而一个规模为 $M\times N$ 的可重构阵列可以抽象为处理节点 PE 的集合 P 和数据连接通道的集合 C。把数据流图映射到处理单元阵列上，也就是为数据流图的节点集合 V 中的各个节点在处理节点集合 P 中找到一个处理节点相对应，并保持节点之间的连接关系不变。数据流图的信息用我们定义的一种描述语言来抽象表示出来，并作为布局算法的输入。

布局算法的框图如图 3。首先，从输入的数据流图中读取信息到 CPE 和 MPE 的数据结构中；然后，按照数据的流动规律构造树；接着，从树中逐个把 CPE 取出来，并为每个 CPE 在阵列中分配一个位置。初始布局就完成了。初始布局完成后，把布局的结果转化为布线接口进行布线。之后还需用模拟退火算法对初始布局进行优化。

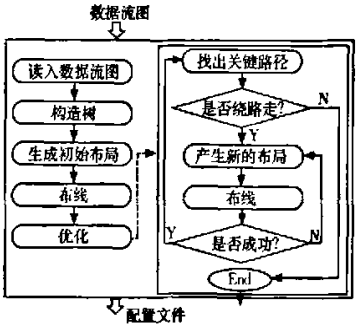

图 3　布局算法框图

3.1.2 利用构造树生成初始布局

为了能生成一个比较优化的初始布局，需按照数据的流动规律构造树。数据流图转化为树的例子如图 2b 所示。树的结构和 PE 的结构分别用图 4a 和图 4b 来表示。

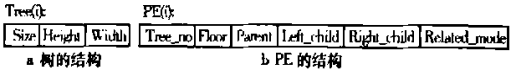

图 4　树和 PE 结构

树结构中各个域的含义：规模 $Size_i$：第 i 棵树中节点个数之和；高度 $Height_i$：第 i 棵树的高度，也就是第 i 棵树中所有节点中层数最大的节点的层数，即 $Height_i=\mathrm{Max}(Floor_j)(j\in 0,1,\cdots,(Size_{i-1}))$，$Floor_j$ 是第 j 号节点的层数；宽度 $Width$：树的每层所含的节点数就是这一层的宽度，而 $Width$ 是树中各层宽度最大的那层的宽度。

PE 结构中各个域的含义：$Tree\_no$：此 PE 所属树的编号；$Floor$：此 PE 位于这棵树的第几层；$Parent$：此 PE 的父节点的编号；$Left\_child$：此 PE 的左子节点；$Right\_child$：此 PE 的右子节点；$Related\_node$：如果此 PE 有两个源节

点，则它的第二个源节点为 *Related_node* 域的值，否则该域为空。

构造树的步骤如下：

(1)从 MPE 开始，以每个 MPE 为根节点构造一棵树，若 MPE 的编号为 i，该 MPE 的 *tree_no* 域置为 i，并置这棵树的 *Size*、*Height* 为 1。

(2)按编号的大小顺序遍历所有的 CPE。对每个 CPE 进行第(3)、(4)步。

(3)检查 i 号 PE 的 *Parent* 域，若为空，则把 i 的第一个源节点 j 置为 i 的 *Parent* 的值，并把 $Tree\_no_j$ 赋给 $Tree\_no_i$，$Floor_j+1$ 赋给 $Floor_i$，同时编号为 *Tree_no* 的树的 *Size* 加 1，如果 $Floor_i$ 大于编号为 *Tree_no* 的树的 *Height*，则把 $Floor_i$ 赋给编号为 *Tree_no* 的树的 *Height*；如果 j 的 *Left_child* 域为空，则修改 j 的 *Left_child* 域为 i；否则修改 j 的 *Right_child* 域为 i；

(4)另外，如果 i 号 PE 有第二个源节点 k，则把 k 置为 i 的 *Related_node* 的值。

下面介绍两个概念：

(1)关联度：一个节点的关联度是跟该节点有直接数据传递关系的节点的个数。计算节点的关联度的函数是：$W=M+N$。其中，M 为该节点的源节点的个数，N 为该节点的目的节点的个数。计算第 i 棵树的关联度的函数是：$R_i=in\_R_i+out\_R_i$。其中，内部关联度为 $in\_Ri=\sum_{j=1}^{n} w_j$，n 为第 i 棵树所包含的节点的个数，$w_j=in\_M+in\_N$ 为第 i 棵树中第 j 个节点的内部关联度，其中 *in_M*、*in_N* 分别为 j 的源、目的节点的个数，源、目的节点都与 j 属于同一颗树；$out\_R_i=\sum_{j=1}^{n} w'_j$ 为外部关联度，$w'_j=out\_M+out\_N$ 为第 i 棵树中第 j 个节点的外部关联度，其中 *out_M*、*out_N* 分别为 j 的源、目的节点的个数，源、目的节点都与 j 在不同的树中。

构造完树之后，根据上面的公式分别计算出每个 *PE* 的关联度和每棵树的内外关联度。跟树有外部关联的节点布局的好坏是影响数据流交叉的重要因素。因此，利用树的 *Size* 和 *Width* 以及树的外部关联度及和树有外部关联的节点的位置来考虑树的布局范围，可以最大限度地避开数据流的交叉，从而提高布局的可布线率及布局的质量。

(2)初始布局的生成：

①根据树的棵数和树的 Size、Width 把可重构阵列划分为大小不同的几个块，每棵树都有一个属于自己的块，便于把各棵树中的节点分配到该块上。若 $\sum_{i=1}^{MPE\_total} Width_i \leqslant$ 阵列宽度，则 i 号树的分配范围为 $(\sum_{j=1}^{i-1} Width_j - 1, \sum_{k=1}^{i} Width_k)$；否则，若 $\sum Width_i >$ 阵列宽度，则根据[$(Size_i/total)\times$阵列宽度]的值来分配列。

②根据树的 *Height*，决定树中节点按照数据流动关系呈垂直或水平方向来分配位置。当 $Height \leqslant$ 阵列高度时，这棵树的节点可以完全按照数据流动关系呈垂直方向为节点分配位置；$Height >$ 阵列高度时，则需要考虑在某一层时按照数据流动关系呈水平方向为节点分配位置。

③为每个节点和每棵树都计算出它们的关联度，即得出树与树之间的连接关系、树中各节点间的连接关系。把关联度大的两棵树相邻放置，而同一棵树中关联度最大的节点需要优先放置。

按照构造的树的结构来为节点在可重构阵列上分配位置，可以在全局的角度统筹分配处理单元的同时也兼顾不同形状的数据流图的映射，在一定程度上依照数据流动规律生成初始布局。

3.1.3 利用模拟退火算法来改进

之后采用模拟退火算法来对结果进行优化。模拟退火算法以初始布局作为输入，关键路径的长度是衡量此次布局是否优化的主要标准。

算法有两个主要概念：关键路径和绕路走的路径。

关键路径是所有路径中路径长度最长的路径。路径长度是一个数据流要经过的路由节点的节点数和 *PE* 数之和。路径长度的函数为：$Length=2+i+\sum n_i$，其中 2 为取出数据和存储数据的 MPE，i 为此数据需要经过 CPE 的个数，而 n_i 表示的是此数据从该路径上第 $i-1$ 个 PE 传递到第 i 个 PE 时所经过的路由节点的个数。

绕路走的路径的定义：对于任意有数据传递关系的两个节点 V_1 和 V_2，它们在此次布局中的坐标位置分别是 (x_1,y_1)，(x_2,y_2)，则它们的曼哈顿距离为 $Distance=|x_1-x_2|+|y_1-y_2|$，而这两个节点的路径长度则为：$L=2+i$，其中 2 为源 PE 和目的 PE，$i$ 为此数据从源 PE 传送到目的 PE 需要经过的路由节点的个数。如果 $L>Distance$，则这两个节点的路径是绕路走的。

关键路径影响程序在可重构阵列上运行的效率。所以，映射结果是否能达到高效，主要看关键路径的长短。当得到一个布局时，要根据下面两点进行分析和优化：(1)找出绕路走的路径，对于绕路走的路径，在不增加其他绕路走的路径外，调整布局，使其不绕路走；(2)关键的任务是使关键路径的路径长度尽量达到最小。如果不存在绕路走的路径，则在一定程度上说明关键路径的长度达到了最小值。

根据前面的分析，我们借用模拟退火算法的基本思想，并结合我们的具体情况设计了下面的优化算法(见图 3)：

(1)通过遍历所有路径，找出当前布局关键路径。如果当前布局的关键路径长度比我们所记录下的最优布局的关键路径长度短，则把此次布局作为最优布局。

(2)判断关键路径中数据从一个节点到另一个节点所走的路径是否是最短的路径。如果关键路径中不存在绕路走的情况，则布局已经是优化的，布局过程完成，转到(5)；否则，如果存在绕路走的情况，则转到(3)。

(3)通过移动关键路径上的绕路走的节点的位置来生成一个新的布局，移动节点时先把离该节点曼哈顿距离最近的空闲的处理节点找到，然后把该节点放在这个位置。

(4)用新的布局布线，如果布线不成功，则转到(3)，再另外生成一个布局；否则，记录下这个新的布局，并把这个新的布局置为当前的布局，并转到(1)。

(5)优化布局的过程完成。把布局和布线的结果翻译成二进制格式，生成直接配置硬件的配置流。

3.2 布线

所采用的布线算法是一种经典的布线算法——Lee 算法。Lee 算法能在一个有障碍的网格中寻找源目的节点的路径。如果网格中的任意两个节点之间的最短路径存在，Lee 算法就能把它们之间的最短路径找出来。为了适应可重构体系结构的数据网络的结构，我们对 Lee 算法做了一些改进。

4 测试结果

我们用 C 语言实现了映射工具的所有模块，并且采用以上的测试程序进行了测试，测试结果如表 1 所示。从测

(下转第 75 页)

独立通道的生物认证算法。

表1 基于分数层的整合实验结果

结果	人脸识别(%)	说话人识别(%)	整合识别(%)
正确率	82.6	75.9	92.2

5 结束语

本文提出的方法和实验结果是我们已经完成的工作的一部分。在系统和算法实现过程中，我们并未将主要精力放在人脸识别和说话人识别算法方面，而是将注意力放在融合算法的效果上。目前，人脸识别率和说话人识别率的现状正好能够说明多通道生物特征认证所具有的优势。如果将来在实际应用中采用更好的人脸识别和说话人识别算法，则可以相信多通道生物特征认证将具有很大的潜力。

参考文献：

[1] Chibelushi C C, Deravi F, Mason J S. Voice and Facial Image Integration for Speaker Recognition[A]. Proc of IEEE Int'l Symp and Multimedia Technologies and Future Applications [C]. 1993.

[2] Dieckmann U, Plankensteiner P, Wagner T. Sesam: A Biometric Person Identification System Using Sensor Fusion[J]. Pattern Recognition Letters, 1997, 18(9): 827-833.

[3] Brunelli R, Falavigna D. Person Identification Using Multiple Cues[J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 1995, 10(7): 955-965.

[4] Demirbas K. Maximum a Posteriori Approach to Object Recognition with Distributed Sensors[J]. IEEE Trans on AES, 1988, 24(3): 309-313.

[5] Brunelli R, Falavigna D. Person Identification Using Multiple Cues[J]. IEEE Trans on PAMI, 1995, 12(10): 955-966.

[6] Turk M, Pentland A. Face Recognition Using Eigenfaces[A]. Proc of IEEE Conf on CVPR[C]. 1991. 586-591.

[7] Abdeljaoued Y. Fusion of Person Authentication Probabilities by Bayesian Statistics[A]. Proc of the 2nd Int'l Conf on Audio-and Video-Based Biometrics Person Authentication[C]. 1999. 172-175.

[8] Ben-Yacoub S, Abdeljaoued Y, Mayoraz E. Fusion of Face and Speech Data for Person Identity Verification[J]. IEEE Trans on Neural Networks, 1999, 10(5): 1065-1074.

(上接第 42 页)

缩品质大于 40 时，仍可从经过压缩处理的图像中提取完整的水印；当加入的高斯噪声方差小于 0.002 时，SVD 算法检测的水印清晰可见，抵抗高斯噪声的能力好于 DCT 算法，在一定程度上满足了对检测正确性的要求。

为了测试两种算法在受到裁剪攻击时的性能，将采用两种算法的嵌入水印的图像都分别裁剪掉右上角的 4×4、8×8、128×128 一块，然后进行水印检测。如图 3，水印的检测结果表明，SVD 算法比 DCT 算法的性能好。

5 结束语

本文提出了一种基于奇异值分解的数字水印改进算法。实验结果表明，改进后的算法对各种常规处理和攻击具有良好的性能，比 DCT 算法具有更好的隐藏性、安全性和鲁棒性。

此外，在水印中应用纠错编码以及水印置乱处理，会进一步提高水印检测的正确性和抗攻击的能力。

参考文献：

[1] Cox I J, Miller M L, Bloom J A. 数字水印[M]. 王颖，黄志蓓，等译. 北京：电子工业出版社，2003. 19-22.

[2] 刘瑞祯，谭铁牛. 基于奇异值分解的数字图象水印方法[J]. 电子学报，2001，29(2)：168-171.

[3] 金融. 基于正交模板的空间域数字水印算法：[硕士学位论文][D]. 上海：同济大学，2002.

[4] 王炳锡，陈琦，邓峰森. 数字水印技术[M]. 西安：西安电子科技大学出版社，2003. 82-83.

[5] Song Y T, Tan T N. Comparison and Four Different Digital Watermarking Technique[A]. Proc of 5th Int'l Conf on Signal Processing. Vol 2[C]. 2000. 944-947.

(上接第 71 页)

试结果我们可以看出，经过模拟退火算法的优化，使得关键路径经过的路由节点的个数大大减少，从而提高了这些测试程序在可重构阵列上的执行效率。

表1 测试结果

测试程序	算法	关键路径经过的路由节点的个数		
		手工布局	初始布局	模拟退火算法优化的布局
Median	中值滤波算法	12	16	10
Quant	量化算法	9	11	8
Swith_d	距离变换算法	11	16	8
Comput	计算匹配算法	7	9	5

5 结束语

粗粒度可重构阵列上的映射工具采用了模拟退火算法来进行布局和 Lee 算法来进行布线，使数据流图在粗粒度可重构阵列上的映射实现了自动化，并且映射的结果在可重构阵列上运行时也能达到比较高的效率。同时，这验证了我们在粗粒度可重构阵列上的布局算法的正确性和优化性能。

参考文献：

[1] Acock S J B, Dimond K R. Automatic Mapping of Algorithms onto Multiple FPGA SRAM Modules[A]. Proc of the 7th Int'l Workshop on Field-Programmable Logic and Applications[C]. 1997. 255-264.

[2] Lee J, Choi K, Dutt N. An Algorithm for Mapping Loops onto Coarse-Grained Reconfigurable Architectures[J]. ACM SIGPLAN Notices, 2003, 38(7): 183-188.

[3] 徐宁，洪先龙，蔡杜勤. BBL 布局算法研究[J]. 算机辅助设计与图形学学报，2004，16(9)，1216-1219.

[4] 李卓远，吴为民，洪先龙. 优化线长和拥挤度的增量式布局算法[J]. 计算机辅助设计与图形学学报，2003，15(6)：651-665.

粗粒度可重构阵列上的布局布线算法

作者： 左艳辉， 窦勇， 徐进辉， ZUO Yan-hui， DOU Yong， XU Jin-hui
作者单位： 国防科技大学计算机学院,湖南,长沙,410073
刊名： 计算机工程与科学 ISTIC PKU
英文刊名： COMPUTER ENGINEERING AND SCIENCE
年，卷(期)： 2007,29(11)
被引用次数： 2次

参考文献(4条)

1. Acock S J B;Dimond K R Automatic Mapping of Algorithms onto Multiple FPGA SRAM Modules 1997

2. Jong-eun Lee;Kiyoung Choi;Nikil D. Dutt An Algorithm For Mapping Loops Onto Coarse-Grained Reconfigurable Architectures[外文期刊] 2003(7)

3. 徐宁,洪先龙,董社勤 BBL布局算法研究[期刊论文]-计算机辅助设计与图形学学报 2004(9)

4. 李卓远,吴为民,洪先龙 优化线长和拥挤度的增量式布局算法[期刊论文]-计算机辅助设计与图形学学报 2003(6)

本文读者也读过(10条)

1. 徐佳庆.邬贵明.窦勇.XU Jia-qing.WU Gui-ming.DOU Yong 二维DCT在粗粒度可重构处理器上的实现[期刊论文]-计算机工程2008,34(20)

2. 张文良.雍珊珊.戴鹏.王新安.谢峥.ZHANG Wenliang.YONG Shanshan.DAI Peng.WANG Xin'an.XIE Zheng 小数运动估计在可重构处理器ReMAP的映射实现[期刊论文]-微电子学2011,41(1)

3. 杨超.谢憬.毛志刚.YANG Chao.XIE Jing.MAO Zhi-gang 一种循环流水阵列架构[期刊论文]-信息技术2010(2)

4. 贺献辉 使用UML2.0的可重构多媒体硬件加速器设计与分析[学位论文]2007

5. 董亚卓.窦勇.刘明政.Dong Yazhuo.Dou Yong.Liu Mingzheng 面向滑动窗口应用的设计空间探索方法[期刊论文]-计算机辅助设计与图形学学报2008,20(5)

6. 方潜生.王煦法.何劲松 硬件进化的快速算法模型研究[期刊论文]-中国科学技术大学学报2003,33(5)

7. 张素兰.张宏烈.ZHANG Su-lan.ZHANG Hong-lie 可重构系统中基于禁忌搜索算法的软硬件划分[期刊论文]-齐齐哈尔大学学报（自然科学版）2009,25(1)

8. 董亚卓.左艳辉.刘明政.窦勇.DONG Ya-zhuo.ZUO Yan-hui.LIU Ming-zheng.BOU Yong 一种面向可重构硬件的编译中间表示方法[期刊论文]-计算机工程与科学2008,30(9)

9. 窦勇.董亚卓.徐进辉.邬贵明.DOU Yong.DONG Ya-Zhuo.XU Jin-Hui.WU Gui-Ming 基于参数化存储结构的滑动窗口IP核自动生成[期刊论文]-软件学报2009,20(2)

10. 董亚卓.刘明政.夏飞.窦勇.DONG Ya-Zhuo.LIU Ming-Zheng.XIA Fei.DOU Yong 滑动窗口应用循环展开及其数据通路生成[期刊论文]-计算机学报2008,31(6)

引证文献(2条)

1. 胡楠楠 演化芯片布局布线算法及其评测[学位论文]硕士 2011

2. 张博为 基于可重构计算的矩阵分解并行结构研究[学位论文]博士 2013