片上网络拓朴优化:在离散平面上布局与布线

马立伟,孙义和
(清华大学微电子学研究所,北京 100084)

摘 要: 微系统芯片(System-on-Chip,SoC)发展到今天,集成密度指数增长和芯片面积的急剧膨胀使得全局连线的延时上升,可靠性下降,成为集成电路的设计瓶颈.片上网络(Network-on-Chip,NoC)是解决整个芯片上数据有效传输的结构之一,以片上网络为基础通信架构的微系统芯片称为片上网上系统芯片(System-on-Network-on-Chip,SoNoC).微系统芯片内通信模式兼有随机性和确定性,应该根据特定应用的通信特征设计片上网络.本文在确定 SoNoC 设计流程的基础上,根据 SoNoC 的通信特征,选择了合适的离散平面结构,对 SoNoC 的运算及控制等模块进行布局、对模块间的通信依赖关系进行布线,发展出 FRoD(Floor-plan and Routing on Discrete Plane)算法,以自动生成片上网络的拓扑结构.该算法定义了离散平面的一般表示方法,并在四种典型的离散平面上使用不同规模的随机系统完成了系列实验.为了处理系统和网络之间的耦合关系,逐点分裂的布局算法可以逐步学习和适应系统的通信需求,同时优化系统的执行时间和通信能量,在运行随机任务流图的模拟系统上与随机布局结果相比可以节省 30 %左右的通信能量,20 %左右的系统通信时间.串行、并行和串并混合的布线算法使用最短路径把通信关系分布在离散平面的通道上,使不同的通信关系尽量复用网络通道,与全连接网络相比可以节省 10 %到 30 %的面积代价.

关键词: 微系统芯片; 片上网络; 片上网上系统芯片; 片上网络综合
中图分类号: TN432 **文献标识码**: A **文章编号**: 0372-2112 (2007) 05-0906-06

Network-on-Chip Topology Optimizations: Floor-plan and Routing on Discrete Plane

MA Li-wei, SUN Yi-he
(*Institute of Microelectronics, Tsinghua University, Beijing, 100084, China*)

Abstract: As feature size shrinking, integration increasing and chip area expanding, delay and reliability of global wires become the bottleneck of IC design. Network-on-Chips (NoCs) have been proposed as one of the solutions of System-on-Chips (SoCs) communication infrastructure and enable a new SoC paradigm—System-on-Network-on-Chip (SoNoC). Traffic patterns in SoNoC are both stochastic and deterministic; so on-chip networks must be application-specific. Based on SoNoC design flow, a group of algorithms—FRoD (Floor-plan and Routing on Discrete plane) has been proposed. General definition of discrete plane and four discretization methods have been discussed, and the FRoD algorithms have been verified on the four planes. In order to decouple the system requirements and network architectures, the splitting growth floor-plan algorithm has been developed and can save about 30 % execution time and 20 % transmission energy, compared with random floor-plan results. Sequential, parallel and mixed routing algorithms have been developed to link all the traffic with shortest paths and can save 10 %～30 % area cost, compared with full-linked networks.

Key words: network-on-chip; system-on-chip; system-on-network-on-chip; NoC synthesis

1 引言

集成电路正在进入微系统芯片(SoC)时代,集成密度指数增长,芯片面积急剧膨胀,芯片工作频率不断提高,使得全局信号在高速芯片传输需要多拍工作时钟[1],而且受工艺和串扰的影响,全局连线的可靠性成为集成电路的设计瓶颈[2].片上网络是在这种情况下发展起来的解决微系统芯片众多处理核之间数据高速传输的一种优选的基础通信架构.片上网络必须服务于微系统芯片的通信需求.微系统芯片的功能、它的使用和设计条件的约束以及芯片的划分结构决定着片上网络的形态和设计方法;而片上网络的通信性能也直接影响

收稿日期:2006-08-01 修回日期 2007-01-18
基金项目:国家自然科学基金(No. 60236020);高等学校博士学科点专项科研基金(No. 20050003083)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

微系统芯片的性能与代价. 所以,面向特定应用的片上网络设计方法学是研究 SoNoC 设计的重要目标之一.

面向特定应用设计片上网络,首先要求为不同的应用系统确立统一的模型及其实现方法,能够分析和提取不同应用的通信特征,然后要根据通信协议定义适当的拓扑结构,并能快速自动生成所要求的片上网络的源代码结构描述[3];这里最重要的是,能够根据系统的通信需求设计出最优化的网络拓扑结构 ——系统模块合理的分布在网络节点上并用标准长度的网络通道相连接,使得通信能量最低,系统执行时间最短,网络面积代价最小. 芯片内的系统和网络需要协同设计以保证复杂芯片的成本、性能和面市时间满足需求[4].

芯片内系统和网络之间具有很强的耦合关系,系统的通信需求决定着网络的实现,网络实现也影响系统模块之间的通信延时、消息数量和信息总量. 因此,系统的通信需求在设计网络拓扑时是动态的,这就需要使用演化的方法逐步优化网络的拓扑结构. 现有的片上网络综合流程中的拓扑优化和生成方法使用固定的通信模型,很少考虑系统和网络之间的耦合关系,也较少考虑网络的物理可布线性[5~7]. 本文采用离散平面的方法,保证片上网络的平坦性和网络频率的统一,使用网络演化的方法在网络设计的同时考虑系统与网络耦合的影响.

2　网络演化的动因和拓扑优化的定义

2.1　网络演化的动因和随机系统建模

信息在网络上传输需要资源、时间和能量. 信息经过通道在相临的两个节点间传输称为网络中的一跳. 网络中两点间最短路径的通道数量称为两点间的拓扑距离. 本文假设单位信息的一跳需要使用的时间、能量和资源是固定的. 要求网络收缩的力量包括:

(1) 通信量大的模块之间的拓扑距离要尽量小以节省能量.

(2) 处于系统执行关键路径上的模块之间通信关系的拓扑距离要尽量小以缩短系统的运行时间.

(3) 路由关系相似的模块之间可以复用通信资源,以减少网络中节点和边的数量,从而优化片上网络使用的资源.

要求网络扩张的力量包括:

(1) 一个网络节点上挂载的模块数量是有限的,模块不能堆叠.

(2) 系统模块之间太长的连线应该分段,以满足网络频率的要求,增加不同通信之间复用通道的机会.

(3) 片上网络的结构必须是平坦的,过于复杂的连线关系不利于二维芯片实现,这也必须延展网络的拓扑结构.

促使网络演化的这些动因大部分不能简单解析,而是随着网络结构的改变而动态变化的.

TGFF 是在嵌入式系统研究中广泛使用的一种随机任务流程图生成器[8]. TGFF 生成的任务流图中每个任务有一个随机的执行时间,每组通信关系有一个随机的通信量,同时为了不至于过胖或者过瘦,任务的入度和出度也做了限定. 本文实现了可以运行随机任务流图的虚拟系统,任务被均匀随机地映射到虚拟模块上,每个模块上执行若干任务. 接收到足够的消息后模块根据内部的消息映射表触发特定消息的运行,当消息执行完毕时它再根据映射表向其它模块发射新的消息. 所有消息的发射、传输、到达和运行都被系统监视并记录下来. 当任务流图中所有的任务都被执行完毕时,视为完成一个系统行为观察周期,开始根据记录的结果对网络结构进行优化. 本文建立了任务数分别为 12、24、48、96、192、384 的任务流图,并映射到 4、8、16、32、64、128 个模块组成的随机系统上.

对于系统总的执行时间来说,性能的瓶颈与消息执行和通信都有关系. 网络设计并不需要优化所有的通信延时,而只需要优化那些处在系统执行瓶颈上的消息通信. 在系统执行时,那些被等待的消息的通信延时对系统执行时间贡献最大. 在一个观察时间段内,两个模块间被等待的消息数目称为该通信关系的延时敏感度. TGFF 模拟器可以观察到网络结构变化对通信关系的延时敏感度的影响. 复杂系统模块间的能量消耗分布也会随着网络结构的变化而变化,本文中使用的 TGFF 模拟器没有实现这一点,但这并不影响布局和布线算法的有效性.

2.2　连续平面离散化

在基于标准单元综合的芯片设计流程中,由于标准单元的面积相对较小,连线关系允许差别很大,硅表面可以被看成是一个连续的平面;但在片上网络的设计中使用结构化的离散平面却更有利:它简化了网络协议和拓扑设计;标准化的网络通道可以保证统一的网络频率;事先确定的平面结构可以保证片上网络是平坦的、可布局的和可布线的.

二维平面可以有很多种离散化的方法,任何一种离散方法只要给出如下定义和计算公式,都可以应用于第 3、4 节介绍的布局和布线算法:

(1) 节点 z 的表示方法,以及加减法则和数乘法则的定义.

(2) 原点的邻接点集合 Pt_{adj},通过这个集合使用加法法则可以列举任意点的邻接点. 满足邻接点关系的两个点 $z_1 - z_2 \in Pt_{adj}$ 之间连接有通道,记为 $e = (z_1, z_2)$.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

(3) 节点和布线的密度,可以指导不同通信需求的系统根据需要选择离散平面.

(4) 点 z 到原点 z_o 的几何长度 $D(z)$ 的计算公式. 则两个点 z_1 和 z_2 间的几何距离为 $D(z_1, z_2) = D(z_1 - z_2)$.

(5) 点 z 到原点 z_o 的最短路径的一般形式,并定义它的长度为点的拓扑长度 $d(z)$. 则两个点 z_1 和 z_2 间的拓扑距离为 $d(z_1, z_2) = d(z_1 - z_2)$.

(6) 点 z 是否在一组点 z_S 和 z_T 的某条最短路径上的判定条件,记为 $z \in (z_S, z_T)$. 它等价于 $d(z_S, z) + d(z, z_T) = d(z_S, z_T)$.

(7) 通道 $e = (z_1, z_2)$ 是否在一组点 z_S 和 z_T 的某条最短路径上的判定条件,记为 $(z_1, z_2) \in (z_S, z_T)$. 它等价于 $d(z_S, z_1) + d(z_2, z_T) + 1 = d(z_S, z_T)$.

表 1 给出了 4 邻接网格和 8 邻接网格的表示方法. 表 2 给出了 6 邻接网格和 3 邻接网格的表示方法.

表 1 4 邻接网格和 8 邻接网格的定义与相关公式

	4 邻接网格	8 邻接网格
表示方法 z	(a, b), a, b 为整数	
加减法则 $z_1 \pm z_2$	$(a_1 \pm a_2, b_1 \pm b_2)$	
数乘法则 $p \cdot z$	$(p \cdot a, p \cdot b)$	
邻接点集合 Pt_{adj}	$\{(0, \pm 1), (\pm 1, 0)\}$	$\{(0, \pm 1), (\pm 1, 0), (\pm 1, \pm 1)\}$
节点密度	1	
通道密度	2	4
几何长度 $D(z)$	$\sqrt{a^2 + b^2}$	
拓扑长度 $d(z)$	$\lvert a\rvert + \lvert b\rvert$	$\max(\lvert a\rvert, \lvert b\rvert)$

表 2 6 邻接网格和 3 邻接网格的定义与相关公式

	4 邻接网格	8 邻接网格
表示方法 z	(a, b, c), $a + b\omega + c\omega^2$, a, b, c 为整数, $\omega = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$	
加减法则 $z_1 \pm z_2$	$(a_1 \pm a_2, b_1 \pm b_2, c_1 \pm c_2)$	
数乘法则 $p \cdot z$	$(p \cdot a, p \cdot b, p \cdot c)$	
邻接点集合 Pt_{adj}	$\{(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)\}$	- *
节点密度	$\frac{4}{\sqrt{3}}$	$\frac{2}{3\sqrt{3}}$
通道密度	$2\sqrt{3}$	$\frac{2}{\sqrt{3}}$
几何长度 $D(z)$	$\sqrt{a^2 + b^2 + c^2 - ab - bc - ca}$	
拓扑长度 $d(z)$	$\max(\lvert a\rvert, \lvert b\rvert, \lvert c\rvert)$	- *

2.3 拓扑优化的形式化定义

给定一组通信关系,记为 $G_C(M, C)$,其中 $m_i \in M$ 表示系统中的一个模块,有向边 $c_{i,j} = (m_i, m_j) \in C$ 表示 m_i 和 m_j 之间的通信关系;给定离散平面 $G_P(V_P, E_P)$,其中 $v_i \in V_P$ 表示平面中的一个点,$e_{i,j} = (v_i, v_j) \in E_P$,$(v_i - v_j) \in Pt_{adj}$ 表示平面中的一组通道;求一个网络拓扑结构 $G(V, E)$ 和一组映射 $Map(m): M \to V$,其中 $V \in V_P$, $E \in E_P$, $v = Map(m)$ 表示模块 m 被布局在 v 节点上,使得:

(1) 系统的通信总能量最小:

$$\mathrm{Min} \sum_{\forall c \in C} I(c) \cdot d(c) \tag{1}$$

其中 $I(c)$ 表示通信关系 c 的总的通信量,$d(c_{i,j}) = d(Map(m_i), Map(m_j))$ 表示通信关系 $c_{i,j} = (m_i, m_j)$ 的通信距离.

(2) 系统的执行时间最小:

$$\mathrm{Min} \sum_{\forall c \in C} S(c) \cdot d(c) \tag{2}$$

其中 $S(c)$ 表示系统运行时间对通信关系 c 的时延敏感度.

(3) 每个节点上挂载的模块总面积满足约束:

$$\sum_{\forall m_i, Map(m_i) = v} a(m_i) < A(v) \tag{3}$$

其中 $a(m)$ 表示模块 m 的面积,$A(v)$ 表示节点 v 的上可以挂载的模块的总面积.

(4) 使用 E 中的边,任何两个模块都可以使用最短路径通信,并且网络本身的面积代价最小:

$$\mathrm{Min}(\lvert E\rvert) \tag{4}$$

3 基于组迁移的逐点分裂的布局算法

该算法假设任何两个模块间的通信都可以使用最短路径实现连接,用来优化目标:式(1)、式(2)、式(3). 逐点分裂算法的思想是从一个单节点的网络开始,系统每进行一个周期的模拟,就在网络中找一个合适的分裂点,在网络内或网络外确定目标点,并在两个节点上重新分配模块,完成节点的生长和扩张. 然后把新的网络结构下的通信参数反标到系统中,重新运行整个系统,并进入下一个演化周期.

3.1 分裂点、目标点和分配比例

假设离散化平面是无限的,整个网络从原点开始. 平面上的节点可以分成三类:黑点、灰点和白点. 黑点是整个网络扩张还未触及的节点;灰点是已经包括在网络结构中,但挂载的模块数目太多,还不能满足面积约束的点;白点是包括在网络结构中,面积约束已经满足的网络节点. 本研究中选取的待分裂点是挂载模块总面积最大的灰点.

目标点的选择关系到算法生成的拓扑结构的最终形状. 首先计算整个系统的重心,目标点的优选方向是距离重心最近的黑点,其次是距离重心最远的灰点,最后是距离重心最远的白点. 即可能的分裂模式是,灰点

* 3 邻接网格中的节点需要分作两种情况讨论,一些量的计算公式需要考虑多种情况,已用程序实现,这里没有详述,但不影响相关的算法设计和结论.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

→黑点、灰点→灰点和灰点→白点.

在分裂点和目标点上重新分配挂载的模块,需要保证一定的比例.如果两个点都有可供继续生长的其它黑点,则根据它们可能的生长方向的个数之比进行分配;如果分裂点没有可供生长的黑点作为进一步的可能的分裂方向,那么比例选择要让分裂点在分裂之后满足面积约束,迫使它变为白点.当在两个节点之间分配好模块之后,只要目标点的面积约束被违反,不论它原来是什么颜色,都把它重新定为灰点.

3.2　组迁移划分算法

本文选取了成熟的组迁移算法作为模块划分的基本算法,可以在 $O(n)$ 的时间完成规模为 n 的问题的划分[9].因为该算法比较成熟,这里重点介绍划分的代价函数.用 L、R 和 T 分别表示分裂点、目标点和系统中其它节点上挂载的所有模块的集合.单次划分的目标是重新划分 L 和 R,使得 L, R 中模块的数目满足一定的比例关系,并且增加通信代价最小.

3.2.1　代价函数

两个模块 m_i, m_j 分布在离散平面的两个节点 z_l, z_r 上,会形成能量和延时上的代价.其能量代价为:

$$Cost_e = (I(c_{i,j}) + I(c_{j,i})) \cdot d(z_l, z_r) \tag{5}$$

其延时代价为:

$$Cost_t = (S(c_{i,j}) + S(c_{j,i})) \cdot d(z_l, z_r) \tag{6}$$

总代价是这两种代价的求和.

当一个模块 m 从 L 迁移到 R,会引起整个系统的代价发生变化,增加的系统代价为:

$$\begin{aligned}\Delta Cost(m, L, z_L, R, z_R) = &\sum_{\forall m_i \in L, m_i \neq m} Cost(m, z_R, m_i, z_L) - \sum_{\forall m_i \in R} Cost(m, z_L, m_i, z_R) \\ &+ \left(\sum_{\forall m_i \in T} Cost(m, Z_R, m_i, z_{m_i}) - \sum_{\forall m_i \in T} Cost(m, Z_L, m_i, z_{m_i}) \right)\end{aligned} \tag{7}$$

其中,z_L, z_R 分别是分裂点和目标点,z_{m_i} 是模块 m_i 所在的网络节点.式(7)说明了模块迁移引起系统代价的增量包括三个部分:把 m 从 z_L 迁移到 z_R 时与 L 中其它模块之间增加的代价,与 R 中所有模块之间减少的代价,和与 T 中所有模块之间改变的代价.

当在 L 和 R 之间交换两个模块 l 和 r 时,整个系统的代价也会发生变化,增加的系统代价为:

$$\begin{aligned}\Delta Cost(l, L, z_L, r, R, z_R) = &\Delta Cost(l, L, z_L, R, z_R) \\ &+ \Delta Cost(r, L, R, z_R, L, z_L) \\ &+ 2\,Cost(l, z_L, r, z_R)\end{aligned} \tag{8}$$

3.2.2　按比例分配和组迁移优化

在 L, R 两个集合里按比例分配模块是通过在集合之间的模块迁移操作完成的.首先合并集合 L, R 到 L,然后在 L 中寻找迁移代价最小的模块 m 迁移到 R 中,直到 L 和 R 的模块数目达到第 3.1 小节中规定的比例要求.

组迁移算法是在 L 和 R 之间寻找交换代价最小的两个模块 l 和 r,然后实施这个交换,并把 l 和 r 锁住,直到有 L 和 R 中有一个集合被完全锁死而结束循环.如果把所有交换操作的结果依次排列成一个表,那么所有交换操作的代价结果是一个 U 型的曲线,在所有结果中找到代价最小的那一种划分方式,作为组迁移算法的最终结果.

3.3　算法的性能

为了验证算法的性能,逐点分裂的布局结果与随机布局结果在能量和延时两个方面做了比较.系统通信总能量 E_s(逐点分裂布局)和 E_r(随机布局)之比 $r_e = \frac{E_s}{E_r}$ 可以评价能量优化程度.令单点网络下系统运行时间为 T_o,在逐点分裂布局下为 T_s,在随机布局下为 T_r,运行时间增加量之比 $r_t = \frac{T_s - T_o}{T_r - T_o}$ 可以表示运行时间的优化程度.表 3 给出了不同规模的随机系统在四种离散平面下的 r_e 和 r_t,可以看出逐点分裂布局算法比随机布局节省 30 %左右的通信能量,节省 20 %左右的系统通信时间.

表 3　逐点分裂算法与随机布局结果的 r_e 与 r_t 数据

平面分类邻接数量		系统规模(模块数)					
		4	8	16	32	64	128
3	r_e(%)	100.00	96.77	88.71	78.41	75.34	68.71
	r_t(%)	125.00	95.45	88.71	90.98	75.77	81.56
4	r_e(%)	88.48	87.86	71.80	85.41	77.44	68.52
	r_t(%)	90.91	108.70	76.92	81.73	66.05	79.02
6	r_e(%)	72.25	87.70	78.85	81.68	68.95	71.16
	r_t(%)	81.82	83.33	76.60	77.23	53.76	70.70
8	r_e(%)	86.36	92.53	77.77	79.22	75.49	72.49
	r_t(%)	53.85	94.12	102.86	84.62	54.96	51.06

4　串并混合布线算法

布线算法使用最短路径连接所有的通信关系,并优化目标式(4),使网络的总代价最小.串行布线是按照通信关系依次布线;并行布线是统一考虑所有的通信关系,首先使用复用率最高的通道.与传统的 ASIC 布线不同,网络布线的目的是尽量复用通道,而不是相互避开[9].

算法假设所有的通信关系都可以在网络中用最短路径相连接,如果布局的结果出现凹形的情况,需要在网络加入新的节点和通道,以保证该假设成立.可以在布线的过程中随时检测并确保这个假设,但因不是算法的重点,述略.

4.1　串行流量布线算法

算法首先初始化所有的布线通道状态,然后对通

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

信关系依次布线,这需要两个布线步骤,广度优先搜索和回溯. 对于通信关系 $c=(v_S,v_T)$,广度优先搜索算法从 v_S 到 v_T 进行搜索,并标记了拓扑距离和资源距离,其中资源距离是从 v_S 到 v_T 需要增加通道的最小数量;回溯算法则找到一条增加通道数目最少的最短路径,并把新增加的通道资源加入到网络结构中去.

4.2 并行流量布线算法

用 $P(e)$ 表示最短路径可能经过通道 e 的通信关系的数量,ξ 表示使算法终止的 $P(e)$ 的最小阈值. 算法首先计算当前通信关系集合 C 下 $P(e)$ 最大的通道 $e^*=(v_s^*,v_t^*)$,并且满足 $P(e)>\xi$;然后把所有可能经过 e^* 的通信关系 $c=(v_S,v_T)$ 分裂成两个小的通信关系 (v_S,v_s^*) 和 (v_t^*,v_T). 如果新通信关系的两个点重合,那么就不必将其加入集合 C 中. 这样随着算法的进行,C 中的通信关系将被全部分裂掉,从而结束算法. 在并行布线算法中取 $\xi=1$.

4.3 先并行后串行的混合流量布线算法

在串行算法中通信关系的选取顺序会影响流量分布的效果;而并行算法分裂已有的通信关系,在算法的后期会扭曲真实的通信分布. 因此可以把这两种算法结合起来,先用并行的布线算法选择出复用率比较高的通道,然后在这个基础上使用串行布线方法,削减通信关系的选取顺序对布线效果的影响. 在实际使用中 ξ 的选取很重要,$\xi=1$ 时,算法退化为并行算法;$\xi=\infty$ 时,算法退化为串行算法;只有它的值适当,才能在前面两种算法的基础上获得更好的效果.

4.4 三种流量布线算法在四种离散平面的实验结果

网络主要由交换器和通道组成,而具有固定缓冲器的网络交换器的面积正比于它的输入输出端口的数量[4]. 网络通道正好对应一组输入输出端口,所以整个网络的面积可以近似认为正比于网络通道的数量. 定义通道布线使用率为 $\rho=\frac{E}{E_P}$,其中 $|E|$ 表示实际网络中使用的通道数量,E_P 表示离散平面中可以使用的通道数量. ρ 的值越低,说明算法在全连接网络的基础上资源节省的效率越高.

表 4 给出不同模块数目、不同平面下布线的结果 ρ,从这个表中可以看出:

(1) 对于模块数多的复杂系统,算法的效果比较好. 这是因为复杂的系统连线比较多,可以优化的余地比较大.

(2) 对于联结度高的离散平面,算法的效果比较好. 这是因为这样的平面提供了更多的连线资源可以选择,优化空间比较大.

图 1 给出了混合算法在 4 邻接平面上对模块数为 64 的随机系统的布线使用率 ρ 与 ξ 的关系,可以看出 ρ 随 ξ 的增长呈 U 型变化,混合算法可以在串行和并行算法的之间获得进一步的优化.

表 4 三种流量布线算法的通道布线使用率 ρ. 其中混合算法中取 $\xi=20$

平面分类邻接数量		系统规模(模块数)					
		4	8	16	32	64	128
3	串行 ρ(%)	100.0	100.0	100.0	100.0	98.81	97.55
	并行 ρ(%)	100.0	100.0	97.37	100.0	96.43	96.47
	混合 ρ(%)	100.0	100.0	97.37	100.0	96.43	96.47
4	串行 ρ(%)	100.0	94.44	97.83	92.00	92.59	95.80
	并行 ρ(%)	100.0	100.0	89.13	94.00	88.43	86.73
	混合 ρ(%)	100.0	100.0	89.13	93.00	87.50	87.83
6	串行 ρ(%)	100.0	80.77	83.33	85.42	86.71	89.61
	并行 ρ(%)	100.0	80.77	80.30	76.39	79.75	82.34
	混合 ρ(%)	100.0	80.77	80.30	76.39	79.11	82.34
8	串行 ρ(%)	90.0	82.35	66.25	65.43	69.23	67.27
	并行 ρ(%)	90.0	76.47	65.00	61.47	60.34	58.92
	混合 ρ(%)	90.0	76.47	65.00	61.47	60.34	59.14

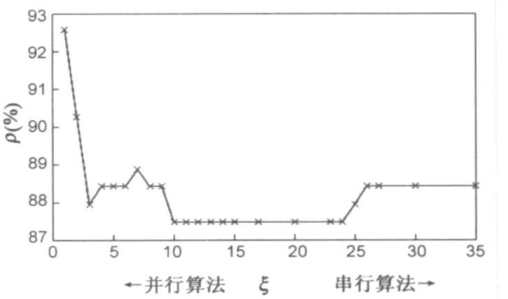

图 1 混合算法在 4 邻接平面上对模块数为 64 的随机系统的通道布线使用率 ρ 的结果

5 结论

网络拓扑的优化问题是片上网络设计自动化的重要组成部分,也是当前片上网络研究的重点之一. 但不同研究小组对片上网络拓扑构造问题定义的角度、条件和目标都不尽相同,使用的优化手段比较广泛,拓扑构造的结果也各有侧重,目前还未出现统一的评价标准,不同工作成果的简单比较是盲目的. 现有的网络拓扑构造算法多强调逻辑上的拓扑结构[10~13],较少关注拓扑在物理上的实现方式. MILP[14] 增强了传统的模块布局算法,在模块之间的空隙构造网络,但网络连线的距离不统一,不利于提高网络的工作频率.

在研究思路上,本文使用网络演化的算法,考虑了片上系统和网络的耦合关系,仿照传统 ASIC 的设计思路建立了离散平面模型,并在离散平面上进行模块布局和流量布线,建立了一组网络拓扑优化算法族. 本文通过把拓扑优化问题转化为离散平面上的系统模块布局和通信关系布线问题,降低了问题的难度. 经过对不同规模的随机系统进行验证,该设计方法比随机布局算法可以节省 30 %左右的能量,20 %左右的系统执行

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

时间,比全连接的网络结构节省 10 %到 30 %的网络代价.该设计流程可以更换其它的局部算法,形成一系列算法簇,与 SoNoC 的系统建模方法、片上网络的自动生成方法一起,可以构成完整的片上网络的设计方法学.

参考文献:

[1] Ho R, Mai K W, Horowitz M A. The future of wires[J]. Proceedings of the IEEE, 2001, 89(4): 490 - 504.

[2] Benini L, De Micheli G. Networks on chips: A new SoC paradigm[J]. Computer, 2002, 35(1): 70 - 78.

[3] Ma Liwei, Sun Yihe. On-chip networks design automationwith source routing switches [J]. Tsinghua Science and Technology, Accepted, 2007, 12(1): 77 - 85.

[4] Vestias M P, Neto H C. Co-synthesis of a configurable soc platform based on a network on chip architecture[A]. In Proceedings of the Asia and South Pacific Design Automation Conference[C]. Piscataway: IEEE, 2006. 48 - 53.

[5] Bertozzi D, Jalabert A, Srinivasan M, Tamhankar R, Stergiou S, Benini L, De Micheli G. Noc synthesis flow for customized domain specific multiprocessor systemson-chip [J]. IEEE Transactions on Parallel and Distributed Systems, 2005, 16(2): 113 - 129.

[6] Lahiri K, Raghunathan A, Dey S. Design space exploration for optimizing on-chip communication architectures [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2004, 23(6), 952 - 961.

[7] Srinivasan K, Chatha K S, Konjevod G. Linearprogramming-based techniques for synthesis of network-on-chip architectures [J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006, 14(4): 407 - 420.

[8] Dick R P, Rhodes D L, Wolf W. Tgff: task graphs for free[A]. In Proceedings of the Sixth International Workshop on Hardware/ Software Codesign[C]. Los Alamitos, CA, USA: IEEE, 1998, 97 - 101.

[9] Sherwani N A. Algorithms for VLSI Physical Design Automation[M]. Norwell, Massachusetts, USA: Kluwer Academic Publishers, 1993.

[10] Hu J, Marculescu R. Energy-and performance-aware mapping for regular NoC architectures[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2005, 24(4): 551 - 562.

[11] Lahiri K, Raghunathan A, Dey S. Design space exploration for optimizing on-chip communication architectures [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, , 2004, 23(6): 952 - 961.

[12] Srinivasan K, Chatha K S. ISIS: A Genetic Algorithm Based Technique for Custom On-Chip Interconnection Network Synthesis[A]. In Proceedings of the 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design (VLSID '05)[C]. Washington, DC, USA: IEEE Computer Society, 2005. 623 - 628.

[13] Murali S, Micheli G D. SUNMAP: a tool for automatic topology selection and generation for NoCs[A]. In Proceedings of the 41st annual conference on Design automation[C]. New York, NY, USA: ACM Press, 2004. 914 - 919.

[14] Srinivasan K, Chatha K, Konjevod G. Linear-programming-based techniques for synthesis of network-on-chip architectures[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006, 14(4): 407 - 420.

作者简介:

马立伟　男,1979 年生于河北,2001 年毕业于上海交通大学信息与控制工程系,获学士学位,现为清华大学微电子学与固体电子学研究所博士研究生.主要研究方向为片上网络设计方法学. E-mail: mlw01@mails.tsinghua.edu.cn

孙义和　男,1945 年生于安徽,清华大学微电子学与固体电子学研究所教授、博士生导师,中国电子学会高级会员.主要研究方向:LSI/ VLSI 测试方法学和可测性设计,多媒体 VLSI 设计技术,片上网络设计,网络和数据安全 VLSI 结构. E-mail: sunyh@tsinghua.edu.cn

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net