用于时分复用技术的多阶段协同优化FPGA布线方法

刘耿耿*①② 许文霖① 周茹平① 徐 宁③
①(福州大学计算机与大数据学院 福州 350116)
②(福建省网络计算与智能信息处理重点实验室 福州 350116)
③(武汉理工大学信息工程学院 武汉 430070)

摘 要：时分复用(Time-Division Multiplexing, TDM)技术被广泛地运用于解决IO瓶颈问题，以提高现场可编程门阵列(Field Programmable Gate Array, FPGA)系统的可布线性，但TDM比率的增大会导致系统时延的显著增加。因此，为了优化FPGA系统时延以及可布线性，该文提出一种用于时分复用技术的多阶段协同优化FPGA布线(Multi-Stage Co-Optimization FPGA Routing, MSCOFRouting)方法。首先，设计自适应布线算法，以减少布线拥塞情况，提高可布线性，解决FPGA间的布线优化问题，为后续的TDM比率分配提供高质量的布线结果。其次，为了避免因大规模线网组的TDM比率过大而导致系统时延劣化的情况，提出基于拉格朗日松弛(Lagrangian Relaxation, LR)的TDM比率分配算法，为布线图的边分配系统时延更小的初始TDM比率。此外，为了进一步减小最大线网组的TDM比率，通过一种多层次的TDM比率优化算法，缩减线网组和FPGA连接对的TDM比率。同时，为了提高MSCOFRouter的运行效率，在上述3个算法中使用多线程并行化方法，有效缩减运行时间。实验结果表明，MSCOFRouting可以获得满足TDM比率约束的结果，取得同类工作中最佳的布线优化结果和TDM比率分配结果。

关键词：FPGA系统；逻辑验证；时分复用；布线；拉格朗日松弛
中图分类号：TN43 **文献标识码**：A **文章编号**：1009-5896(2023)09-3430-09
DOI: 10.11999/JEIT221158

Multi-Stage Co-Optimization FPGA Routing for Time-Division Multiplexing Technique

LIU Genggeng①② XU Wenlin① ZHOU Ruping① XU Ning③
①(*College of Computer and Data Science*, *Fuzhou University*, *Fuzhou* 350116, *China*)
②(*Fujian Key Laboratory of Network Computing and Intelligent Information Processing* (*Fuzhou University*), *Fuzhou* 350116, *China*)
③(*School of Information Engineering*, *Wuhan University of Technology*, *Wuhan* 430070, *China*)

Abstract: Time-Division Multiplexing (TDM) technology is widely applied to solving the IO limitation problem to improve the routability of FPGA system. However, the increase of the TDM ratio leads to a significant increase in system delay. Therefore, a Multi-Stage Co-Optimization FPGA routing (MSCOFRouting) for Time-Division Multiplexing is proposed in this paper to optimize the system delay and the routability of FPGA system. First, an adaptive routing algorithm is proposed to reduce routing congestion, improve the routability, solve the routing optimization problem between FPGAs, and provide high-quality routing results for subsequent TDM ratio assignment. Second, to avoid the delay degradation caused by excessive TDM ratio of large-scale net groups, a TDM ratio assignment algorithm based on Lagrangian relaxation is utilized to assign the initial TDM ratio with a smaller delay to the edge distribution system of the routing graph. In addition, a multi-level TDM ratio optimization algorithm is used to reduce the TDM ratios of the net group with maximum TDM ratios. The TDM ratio reduction is employed for the net group and the FPGA connection pair. Meanwhile, a

收稿日期：2022-09-06；改回日期：2023-03-31；网络出版：2023-04-04
*通信作者： 刘耿耿 liugenggeng@fzu.edu.cn
基金项目：国家自然科学基金(61877010)
Foundation Item: The National Natural Science Foundation of China (61877010)

multi-thread parallelization method is integrated into the three algorithms above to improve further the efficiency of MSCOFRouter. Experiments show that MSCOFRouting can obtain the results satisfying the TDM ratio constraint, and achieve the best routing optimization results and TDM ratio assignment results.

Key words: FPGA systems; Logic verification; Time-division multiplexing; Routing; Lagrangian relaxation

1　引言

近年来，现场可编程门阵列(Field Programmable Gate Array, FPGA)广泛应用于各个领域，如深度学习[1]、云计算[2]和芯片设计的验证问题[3]。复杂的专用集成电路(Application Specific Integrated Circuit, ASIC)芯片设计需要通过多种验证方法保证设计的正确性。验证的主要方法有软件仿真、形式化验证和硬件仿真。随着芯片设计越来越复杂，软件仿真需要花费大量的运行时间仿真每个逻辑电路，形式化验证难以适用于大型ASIC设计的验证问题，同时传统的硬件仿真方法需要花费大量的成本来实现验证。而基于FPGA原型验证的硬件仿真方法可以解决大型ASIC设计的验证问题，并且能够在成本和运行时间之间做到平衡。因此，许多先进的微处理器制造商在其验证过程中使用了基于FPGA原型验证的硬件仿真方法。

ASIC设计随着规模的不断扩大，越来越难在单个FPGA上实现逻辑验证[4,5]。因此，大型的ASIC设计将根据设计需求被划分到多个FPGA内。与单FPGA系统相比，多FPGA系统逻辑复杂性更高，设计能力更强。高速收发器被运用于FPGA系统中，可以提高FPGA间的数据交换速度。随着FPGA内部构造越发复杂，FPGA间信号数量会远远超过I/O引脚数量。而时分复用(Time-Division Multiplexing, TDM)技术被广泛运用于解决I/O引脚数量不足的问题。然而，TDM技术会造成FPGA间的信号延迟，导致系统时延的增加。因此，减少TDM技术所导致的系统时延是十分重要的问题。

时分复用比率是用来衡量系统时钟周期使用情况的数值。在多FPGA原型系统的设计流程中，TDM比率通常是在FPGA布线后确定的。文献[6]提出了同时进行信号分割和分组的方法，可以优化分区和TDM比率。为了解决FPGA间的布线问题，文献[7]和文献[8]使用Pathfinder迭代地优化FPGA间的布线问题。然而，这两类工作都未考虑线网组的概念。文献[9]通过减少线长优化FPGA间的布线结果。然而，多FPGA系统的优化目标不仅仅是线长，TDM的比率分配也非常重要，因为系统性能很大程度上受到FPGA间线网的延迟影响。为了优化TDM比率，文献[10]提出了一种基于整数线性规划(Integer Linear Programming, ILP)的优化算法，但ILP只适合解决较小尺寸的问题。此外，以往工作中的TDM比率通常是任意整数，并不符合实际情况。

根据实际应用中的设计，本文提出一种用于时分复用技术的多阶段协同优化FPGA布线(Multi-Stage Co-optimization FPGA Routing, MSCOFRouting)方法，旨在满足TDM比率约束的前提下，布线拓扑生成阶段、TDM比率分配阶段和TDM比率优化阶段3个阶段协同优化FPGA间的可布线性和TDM比率。本文的主要贡献如下：

(1) 提出一种自适应布线算法，以避免布线拥塞，解决FPGA间布线优化问题，有力减少系统时延。

(2) 提出一种基于拉格朗日松弛(Lagrangian Relaxation, LR)的TDM比率分配算法，使小规模线网组获得较大TDM比率，大规模线网组获得较小TDM比率，有效解决TDM比率分配问题。

(3) 提出了一种TDM比率优化算法，缩减线网组和FPGA连接对的TDM比率。

(4) 将多线程并行化方法运用到上述3个算法，进一步提高MSCOFRouter的运行效率。

(5) 实验结果表明，本文算法不仅能满足TDM约束条件，而且可以获得比同类工作更好的解决方案。

后文组织如下：第2节介绍了时分复用技术和问题模型；第3节阐述了MSCOFRouting的整体框架。第4节给出本文相关策略的有效性验证及实验结果的比较分析。第5节总结全文。

2　问题模型

2.1　时分复用技术

使用时分复用技术可以有效地解决I/O引脚数量不足的问题，使得多个信号可以在不同时间段共用一个I/O引脚。然而，使用时分复用技术会造成系统时延的增加。TDM比率是用来衡量系统时钟周期使用情况的数值，可以作为衡量系统时延的指标。系统时延与TDM比率呈单调递增关系。因此，通过降低TDM比率可以有效减少系统时延。FPGA连接对p上边e的时分复用比率如式(1)所示。

$$\mathrm{TR}(e)=\frac{\mathrm{DN}(p)}{\mathrm{Cap}(p)} \tag{1}$$

其中，$\mathrm{DN}(p)$, $\mathrm{Cap}(p)$和$\mathrm{TR}(e)$分别代表通过FPGA连接对p的信号数、FPGA连接对p的容量和边e的

TDM比率。由于一个FPGA连接对间只有一根物理导线，所以FPGA连接对的容量固定为1。

图1是时分复用技术的示意图。图中的长方形、正方形和圆形分别表示转换器、实例和分区。红色箭头、蓝色箭头和绿色箭头分别表示3种不同的信号，两个FPGA中间的黑色箭头表示两个FPGA之间唯一的物理导线。在未使用时分复用技术的情况下，一个系统时钟周期内，一条物理导线只能传输一种信号。这会导致FPGA系统运行效率的下降。为了提升FPGA系统的运行效率，时分复用技术被运用在FPGA系统中，使得在一个系统时钟周期内可以传输3种不同的信号。

2.2 FPGA间的布线问题

2019年ICCAD比赛[11,12]提出的FPGA间布线问题将FPGA系统中的FPGA抽象为节点，忽略了一些FPGA的特殊性，着重解决FPGA间的布线问题。本文常用的符号如表1所示。

本节通过图2和表2的例子介绍FPGA间的布线问题。给定一组线网N和一组线网组NG。线网组是根据设计目的给定的。例如，具有相似属性或相同功耗的线网将在同一个线网组中。同时，给定一个无向图G，其中包括了多个由F表示的FPGA和多个由P表示的FPGA连接对。问题目标是根据无向图G对所有线网进行布线，并且为每条边分配合理的TDM比率，以最小化线网组的最大TDM比率。

表2给出了线网N和线网组NG的信息，其中，不同的线网用不同颜色表示。图2(a)是一个结构图。f_i代表第i个FPGA，p_k代表第k个FPGA连接对。图2(b)是表1基于图2(a)所生成的布线结果。

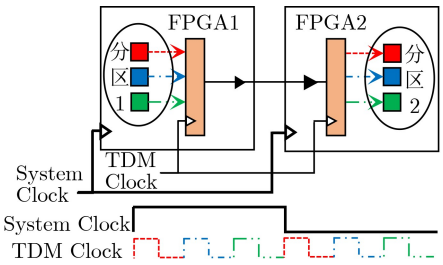


图 1 时分复用技术的示意图

$e_{j,k}$代表线网n_j经过连接对p_k所布的边。连接对p_k间边的数量就是通过连接对p_k的信号数量。为了分析系统时延，每条边$e_{j,k}$应该被分配一个TDM比率。一个FPGA连接对应该满足TDM比率约束，如式(2)所示。

$$\sum_{\mathrm{e}_{j,k}\in \mathrm{el}_k} \frac{1}{\mathrm{etr}_{j,k}} \le 1 \tag{2}$$

$$\mathrm{etr}_{j,k} \in \{x \,|\, x = 1 \times y, y \in N^*, 2 \le x\} \tag{3}$$

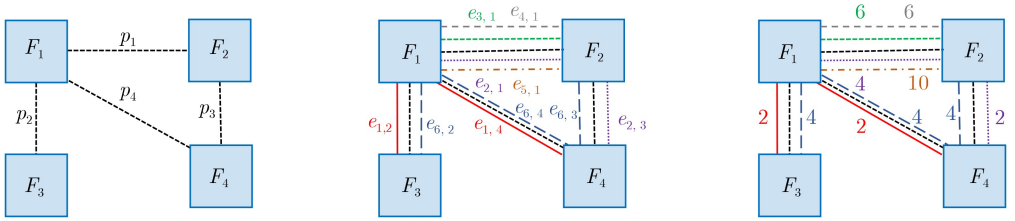
其中，$\mathrm{etr}_{j,k}$表示$e_{j,k}$的TDM比率。如图1所示，由于所有的信号必须在半个周期内完成1次传输，故$\mathrm{etr}_{j,k}$的值必须是偶数。每条边的TDM比率分配结果如图2(c)所示，对于p_1, $e_{2,1}$, $e_{3,1}$, $e_{4,1}$和$e_{5,1}$的TDM比率分别是4, 6, 6和10。由于1/4+1/6+1/6+1/10 <1，所以p_1满足TDM比率约束。

线网n_j的TDM比率和线网组ng_j的TDM比率定义为

$$\mathrm{ntr}_j = \sum_{e_{j,k}\in \mathrm{el}_j} \mathrm{etr}_{j,k} \tag{4}$$

表 1 常用符号

符号	含义
NG	所有线网组
N	所有线网
P	所有FPGA连接对
E	所有线网的边
Ne	经过边e的所有线网
ng_i	NG中第i个线网组
n_j	N中第j个线网
p_k	P中第k个FPGA连接对
ngl_j	包含n_j的线网组，$\mathrm{ngl}_j \subseteq$NG
$\mathrm{ng}_{j,m}$	ngl_j中的第m个线网组
$e_{j,k}$	n_j中使用p_k的边
el_k	p_k的边
el_j	n_j的边
nl_i	ng_i的线网
α	线网组的数量
mng_j	ngl_j中带有最大TDM比率的线网组



(a) 初始布线图 (b) 布线结果 (c) TMD比率分配结果

图 2 TDM比率分配示意图

表 2 线网组信息

线网组	线网	FPGA	示例
NG_1	N_1	F_3, F_4	
	N_2	F_1, F_4	
NG_2	N_3	F_1, F_2	
	N_4	F_1, F_2	
NG_3	N_1	F_3, F_4	
NG_4	N_5	F_1, F_2	
NG_5	N_6	F_2, F_3	

$$\mathrm{gtr}_j = \sum_{n_j \in \mathrm{nl}_i} \mathrm{ntr}_j \tag{5}$$

其中，ntr_j和gtr_j分别表示n_j的TDM比率和ng_i的TDM比率。如图2(c)所示，线网n_2上的边有$e_{2,1}$=4和$e_{2,3}$=2，所以ntr_2=$\mathrm{etr}_{2,1}$+$\mathrm{etr}_{2,3}$=2+4=6。线网组ng_2的包括n_3和n_4，所以gtr_2是12。

本文的优化目标是最小化TDM比率最大线网组的TDM比率，从而优化系统时延，具体如式(6)所示。

$$\min : \mathrm{gmt} = \{x | x = \max(\mathrm{gtr}_1, \cdots, \mathrm{gtr}_a)\} \tag{6}$$

其中，gmt代表了TDM比率最大的线网组的TDM比率。如图2(c)所示，gmt是12。

3　总体框架

用于时分复用技术的多阶段协同优化FPGA布线方法的总体框架如图3所示，分别为布线拓扑生成阶段、TDM比率分配阶段和TDM比率优化阶段3个阶段。具体地，第1阶段根据问题定义的线网和线网组进行布线，得到未分配TDM比率的布线结果。第2阶段使用拉格朗日松弛的方法为FPGA连接对的每条边分配初始的TDM比率。第3阶段通过松弛TDM比率较小的线网组，减小TDM比率倒数之和相对较小的FPGA连接对的最大TDM比率来优化TDM比率比较大的线网组。MSCOFRouting通过上述3个阶段协同优化FPGA的可布线性和TDM比率分配结果。

3.1　布线拓扑生成

布线拓扑生成阶段的目标是根据给定的线网和线网组定义将每个线网的FPGA连接在一起。布线生成的Steiner树的质量会影响后续的TDM比率分配。由于Dijkstra算法可以有效构建Steiner树[13,14]，因此本文使用基于Dijkstra算法的FPGA间布线算法连接所有线网，解决FPGA间的布线优化问题，使可布线性得到优化。算法中各变量的定义如下，f_s代表从F_n中选择的第1个FPGA、F_n表示线网n_j必须连接的目标FPGA、d表示两个FPGA间的布线代价、F_{all}表示所有FPGA、f_u表示与f_s布线代价最小的FPGA，f_v表示f_u的每个邻居FPGA，sp_v表示该线网的Steiner树。

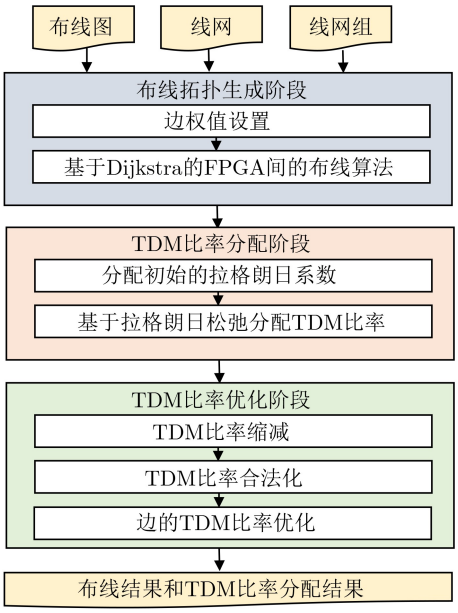


图 3 MSCOFRouting总体流程图

自适应布线算法如下所示。由于线网所在的线网组中的线网数量和线网需要连接的FPGA数量对于线网的布线方案有重要的影响。所以所有线网都按照这两个指标进行排序。然后，初始化布线图，每个FPGA连接对的初始布线代价为1。最后，对所有的线网进行迭代布线。

对所有的线网进行迭代布线的具体步骤如下。首先，从F_n中选择一个FPGA赋值给f_s。其次，初始化F_n中各FPGA与f_s的距离。然后，集合F_{all}等于集合F。最后，通过Dijkstra算法构造Steiner树。Dijkstra算法首先找到与F_{all}中与f_s代价最小的FPGA f_u，然后更新集合F_{all}，最后更新f_u的所有邻居节点的布线代价d_v。当找到所有目标FPGA后，Steiner树被构造出来，把sp_x记录下来。Steiner树所使用的FPGA连接对的布线代价更新公式为

$$\mathrm{Tp}'_k = \begin{cases} \mathrm{Tp}_k + t_e, \mathrm{Np}_k = 2t \\ \mathrm{Tp}_k + t_o, \mathrm{Np}_k = 2t+1 \end{cases} \tag{7}$$

其中，Tp_k代表FPGA连接对的布线代价，Np_k代表FPGA连接对上已布线的边数，t_e和t_o分别表示FPGA连接对上已布线的边数为偶数和奇数的更新代价。实验得出t_e, t_o分别取0.81和1.19时优化效果最好。

图4只考虑两个FPGA间的布线情况。两个FPGA间有1条或者2条边时，每条边分配的TDM比率都是2，最大TDM比率都是2；当两个FPGA间有3条边时，每条边分配的TDM比率分别是2, 4, 4。当两个FPGA间有4条边时，每条边分配的TDM比率分别是4, 4, 4, 4。最大TDM比率都是4。由此可

以得出结论：设i为奇数，当两个FPGA间有i条边时，布下1条边之后，最大TDM比率不变，为i+1；当两个FPGA间有i+1条边时，布下1条边之后，最大TDM比率值增加2，为i+3。所以布线代价更新式(7)可以减少两个FPGA间奇数条边的情况，优化布线结果，减少最大的TDM比率。

3.2 TDM比率分配

TDM比率分配阶段需要为每条边分配满足约束的TDM比率并且使TDM比率最大的线网组的TDM比率最小化。在这一阶段，本文提出了一种基于拉格朗日松弛算法的TDM比率分配方法。

由于优化目标是将线网组中最大的TDM比率最小化，所以问题模型可以写成

$$\left.\begin{aligned}&\min_{\mathrm{t}}\ \mathrm{gmt}\\&\text{s.t.}\ \sum_{n\in N}\sum_{e\in E}\mathrm{tr}_{\mathrm{ne}}\le\mathrm{gmt}\\&\quad\ \sum_{n\in\mathrm{Ne}}\frac{1}{\mathrm{tr}_{\mathrm{ne}}}\le 1\end{aligned}\right\}\tag{8}$$

其中，gmt是布线图中TDM比率最大的线网组的TDM比率，$\mathrm{tr}_{\mathrm{en}}$是线网$n$中边$e$的TDM比率。本文将此公式称为主问题(Primal Problem, PP)。为了解决这个NP难问题，算法松弛第1个约束并引入非负拉格朗日乘数λ。λ作为违反约束的惩罚值。式(8)引入λ后可以得到

$$L(t,\lambda,\mathrm{gmt})=\mathrm{gmt}+\sum_{\mathrm{ng}\in\mathrm{NG}}\lambda\left(\sum_{n\in N}\sum_{e\in E}\mathrm{tr}_{\mathrm{ne}}-\mathrm{gmt}\right)\tag{9}$$

对于给定的拉格朗日乘数，拉格朗日乘数子问题LRS(λ)如式(10)所示。

$$\left.\begin{aligned}&\min_{\mathrm{t}}\ L(t,\lambda,\mathrm{gmt})\\&\text{s.t.}\ \sum_{n\in\mathrm{Ne}}\frac{1}{\mathrm{tr}_{\mathrm{ne}}}\le 1\end{aligned}\right\}\tag{10}$$

通过应用Karush-Kuhn-Tucker (KKT)条件以获得最优解，LRS(λ)问题可以简化为

$$\left.\begin{aligned}&\mathrm{LRS}^*(\lambda):\ \min_{\mathrm{t}}\ \sum_{e\in E}\sum_{n\in\mathrm{Ne}}\sum_{g\in\mathrm{NG}_n}\lambda\times\mathrm{tr}_{\mathrm{ne}}\\&\text{s.t.}\ \sum_{n\in\mathrm{Ne}}\frac{1}{\mathrm{tr}_{\mathrm{ne}}}\le 1\end{aligned}\right\}\tag{11}$$

拉格朗日对偶问题(Lagrangian Dual Problem, LDP)可以定义为

$$\left.\begin{aligned}&\max\ \mathrm{LRS}^*(\lambda)\\&\text{s.t.}\ \sum_{n\in\mathrm{Ne}}\frac{1}{\mathrm{tr}_{\mathrm{ne}}}\le 1\end{aligned}\right\}\tag{12}$$

对给定λ集合，求解LDP就是求解下界的最大值。LDP的主要目的是找到合适的λ集合惩罚PP中违反的约束。拉格朗日乘子根据时序弧的时序临界性更新，以满足KKT条件[15–17]。更新公式为

$$\lambda_i=\lambda_i\times(\mathrm{TDM}_{i,g})^{K_{i,g}}\tag{13}$$

其中，$\mathrm{TDM}_{i,g}$为第i次迭代线网组g与最大线网组的TDM比率之比，$K_{i,g}$为第i次迭代的加速因子。加速因子越大，收敛速度越快，但是收敛效果越差。λ越接近目标值，加速因子应该越小。$K_{i,g}$的更新公式为

$$K_{i,g}=\frac{1}{1-(\mathrm{gmt}_i-\mathrm{str})/\mathrm{str}}\tag{14}$$

其中，str是由用户定义的gmt的优化目标。

TDM比率分配算法如下所示。首先，计算边$e_{j,k}$的所在FPGA连接对边的数量。接着，计算线网组ng_i的λ_i。然后，根据λ集合计算线网n_i的$\mathrm{sn}\lambda_i$值，计算分配给每条边的TDM比率，再更新λ集合。最后，根据式(13)更新λ_i并根据式(14)更新$K_{i,g}$。

3.3 TDM比率优化

在TDM比率分配阶段，通过拉格朗日松弛算法得到的初始TDM比率并非最优解，还存在优化空间。因此，TDM比率优化阶段可以通过增加TDM比率较小的线网组边的TDM比率以减少TDM比率较大的线网组边的TDM比率，并可以针对具体连接对进行优化系统时延，从而使TDM比率最大的线网组的TDM比率最小化。

TDM比率优化算法包括3个步骤。第1步是缩减操作。具体是增加TDM比率较小的线网组的TDM比率，减少TDM比率较大的线网组的TDM比率。第2步是合法化操作。经过缩减步骤后，TDM比率增加的FPGA连接对可能会违反TDM比率约束。因此，违反约束的FPGA连接对需要进行合法化操作。第3步是针对性优化操作。经过上述步骤后，所有FPGA连接对的TDM比率的倒数和与最大值1还有一定距离。通过减少连接对中TDM比率最大的$\mathrm{etr}_{j,k}$的方式，使得每个FPGA连接对的TDM比率倒数和更接近1，从而使TDM比率最大的线网组的TDM比率更小。

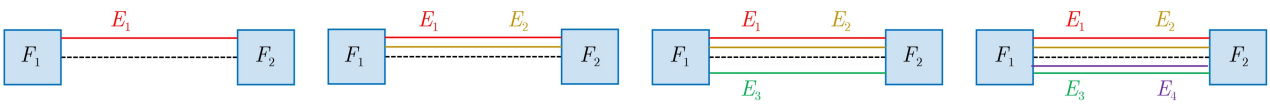


图 4 FPGA之间不同布线情况示意图

缩减操作的具体步骤如下所示。首先，所有线网按其所在线网组的最大TDM比率从大到小排序。然后，线网n_j中每条边的TDM比率$\mathrm{etr}'_{j,k}$根据以下公式更新。

$$\mathrm{etr}'_{j,k}=\begin{cases}\mathrm{etr}_{j,k}+f\left(\left(\dfrac{\mathrm{mng}_j}{\mathrm{str}}-1\right)\times\mathrm{etr}_{j,k}\right), \mathrm{mng}_j<\mathrm{str}\\ \mathrm{etr}_{j,k}-g\left(\left(1-\dfrac{\mathrm{mng}_j}{\mathrm{str}}\right)\times\mathrm{etr}_{j,k}\right), \mathrm{mng}_j>\mathrm{str}\end{cases} \tag{15}$$

其中，$\mathrm{etr}'_{j,k}$是$e_{j,k}$新的TDM的比率。$f(x)$表示大于等于x的最小偶数，$g(x)$表示小于等于x的最大偶数。

线网n_j的TDM比率减少后，更新ngl_j中各线网组的TDM比率，有利于后续的缩减。

第2步是对FPGA连接对进行合法化操作。经过缩减步骤，TDM比率增加的边$e_{j,k}$可以直接使用新的TDM比率$\mathrm{etr}'_{j,k}$；对于TDM比率减小的边$e_{j,k}$，如果p_k满足TDM比率约束，则用$\mathrm{etr}'_{j,k}$替换对应的p_k中的每条边的$\mathrm{etr}_{j,k}$。然而，如果p_k不满足TDM比率约束，应由式(16)合法化。

$$\mathrm{etr}''_{j,k}=\frac{\mathrm{etr}'_{j,k}\times\mathrm{rec}\times(\mathrm{rec}+\mathrm{ad})}{1-\mathrm{ad}} \tag{16}$$

其中，rec是$\mathrm{etr}'_{j,k}$减少的倒数之和，ad是$\mathrm{etr}'_{j,k}$增加的倒数之和。

算法第3步是减小具体FPGA连接对的最大TDM比率。经过前两个阶段后，当FPGA连接对边的TDM比率倒数和res_k小于0.95时，这个FPGA连接对的最大TDM比率由式(17)进行更新。

$$\mathrm{elt}'_{k,\max}=\frac{\mathrm{elt}_{k,\max}}{(1-\mathrm{res}_k)\times\mathrm{elt}_{k,\max}+1} \tag{17}$$

其中，res_k是FPGA连接对p_k上TDM比率倒数和，$\mathrm{elt}_{k,\max}$是FPGA连接对p_k上最大的TDM比率。

如果在缩减步骤中，当$\mathrm{mng}_j>\mathrm{str}$，减少的部分都向下取偶数会使得当前FPGA连接对的TDM倒数和更小，更有利于第3步FPGA连接对中最大TDM比率的减小。比如，当有3个线网组的TDM比率为4, 14和24，优化的目标值为8。如果不论mng_j是否大于str，变化值都向上取偶数，经过缩减阶段得出的TDM比率是6, 10和16，TDM倒数和为0.329；如果按照式(15)优化TDM比率，得到的TDM比率是6, 12和16，TDM倒数和为0.25。后者TDM比率倒数和小于前者，更有利于第3步FPGA连接对最大TDM比率的减小。所以选择式(15)的缩减方式。

3.4 多线程并行化

为了提高布线器的效率，使用并行编程模型OpenMp在布线器的各个阶段集成多线程并行化方法。编译器通过识别编译制导语句自动创建线程进行并行化，从而有效提高算法的效率。在布线拓扑生成阶段，各线网的布线操作可以并行执行。在TDM比率分配阶段，每个线网的TDM比率分配都是完全独立的。因此，这个阶段可以对不同线网进行并行化操作。在系统时延优化阶段，每个线网的缩减操作都是并行的。合法化操作和针对性优化中不同FPGA连接对可以并行处理。但是，由于不同线网之间存在资源冲突，自适应布线算法的记录sp_x操作和TDM比率优化算法的更新ngl_j中各线网组的TDM比率操作应该加锁。通过对上述3个阶段使用多线程并行化方法可以有效减少算法运行时间。

4 实验结果

所提出的优化框架采用C/C++语言实现，并在Intel Xeon Linux服务器上运行。本文在2019年的ICCAD竞赛[12]发布测试用例上展开实验。该测试用例将FPGA系统中的FPGA抽象为节点，忽略了一些FPGA的特殊性，着重解决FPGA间的布线问题。所以本文算法可以解决不同FPGA的布线问题，具有普适性。表3为测试用例具体信息，其中#FPGA表示FPGA数量，#Net表示线网数量，#NG表示线网组数量，#Edge表示FPGA连接对数量。

4.1 与ALIFRouter及MSFRoute的实验比较

本节参考ICCAD2019比赛[12]的计算评估分数公式将MSCOFRouting与ALIFRouter[11]及MSFRoute[18]进行比较。为了强调运行时间的影响，计算评估分数时考虑了运行时间。评估分数sco计算公式为

$$\mathrm{sco}=\mathrm{TR}\times\left(1+\lg\frac{\mathrm{RT}}{\mathrm{MRT}}\times0.01\right) \tag{18}$$

其中，TR为TDM最大的线网组的TDM比率，RT为运行时间，MRT为3个布线器运行时间的中位数。sum为所有测试用例的sco之和。布线器的sum越小表示它的优化效果越好。

如表4所示，MSCOFRouting获得了最好的sum，并且达到了最好的TDM比率和运行时间。与MSFRoute和ALIFRouter相比，MSCOFRouting的TDM比率分别降低了4.57%和1.05%，运行时间分别缩短了20.8%和0.44%。由于每个标准测试用例涉及多个线网组，总的布线边数量非常多，因此时钟周期数的基数很大，TDM比率数值很大。虽然TDM比率的优化率较小，但是TDM比率优化值较大，系统时延的优化效果较好，所以实验结果表明MSCOFRouting可以有效优化系统时延。

4.2 多线程并行化方法的有效性验证

MSCOFRouting在不同线程数下的运行时间如图5所示。不同的线段表示不同测试用例的运行时间。本节在具有典型性的中等规模测试用例S3, S4和H2上展开实验。通过使用8个线程、16个线程、24个线程和多线程并行化方法，MSCOFRouting分别可以得到3.11倍、3.82倍和4.08倍的加速。各线网间存在共用同一变量的情况。为了避免多个线程同时修改变量而导致数据错误，则需要对共享变量进行加锁操作。如果同时运行中的线程需要用到已使用的共享变量，则需要等待正在使用该资源的线程运行结束。所以随着线程数的增多，等待共享变量的时间逐渐增加，运行时间的优化率逐渐减少。

4.3 自适应布线算法有效性验证

为了验证自适应布线算法的有效性，本节将采用不同更新代价的自适应布线算法与未采用自适应布线算法的最大TDM比率进行比较。表5为采用不同权重值的自适应布线算法的实验结果。数据都是从同一环境中由8个线程运行的程序中获得的。表5中△TR的优化率的计算公式为

$$\Delta \mathrm{TR}_{t_e=x} = \mathrm{TR}_{t_e=x} - \mathrm{TR}_{\mathrm{MSF}} \tag{19}$$

不同的更新代价分别将△TR比率的优化率增加4.26%, 11.63%, 4.88%和10.93%。根据实验结果可知，当t_e=0.19, t_o=1.81时，自适应布线算法效果最好。由于较好的布线结果较少出现布线拥塞的情况，从而减少最大TDM比率。因此通过这些△TR的优化率可以得出自适应布线算法可以有效避免布线拥塞的情况出现，解决FPGA间的布线优化问题，优化布线结果，有效地减少系统时延。

4.4 LR分配算法和TDM比率优化算法有效性验证

本节将TDM比率分配算法和TDM比率优化算法运用到当前最先进的FPGA布线器ALIFRouter中进行比较。表6为LR分配算法和多层次的TDM比率优化算法的实验结果，并与ALIFRouter的实验数据进行对比。数据都是从同一环境中由8个线程运行的程序中获得的。表格中△TR的优化率是由MSFRoute在各测试用例上得到的线网组的最大TDM比率减去使用ALIFRouter、使用TDM比率分配算法和使用TDM比率优化算法后获得的最大TDM比率计算得到的。与ALIFRouter相比，两种

表 3 测试用例信息

测试用例	#FPGA	#Net	#NG	#Edge
S1	43	68456	40552	214
S2	56	35155	56308	157
S3	114	302956	334652	350
S4	229	551956	464867	1087
S5	301	881480	879145	2153
S6	410	785539	910739	1852
H1	73	54310	50417	289
H2	157	610675	501594	803
H3	487	720520	886720	2720

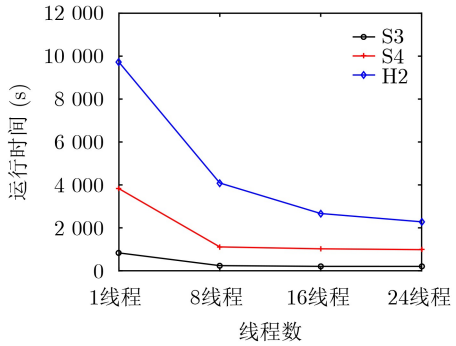


图 5 并行化方法有效性折线图

表 4 本文算法与ALIFRouter及MSFRoute的实验比较(TR为TDM Ratio)

测试用例	MSFRoute			ALIFRouter			本文算法		
	TR(10^3)	RT	sco	TR(10^3)	RT	sco	TR(10^3)	RT	sco
S1	39	34.32	39	38	24.88	38	37	30.58	37
S2	32097	20.51	32097	31736	19.58	31730	31671	20.86	31673
S3	129396	289.91	129375	127826	304.50	127832	127046	300.99	127046
S4	7301	642.51	7301	6466	641.62	6466	6312	619.88	6311
S5	5370	1673.51	5371	4766	1558.48	4766	4618	1570.51	4618
S6	15759857	4536.33	15757896	15751307	4668.18	15751307	15747236	4845.90	15749791
H1	409654	76.86	409661	409026	76.54	409026	408487	66.81	408246
H2	45947775	1322.96	45947798	45942757	1322.81	45942757	45934257	1217.81	45917758
H3	4871917	5283.91	4868324	4866484	6593.58	4867575	4861401	6261.79	4861401
sum	–	–	62289539	–	–	62273922	–	–	62245480
Normalized	1.0457	1.0208	–	1.0105	1.0044	–	1.0000	1.0000	–

表 5　自适应布线算法有效性验证

测试用例	采用(t_e=1, t_o=1)		采用(t_e=0.21, t_o=1.79)		采用(t_e=0.2, t_o=1.8)		采用(t_e=0.19, t_o=1.81)		采用(t_e=0.18, t_o=1.82)	
	△TR(10^3)	Ratio	△TR(10^3)	Ratio	△TR(10^3)	Ratio	△TR(10^3)	Ratio	△TR(10^4)	Ratio
S1	1	1.0000	1	1.0000	1	1.0000	2	2.0000	1	1.0000
S2	375	1.0000	417	1.1120	423	1.1280	428	1.1413	412	1.0987
S3	1841	1.0000	1864	1.0125	1948	1.0581	1986	1.0788	1941	1.0543
S4	832	1.0000	857	1.0300	931	1.1190	950	1.1418	927	1.1142
S5	604	1.0000	627	1.0381	697	1.1540	717	1.1871	687	1.1374
S6	11574	1.0000	13614	1.1763	12001	1.0369	12009	1.0376	12004	1.0372
H1	798	1.0000	801	1.0038	807	1.0113	812	1.0175	808	1.0125
H2	7742	1.0000	7785	1.0056	7795	1.0068	7821	1.0102	7801	1.0076
H3	6214	1.0000	6245	1.0050	9521	1.5322	10098	1.6250	9457	1.5219
平均	–	1.0000	–	1.0426	–	1.1163	–	1.2488	–	1.1093

表 6　LR分配算法(即TDM比率分配算法)和TDM比率优化算法有效性验证

测试用例	ALIFRouter		只使用LR分配算法		只使用TDM比率优化算法	
	△TR(10^3)	Ratio	△TR(10^3)	Ratio	△TR(10^3)	Ratio
S1	1	1.0000	1	1.0000	1	1.0000
S2	384	1.0000	421	1.0964	419	1.1607
S3	1877	1.0000	1960	1.0442	2155	1.3726
S4	848	1.0000	889	1.0483	937	1.1222
S5	621	1.0000	635	1.0225	690	1.1424
S6	11994	1.0000	18550	1.5466	11989	1.4022
H1	799	1.0000	804	1.0063	804	0.0022
H2	7775	1.0000	7776	1.0001	7776	1.5496
H3	6220	1.0000	6419	1.0320	6429	1.1833
平均	–	1.0000	–	1.0885	–	1.1039

算法可分别将ΔTR比率的优化率增加8.85%和10.39%。这些ΔTR的优化率表明LR分配算法和TDM比率优化算法可以有效地降低系统时延。

5　结束语

针对FPGA系统中运用TDM技术后导致系统时延增加的问题，本文提出了一种用于时分复用技术的多阶段协同优化FPGA布线方法，通过布线拓扑生成阶段、TDM比率分配阶段和TDM比率优化阶段3个阶段协同优化FPGA的可布线性和TDM比率分配结果。首先，为了生成高质量的布线拓扑，避免布线拥塞，解决FPGA间的布线优化问题，提出了自适应布线算法。其次，使用基于拉格朗日松弛的TDM比率分配算法，为布线图的边分配系统时延更小的初始TDM比率。然后，为了进一步减小最大线网组的TDM比率，通过一种多层次的TDM比率优化算法，同时缩减线网组和FPGA连接对的TDM比率。并且，为了提高MSCOFRouting的运行效率，在上述3个算法中使用多线程并行化方法，降低算法的运行时间。实验结果表明，与同类布线器相比，本文提出的MSCOFRouting能够获得最佳的系统时延优化质量。未来的工作中，将扩展本文算法应用到考虑带高速收发器的FPGA系统的TDM比率优化问题。

参 考 文 献

[1] TURKI M, MARRAKCHI Z, MEHREZ H, *et al*. Signal multiplexing approach to improve inter-FPGA bandwidth of prototyping platform[J]. *Design Automation for Embedded Systems*, 2015, 19(3): 223–242. doi: 10.1007/s10617-014-9155-4.

[2] LING A and ANDERSON J. The role of FPGAs in deep learning[C]. Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, USA, 2017: 3. doi: 10.1145/3020078.3030013.

[3] CHIU G R, LING A C, CAPALIJA D, *et al*. Flexibility:

FPGAs and cad in deep learning acceleration[C]. Proceedings of the 2018 International Symposium on Physical Design, Monterey, USA, 2018: 34–41. doi: 10.1145/3177540.3177561.

[4] HUNG W N N and SUN R. Challenges in large FPGA-based logic emulation systems[C]. Proceedings of the 2018 International Symposium on Physical Design, Monterey, USA, 2018: 26–33. doi: 10.1145/3177540.3177542.

[5] HAUCK S. The roles of FPGAs in reprogrammable systems[J]. *Proceedings of the IEEE*, 1998, 86(4): 615–638. doi: 10.1109/5.663540.

[6] CHEN S C, SUN R, and CHANG Yaowen. Simultaneous partitioning and signals grouping for time-division multiplexing in 2.5D FPGA-based systems[C]. 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Diego, USA, 2018: 1–7. doi: 10.1145/3240765.3240847.

[7] TURKI M, MARRAKCHI Z, MEHREZ H, *et al*. Iterative routing algorithm of inter-FPGA signals for multi-FPGA prototyping platform[C]. The 9th International Symposium on Reconfigurable Computing: Architectures, Tools and Applications, Los Angeles, USA, 2013: 210–217. doi: 10.1007/978-3-642-36812-7_20.

[8] FAROOQ U, CHOTIN-AVOT R, AZEEM M, *et al*. Inter-FPGA routing environment for performance exploration of multi-FPGA systems[C]. 2016 International Symposium on Rapid System Prototyping (RSP), Pittsburgh, USA, 2016: 1–7. doi: 10.1145/2990299.2990317.

[9] PUI C W and YOUNG E F Y. Lagrangian relaxation-based time-division multiplexing optimization for multi-FPGA systems[J]. *ACM Transactions on Design Automation of Electronic Systems*, 2020, 25(2): 21. doi: 10.1145/3377551.

[10] INAGI M, TAKASHIMA Y, and NAKAMURA Y. Globally optimal time-multiplexing in inter-FPGA connections for accelerating multi-FPGA systems[C]. 2009 International Conference on Field Programmable Logic and Applications, Prague, Czech Republic, 2009: 212–217. doi: 10.1109/FPL.2009.5272309.

[11] ZHUANG Zhen, HUANG Xing, LIU Genggeng, *et al*. ALIFRouter: A practical architecture-level inter-FPGA router for logic verification[C]. 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2021: 1570–1573. doi: 10.23919/DATE51398.2021.9474255.

[12] SU Y H, SUN R, and HO P H. 2019 CAD contest: System-level FPGA routing with timing division multiplexing technique[C]. 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Westminster, USA, 2019: 1–2. doi: 10.1109/ICCAD45719.2019.8942051.

[13] FRESCHI V and LATTANZI E. A prim–Dijkstra algorithm for Multihop calibration of networked embedded systems[J]. *IEEE Internet of Things Journal*, 2021, 8(14): 11320–11328. doi: 10.1109/JIOT.2021.3051270.

[14] CHEN Xiaohua, ZHOU Ruping, LIU Genggeng, *et al*. Timing-driven X-architecture Steiner minimum tree construction based on social learning multi-objective particle swarm optimization[C]. Companion Proceedings of the Web Conference, Ljubljana, Slovenia, 2021: 77–84. doi: 10.1145/3442442.3451143.

[15] KULIK A, SHACHNAI H, and TAMIR G. On Lagrangian relaxation for constrained maximization and reoptimization problems[J]. *Discrete Applied Mathematics*, 2021, 296: 164–178. doi: 10.1016/j.dam.2020.10.001.

[16] SHARMA A, CHINNERY D, DHAMDHERE S, *et al*. Rapid gate sizing with fewer iterations of Lagrangian relaxation[C]. 2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Irvine, USA, 2017: 337–343. doi: 10.1109/ICCAD.2017.8203797.

[17] MANGIRAS D, STEFANIDIS A, SEITANIDIS I, *et al*. Timing-driven placement optimization facilitated by timing-compatibility flip-flop clustering[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020, 39(10): 2835–2848. doi: 10.1109/TCAD.2019.2942001.

[18] ZHUANG Zhen, LIU Genggeng, HUANG Xing, *et al*. MSFRoute: Multi-stage FPGA routing for timing division multiplexing technique[C/OL]. Proceedings of the 2020 on Great Lakes Symposium on VLSI, 2020: 107–112. doi: 10.1145/3386263.3406902.

刘耿耿：男，博士，副教授，研究方向为微流控生物芯片及VLSI设计自动化.

许文霖：男，硕士生，研究方向为EDA算法.

周茹平：女，硕士生，研究方向为超大规模集成电路布线.

徐 宁：男，博士，教授，研究方向为电子设计自动化、计算机软件与系统.

责任编辑：陈 倩