doi:10.11887/j.cn.201801010 http://journal.nudt.edu.cn

时序驱动的详细布局方法*

刘 畅[1],郭泽晖[1],贺 旭[1,2],郭 阳[1]
(1. 国防科技大学 计算机学院,湖南 长沙 410073;2. 湖南大学 信息科学与工程学院,湖南 长沙 410082)

摘 要:针对超大规模集成电路布局过程中时序优化问题,提出一种时序驱动的详细布局方法。对设计进行时序分析并获取时序违反路径集合,对路径上两个连续固定单元间的线网进行平滑处理,以减小路径曲折度以及减少线长。再针对每一个可移动单元与其相邻的线网建立二次规划时序模型,求解局部最优布局位置。对于给定的测试电路,实验结果表明,最差的时序违反与总的时序违反均有明显改善,采用ICCAD 2015竞赛的测试模板和评价方法,总的时序性能有45~350 min的提升。

关键词:时序驱动;详细布局;时序优化;松弛度;埃尔莫尔延时模型
中图分类号:TP391 **文献标志码**:A **文章编号**:1001-2486(2018)01-067-07

Timing-driven method for detailed placement

LIU Chang[1], *GUO Zehui*[1], *HE Xu*[1,2], *GUO Yang*[1]
(1. College of Computer, National University of Defense Technology, Changsha 410073, China;
2. College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China)

Abstract: To cope with the timing problem of placement in the very large integrated circuit, a timing-driven optimization method for placement was proposed. Firstly, the design was analyzed by a timing evaluation tool and the timing violation paths were collected. A rough placement method was used on the moved cells between any two successive fixed cells in those paths to smooth the nets. After that, a detailed placement based on quadratic timing model was used to optimize the timing characteristics. For the given benchmarks and the evaluation method in ICCAD 2015 contest, the experimental results show that both the worst negative slack and the total negative slack are improved, and the overall timing performance is improved by 45~350 min.

Key words: timing driven; detailed placement; timing optimization; slack; Elmore delay mode

在超大规模集成电路(Very Large Scale Intergrated,VLSI)设计过程中,时序收敛是完成设计的一个标志,通常要考虑的因素包括时序约束、保持/建立时间、时钟频率等。从物理设计前期的规划到后期的版图布局布线[1],每一个环节都应考虑时序收敛问题。布局是物理设计中最重要的一步,一般布局方法考虑的因素主要包括线长、面积、密度、温度和可布线性等,对时序考虑较少。虽然以线长驱动或可布线性驱动的算法也可以优化时序性能,但是在某些情况下,效果并不理想[2]。时序驱动布局(Timing-Driven Placement,TDP)在布局时将时序特征作为优化目标之一,可以在布局阶段有效提高电路时序性能,有助于时序收敛。

TDP采用的方法主要可以分为三类:基于线网的方法、基于路径的方法和混合方法[3]。基于线网的方法将时序信息转化为线的权重或者线约束,以加权后的线长为优化目标,通过迭代的方法使全部加权网络长度最小化[4-7],线的权重可以静态或动态分配。静态分配时计算松弛时间(slack),这在布局过程中是不变的。文献[8]提出了一个以敏感度为目标的加权线网方法,通过迭代减少总的负松弛时间(Total Negative Slack,TNS)。动态分配线网权重时线网的权重会在布局过程中进行调整。在文献[9]中,使用拉格朗日乘子作为线网权重,动态地进行时序更新,采用离散的局部搜索方法来优化时序。基于路径的方法是在布局时,不断对电路进行时序分析[10],选出待优化的关键路径,使用线性规划方法最小化TNS[11]。相比前者,后者更准确地表示了时序驱动布局问题。由于时序违反的路径数目可能是指数级,文献[12]在构造线网约束的线性规划方程

* **收稿日期**:2016-11-24
基金项目:国家自然科学基金资助项目(61133007,61402505)
作者简介:刘畅(1988—),男,内蒙古巴彦淖尔人,博士研究生,E-mail:chancews@qq.com;
郭阳(通信作者),男,研究员,博士,博士生导师,E-mail:guoyang@nudt.edu.cn

时使用目标时序来减少路径数目。文献[13]中采用 Bezier 曲线来平滑路径,同时利用空闲空间感知技术保障布局的合法性。文献[14]使用基于松弛时序的线网加权方法评估关键线网,同时采用模拟退火方法来平衡非关键路径的线长和关键路径的时序。文献[15]为关键路径上的双端线网赋予更高的权重,以此来使关键路径更平滑。

TDP 主要面临的问题有三个:①布局器的稳定性。一些全局布局方法会从根本上改变版图的结构,可能会彻底破坏原时序信息,尤其是在设计的早期阶段;②时序改善的收敛性。TDP 过程中时序性能改善是难以收敛的,这是因为改善子路径松弛度不能保证 TNS 或者最差负松弛度(Worst Negative Slack,WNS)性能的提升,而且改善一条违反路径可能会使一些其他原来合格的路径变成违反路径;③对原始布局优化的保护。时序约束往往与传统布局目标发生冲突,例如线长、单元密度、布通率等,为了满足时序要求可能会违反原来布局中的约束要求。例如,文献[16]提出的算法在每次移动单元后预测 Steiner 树的拓扑结构,以保证在优化时序的同时不破坏原有优化结果。

本文提出的 TDP 算法包括粗粒度布局和细粒度布局两个部分。粗粒度方法通过多次迭代对固定单元间的路径进行平滑处理。细粒度利用 Elmore 公式对线网延迟进行建模,通过求解二次方程最小值来对线网时序进行优化。采用 ICCAD 2015 竞赛给出的测试模板作为基准测试程序,对算法进行测试,实验结果表明,与原始布局相比,采用本文方法布局后的结果有 45 ~ 350 min的提升。

1 问题描述

时序驱动布局算法依赖于精确的时序模型、松弛度计算模型和时序传播模型。采用静态时序分析(Static Timing Analysis,STA)可计算从输入端到输出端整条路径上信号传播时间,包括组合单元、时序单元和互连线的延迟、偏斜和松弛度等。本文采用 ICCAD 2014 竞赛中提供的方法对结果进行静态时序分析及性能评价[2],本章将简述电路单元时序模型、时序传播模型、松弛度计算及布局性能评价方法。

1.1 时序模型

考虑到工业制造的复杂性,一般情况下,时序分析模型通常采用最小延时模式(early 模式)和最大延时模式(late 模式),这两种模式是时序的上下边界。

线网的时序模型。线网可以看作是由一个输入引脚和多个输出引脚组成的。信号从出入引脚传播到各个输出引脚的时延可以使用 Elmore 公式进行估算[17]。

图 1 所示为一个简单的三端线网和对应时序模型,根据 Elmore 公式,输出引脚的延时为:

$$d_e = \sum_{k \in N} R_{ke} C_k \tag{1}$$

其中,e 是线网内任意一点,R_{ke}是线网内点 k 和 e 共同路径的电阻,C_k 是点 k 对地电容。

输出偏斜(slew)也是线网时序特征的一个重要特征,其计算可以分为两步[2],第一步:

$$\bar{s}_{oT} \approx \sqrt{2\beta_T - d_T^2} \tag{2}$$

式中,d_T 是由 Elmore 公式计算出的延时,第二步,计算参数 β_T,可以利用公式:

$$\beta_e = \sum_k R_{re} C_k d_k \tag{3}$$

若输入端的偏斜为 s_i,则输出端的偏斜 s_{oT}可以近似表示为:

$$s_{oT} \approx \sqrt{s_i^2 + \bar{s}_{oT}^2} \tag{4}$$

组合电路单元建模。组合电路中单元延时和输出端的偏斜可用式(5)和式(6)计算[2]:

$$d = a + bC_L + cs_1 \tag{5}$$

$$s_o = x + yC_L + zs_1 \tag{6}$$

式中:a、b、c、x、y 和 z 均为与单元相关的参数,由 ICCAD 2014 竞赛提供;C_L 是单元 P 的负载电容,包括 RC 网络中的寄生电容和输出端口的电容,因此

$$C_L = \sum_{k \in N} C_k \tag{7}$$

时序电路单元建模。时序单元在时钟信号同

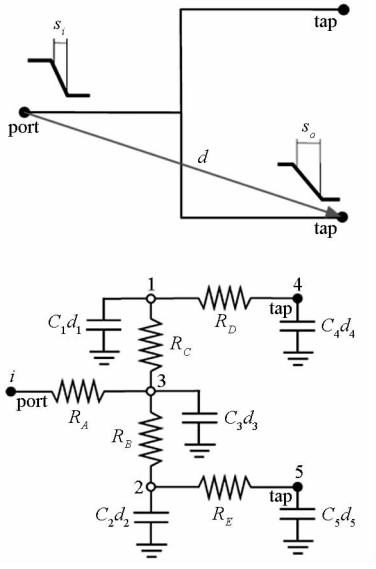

图 1 简单线网及其对应的时序模型

Fig. 1 Simple net and its timing model

步下进行采样,输入端信号在时钟到来时需要满足建立时间(setup time)和保持时间(hold time),二者都与时钟和数据信号的偏斜相关,可以用式(8)和式(9)进行估算[2]。

$$t_{\text{setup}} = g + h \cdot s_{iCK}^{\text{E}} + j \cdot s_{iD}^{\text{L}} \tag{8}$$

$$t_{\text{hold}} = m + n \cdot s_{iCK}^{\text{L}} + p \cdot s_{iD}^{\text{E}} \tag{9}$$

式中:g、h、j、m、n 和 p 是与时序单元相关的参数;s_{iCK}和 s_{iD}分别是时钟信号和数据信号的偏斜,由 ICCAD 2014 竞赛提供。

1.2　松弛度和时序传播模型

电路的时序性能通常用松弛度(slack)来表征,松弛度由信号到达时间(Arrive Time,AT)和要求到达时间(Required Arrive Time,RAT)计算得到。

到达时间的计算是从输入端开始,沿信号传播方向累加单元和线网的时延,计算每个节点处 early 模式(late 模式)下的最小值(最大值),得到信号实际到达每个节点的时间范围。对于组合单元,假设 A 和 B 是单元的输入,Y 是单元的输出,$at^{\text{E}}(A)$、$at^{\text{E}}(B)$、$at^{\text{L}}(A)$、$at^{\text{L}}(B)$分别定义为信号在节点 A 和节点 B 处 early 模式和 late 模式的到达时间,则 Y 处在 early 模式和 late 模式下的到达时间为:

$$at^{\text{E}}(Y) = \min[at^{\text{E}}(A) + d^{\text{E}}(A,Y), at^{\text{E}}(B) + d^{\text{E}}(B,Y)] \tag{10}$$

$$at^{\text{L}}(Y) = \max[at^{\text{L}}(A) + d^{\text{L}}(A,Y), at^{\text{L}}(B) + d^{\text{L}}(B,Y)] \tag{11}$$

对于时序单元,假设时钟延迟为 l_i,时序单元内部延迟为 $d_{CK->Q}$,组合逻辑部分延时为 d_{comb},则时序单元数据输出端 D 的到达时间为:

$$at^{\text{E}}(D) = l_{\text{i}}^{\text{E}} + d_{CK->Q} + d_{\text{comb}}^{\text{E}} \tag{12}$$

$$at^{\text{L}}(D) = l_{\text{i}}^{\text{L}} + d_{CK->Q} + d_{\text{comb}}^{\text{L}} \tag{13}$$

要求到达时间的计算从输出端开始,沿电路传播方向逆向递减路径上单元和线网的时延。对于组合电路,假设输出端为 T_1 和 T_2,输入端为 Z,$rat^{\text{E}}(T_1)$、$rat^{\text{E}}(T_2)$、$rat^{\text{L}}(T_1)$、$rat^{\text{L}}(T_2)$分别定义为信号在节点 T_1 和节点 T_2 处 early 模式和 late 模式下的要求到达时间。节点 Z 在 early 模式和 late 模式下的要求到达时间分别为:

$$rat^{\text{E}}(Z) = \max[rat^{\text{E}}(T_1) - d_{Z->T_1}^{\text{E}}, rat^{\text{E}}(T_2) - d_{Z->T_2}^{\text{E}}] \tag{14}$$

$$rat^{\text{L}}(Z) = \min[rat^{\text{L}}(T_1) - d_{Z->T_1}^{\text{L}}, rat^{\text{L}}(T_2) - d_{Z->T_2}^{\text{L}}] \tag{15}$$

对于时序单元,输入端信号必须满足建立时间 t_{setup}和保持时间 t_{hold}的要求,假设时序单元信号输入端为 D,时钟周期为 P,则时序单元要求到达时间为:

$$rat^{\text{E}}(D) = l_o^{\text{L}} + t_{\text{hold}} \tag{16}$$

$$rat^{\text{L}}(D) = P + l_o^{\text{E}} - t_{\text{setup}} \tag{17}$$

对于 early 模式和 late 模式,松弛度计算分别如式(18)和式(19)所示,对于一个设计,松弛度为正,则满足时序约束,若为负,则不满足。

$$slack^{\text{E}} = at^{\text{E}} - rat^{\text{E}} \tag{18}$$

$$slack^{\text{L}} = rat^{\text{L}} - at^{\text{L}} \tag{19}$$

1.3　性能评价标准

本文采用 UI-Timer[1] 对电路时序性能进行计算,同时参照 ICCAD 2014 和 ICCAD 2015 竞赛,以松弛度的改善作为算法性能的评价指标。松弛度改善的评价主要关注两个方面:WNS 和 TNS。WNS 体现了电路中时序性能最差节点的违反程度,而 TNS 则体现了电路时序总体的违反程度。考虑布局密度和算法运行时间,算法性能评价公式为:

$$\begin{aligned} slack\_improv = {} & w_{\text{TNS}} \times (w_{\text{late}} \times improv_{\text{TNS}}^{\text{L}} + \\ & w_{\text{early}} \times improv_{\text{TNS}}^{\text{E}}) + \\ & w_{\text{WNS}} \times (w_{\text{late}} \times improv_{\text{WNS}}^{\text{L}} + \\ & w_{\text{early}} \times improv_{\text{WNS}}^{\text{E}}) \end{aligned} \tag{20}$$

式中,$improv_{\text{TNS}}^{\text{L}}$、$improv_{\text{TNS}}^{\text{E}}$、$improv_{\text{WNS}}^{\text{L}}$和 $improv_{\text{WNS}}^{\text{E}}$分别是 TNS 和 WNS 在 late 模式和 early 模式下的改善率,w_{TNS}、w_{WNS}、w_{late}、w_{early}为计算权值。

2　时序优化布局方法

2.1　时序优化布局算法框架

本文布局算法的整体流程如图 2 所示,程序首先初始化 UI-Timer 并获得原始设计的时序违反路径集合 P_s,接着对每一条路径上任意两个连续固定节点间的可移动单元进行分析,调用粗粒度算法;然后针对每一个可移动单元进行精细的时序调整,每一次调整都是以局部最优为目标;调整完成后,更新节点位置,判断是否可以结束,如果没有达到结束条件,则再次运行 UI-Timer 并重新获取时序违反的路径集合 P_s,重复到第二步执行,如果达到了结束条件,结束程序。

2.2　粗粒度布局优化

粗粒度布局优化的核心思想是,尽可能减少电路路径的曲折程度,减少线长,进而能够改善时序性能。具体实现方法如图 3 所示。

如图 3 左图中,黑色方块代表固定节点,浅灰色方块代表可移动节点,网格区域代表可移动的最优区域。由于固定节点的存在,为尽可能减少路径曲折度,则期望将固定节点间的可移动单元放置在

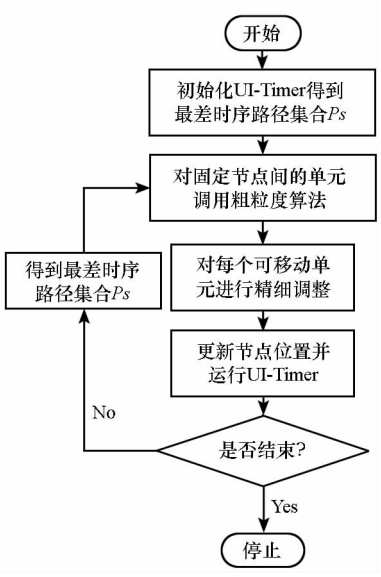

图 2 时序优化布局流程

Fig. 2 Process of timing driven placement

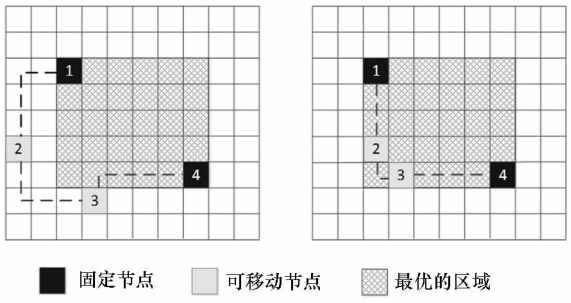

图 3 粗粒度摆放前(左)及粗粒度摆放后(右)

Fig. 3 Before placement(left) and after placement(right)

尽可能靠近固定节点连接线附近,如果考虑曼哈顿距离,则将可移动单元摆放在最优区域内即可,图 3左图中情况在调整摆放后如图 3 右图所示。

进行粗粒度优化时需要注意以下几点:

1)最优区域是由固定节点确定的回形区域,可移动单元随机放置在该区域未被占用的位置;

2)随机位置的选择符合均匀分布,但放置后的目标是使两两单元间曼哈顿距离最小;

3)移动时,两个固定单元之间的可移动单元,按照从输入到输出的顺序对单元进行编号$(1,\cdots,n)$,然后两个一组进行移动,例如 1 和 n 为一组;

4)布局完成后要对结果进行合法性检查。检查的目标包括重叠、对齐、密度和顺序排列等。

2.3 细粒度布局优化

细粒度布局优化的核心思想是利用 Elmore 公式对可移动单元所连线网计算线延迟,建立二次规划模型,并以此计算移动单元对线网时序造成的影响,由此判断单元移动的方向和距离。

在式(1)中,时序模型里电阻 R 和电容 C 的简化估计与线段的长度成正比,即

$$R_i = rL_i \tag{21}$$

$$C_i = cL_i \tag{22}$$

其中,r 和 c 是在给定工艺下单位长度的电阻、电容值,L_i 是线网段长度。利用 Elmore 公式,得图 1 中输出节点 5 处的线网延迟 d_5,则

$$d_5 = \rho(L_A^2 + L_B^2 + L_E^2 + L_AL_B + L_AL_C + L_AL_D + L_AL_E + L_BL_E) \tag{23}$$

其中 $\rho = rc$,假设图 1 的输出端 5 连接的是一个可移动单元,在不移动其他点的条件下,延时可以写为

$$d_5 = \rho(L_E^2 + \gamma L_E + \delta) \tag{24}$$

其中,$\delta = L_A^2 + L_B^2 + L_AL_B + L_AL_C + L_AL_D$,$\gamma = L_A + L_B$。

对每一个可移动单元进行局部移动,考虑如图 4的一种简单情况,可移动单元 P 是一个二输入单输出的单元,P 与三个线网 NET 1,NET 2 和 NET 3 相连。假设只移动 P 而不改变线网中其他点的位置,且移动 P 时,只能沿着连接线方向移动。

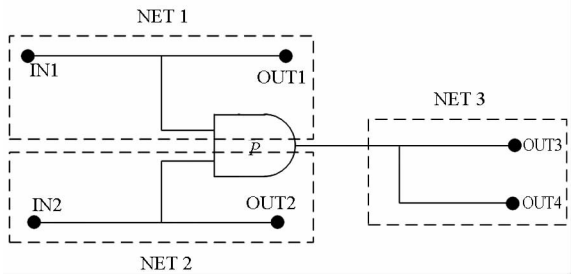

图 4 局部时序移动简单模型

Fig. 4 Timing model of local movement

P 的移动可以使得与 P 相连 NET 的端延时发生变化,图 5 所示为图 4 网络对应的 RC 模型,当 P 沿着图 5 中虚线方向向左移动时,输入 IN1 到结点 4 和输入 IN2 到结点 8 的延时减小,P 的输出端到 OUT3 和 OUT4 的延时增大;当 P 沿着虚线方向向右移动时,输入 IN1 到结点 4 和输入 IN2 到结点 8 的延时增大,而 P 的输出端到 OUT3 和 OUT4 的延时减小。

考虑从 IN1 到 OUT4 的时序路径,计算总的延时可以分成两部分,由式(24)得到单元 P 之前的部分延时及 D_1 对 L_{1E}的差分形式可以写为:

$$D_1 = \rho(L_{1E}^2 + \gamma_1 L_{1E} + \delta_1) \tag{25}$$

$$\Delta d_1 = \Delta L_{1E}(2\rho L_{1E} + \rho\delta_1) \tag{26}$$

同理,单元 P 之后部分的延时及差分形式为:

$$D_2 = \rho(L_{2E}^2 + \gamma_2 L_{2E} + \delta_2) \tag{27}$$

$$\Delta d_2 = \Delta L_{2E}(2\rho L_{2E} + \rho\delta_2) \tag{28}$$

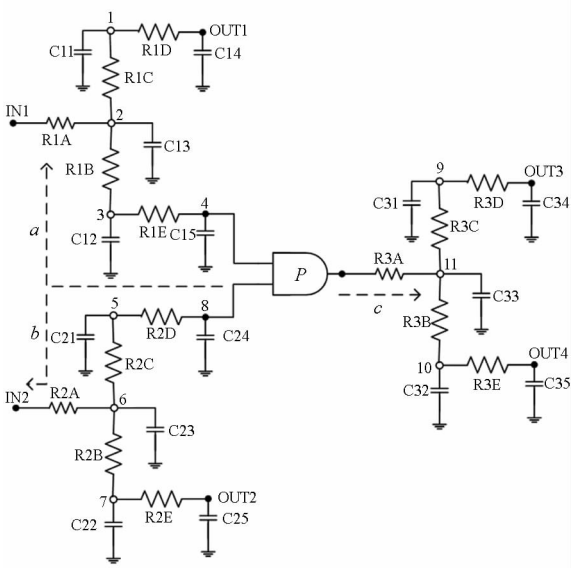

图 5　局部时序移动简单模型 RC 结构

Fig. 5　RC model of local movement

通过比较 Δd_1 和 Δd_2 表达式括号里的系数可以确定单元 P 的移动方向，当$(2\rho L_{1E} + \rho\delta_1) > (2\rho L_{2E} + \rho\delta_2)$时，单元 P 沿着图 5 中虚线 a 的方向移动可以减小路径总延时；当$(2\rho L_{1E} + \rho\delta_1) < (2\rho L_{2E} + \rho\delta_2)$时，单元 P 沿着虚线 c 的方向可以减小路径总延时。

移动单元 P 会改变与之相连的线网的电容，而单元 P 的延时与反转延时与其负载电容和输入反转延时有密切关系，因此，在判断单元 P 移动方向时还需考虑移动 P 时负载电容和输入反转延时的变化。单元 P 的延时 d 和反转延时 s_o 可以近似地估计为：

$$d = a + bC_L + cs_i \tag{29}$$

$$s_o = x + yC_L + zs_i \tag{30}$$

式中：a，b，c，x，y 和 z 均为参数；C_L 是单元 P 的负载电容，包括 RC 网络中的寄生电容和输出端口的电容，因此

$$C_L = \sum_{k \in N} C_k \tag{31}$$

式中，N 是负载 RC 网络中全部点的集合。移动单元 P 对其延时和反转延时的影响是复杂的，体现在移动单元 P 减小 C_L 的同时会增加 s_i，而 s_i 减小的同时 C_L 会增加，由式(4)可以看到，要找到一个确切的表达式来判断 s_i 和 C_L 的变化趋势是复杂而费时的，因此，可以记移动单元 P 引起单元 P 的延时变化为 Δd_p，移动单元 P 之后计算 $\Delta d_1 - \Delta d_2$(或者 $\Delta d_2 - \Delta d_1$)与 Δd_p 的差值，如果为正，则接受该次移动，如果为负则不接受该次移动。以此来修正每次移动可能引入的错误。同理，更一般的情况可以用相同的方法求解。

由于单元 P 以及和 P 相连的线网中存在多条时序路径，如图 5 中就存在 IN1-OUT1、IN1-OUT3、IN1-OUT4、IN2-OUT2、IN2-OUT3 和 IN2-OUT4 这六条路径，其中经过单元 P 的有四条，不经过单元 P 的有两条，主要关注经过单元 P 的路径，整个电路的关键路径可能经过单元 P 也可能不经过，线网中可能存在整个电路的 WNS，因此需要分情况考虑：

1)线网中没有整个电路的关键路径，也没有 WNS。按路径延时长短将所有路径排序，找到延时最长的路径，移动单元 P 使得 P 两端延时达到均衡。移动后如果该路径仍是线网，延时最长的路径，则不再对 P 进行移动，否则继续重复 1)操作；

2)线网中有整个电路的关键路径，没有 WNS。找到关键路径，移动单元 P 使得 P 两端延时达到均衡，若移动后该路径仍是关键路径，则不再对 P 进行移动，否则继续重复 2)操作；

3)线网中没有整个电路的关键路径，有 WNS。找到 WNS 所在的路径，移动单元 P 使得 P 两端达到均衡，若移动后 WNS 仍在该路径，则不再对 P 进行移动，否则继续对 WNS 所在路径进行移动；

4)线网中既有整个电路的关键路径，又有 WNS。找到关键路径，移动单元 P 使得 P 两端延时达到均衡，若移动后该路径仍是关键路径，则考虑 WNS 所在路径，若 WNS 所在路径已经移动过，则不再移动 P，否则，重复 3)中步骤。

3　实验结果与分析

3.1　实验环境

本文局部时序优化方法实验环境为 Intel Xeon CentOS 6.5 工作站，采用 C++完成代码的实现，g++ 4.8.2 进行编译。实验采用的测试基准电路共七套电路，全都运行在 2.60 GHz、64 GB 内存的服务器上。

3.2　测试模板

本文使用的测试模板是 ICCAD 2015 竞赛测试电路，2015 年的竞赛与 2014 年竞赛时序驱动布局问题相同，但是 2015 年的竞赛提供的测试模板更加丰富，增加了不规则形状的固定单元，且来自于当代工业专用集成电路(Application Specific Integrated Circuit，ASIC)设计，有很好的物理设计特征，具有很高的代表性。其参数信息包括集成电路设计的基本信息，例如标准单元的数目和坐

标、线网数目、单元之间的互连关系、布局区域参数、各单元的形状大小和布局约束等。ICCAD 2015 测试模板信息如表 1 所示，其中 Utility 是指设计利用率，定义为所有标准单元总面积占可布局空白区域总面积的百分比。

布局结果如表 2 所示，表中分别给出了不同测试电路在算法运行前和运行后在 late 和 early 模式下的 WNS 与 TNS。slack_improv 为时序评价公式计算出的时序质量提升分数（在实验设定参数下，满分为 1800），run time 是程序运行时间。

表 1 ICCAD 2015 测试模板参数信息

Tab. 1 Information of benchmarks in ICCAD 2015

电 路	总单元数	可移动单元数	固定单元数	线网数	面积/μm^2	行数	Utility/%
Superblue1	1 216 315	1 159 346	56 969	1 215 710	1.07E+14	1829	51.0
Superblue3	1 219 842	1 160 765	59 077	1 224 979	1.22E+14	1840	46.3
Superblue4	802 305	756 979	45 326	802 513	7.11E+13	1840	59.4
Superblue7	1 938 219	1 865 884	72 335	1 933 945	1.31E+14	3163	57.9
Superblue10	1 888 389	1 786 523	101 866	1 898 119	1.94E+14	3437	46.1
Superblue16	986 037	981 140	4897	999 902	6.89E+13	1788	61.1
Superblue18	772 094	744 947	27 147	771 542	5.55E+13	1788	48.5

表 2 局部时序优化后的 benchmarks 电路布局结果

Tab. 2 Results after timing driven placement

电路		early WNS	early TNS	late WNS	late TNS	slack_improv	run time/s
Superblue1	布局前	−9.344 02	−317.437	−4976.58	−459 744	45.4	99.21
	布局后	−9.240 49	−328.581	−4839.61	−443 628		
Superblue3	布局前	−78.358 8	−1458.78	−10 145.9	−1.502E+6	144.4	97.70
	布局后	−72.017 6	−1425.61	−9854.6	−1.455E+6		
Superblue4	布局前	−12.552 1	−519.386	−6220.67	−3.476E+6	153.6	70.96
	布局后	−11.940 8	−494.99	6220.67	−3.354E+6		
Superblue7	布局前	−7.645 67	−1985.85	−15 216.3	−1.857E+6	48.9	199.45
	布局后	−16.737 2	−1892.06	−14 516.3	−1.806E+6		
Superblue10	布局前	−8.619 71	−620.952	−16 486.9	−3.315E+7	143.6	223.37
	布局后	−24.330 8	−604.952	−16 128.8	−3.198E+7		
Superblue16	布局前	−10.653 6	−113.749	−4581.07	−7.760E+6	348.7	91.27
	布局后	−9.768 3	−110.367	−4533.2	−7.590E+6		
Superblue18	布局前	−19.005 4	−282.999	−4550.7	−1.034E+6	353.8	81.14
	布局后	−14.027 5	−278.183	−5269.2	−9.986E+5		

注：$w_{TNS}=2$、$w_{WNS}=1$、$w_{late}=5$、$w_{early}=1$，$disp1=200$ μm。

由表 2 可以看出，相对于原始的全局布局结果，局部时序优化布局结果在 early 模式和 late 模式下总体松弛度都有一定程度的降低。本文时序优化方法的局部布局算法有效地优化了原始的布局电路，将时序质量提高 45 ~ 350 min。原因是本文优化算法每进行一次局部优化布局，都要利用 UI-Timer 分析时序，再根据时序分析结果进行一次粗粒度优化将标准单元放置到优化区域，然后根据优化结果再进行一次更加合理的细粒度时序优化，在不影响整体布局的情况下，有效地对局部区域进行时序优化，从而优化了整体时序质量。此外，实验结果还表明，虽然最差路径时序情况优化后可能会变差，但整体时序质量的情况会改善，如 Superblue7、Superblue10、Superblue18 等。

4 结论

本文在 ICCAD 2015 竞赛的基础上提出一种

时序驱动的局部时序优化布局算法。采用 ICCAD 2015 测试模板作为测试电路,验证时序优化方法的合理性和有效性。实验表明,在原始的布局电路设计上使用本文时序优化算法后,布局器能够将可移动单元移动到位置更好的区域,从而得到优化的时序布局结果。

在下一步的工作中,将对本文的方法进行进一步优化,表 3 给出了 ICCAD 2015 竞赛前五名的实验结果[18],对比发现,本文的实验结果还有较大改进空间。

表 3　ICCAD 2015 竞赛前五名的布局结果

Tab. 3　Results of top 5 team in ICCAD 2015 contest

电路	slack_improv					run time/s				
	Team A	Team B	Team C	Team D	Team E	Team A	Team B	Team C	Team D	Team E
Superblue1	77.53	346.64	0	163.68	185.79	1490.98	1917.67	43 201.97	2420.19	21 199.88
Superblue3	128.3	551.74	403.63	427.69	167.84	1387.44	1619.07	43 201.36	2143.66	35 083.48
Superblue4	193.39	507.31	0	208.53	197.5	1463.37	1117.46	43 201.83	1066.13	32 645.49
Superblue7	19.97	200.72	129.62	38.6	22.72	2715.72	3186.75	43 201.92	2183.24	39 277.98
Superblue10	235.95	181.33	161.75	231.83	276.89	3330.59	2245.89	43 202.16	2920.63	40 224.32
Superblue16	293.42	894.76	558.78	394.11	423.91	1111.81	1345.16	43 162.43	1440.52	24 323.32
Superblue18	194.21	613.07	484.2	355.11	315.65	955.84	951.87	40 550.36	726.42	10 198.86
Average	155.41	434.39	248.16	258.62	210.09	1720.78	1738.00	42 852.83	1756.10	27 565.58

注:$w_{\mathrm{TNS}}=2$、$w_{\mathrm{WNS}}=1$、$w_{\mathrm{late}}=5$、$w_{\mathrm{early}}=1$,$disp1=200$ μm。

参考文献(References)

[1] 郭阳, 李思昆, 杨强. 一种性能驱动的时序规划方法[J]. 国防科技大学学报, 2001, 23(2): 71 – 74.
GUO Yang, LI Sikun, YANG Qiang. An algorithm for performance-driven timing-planning[J]. Journal of National University of Defense Technology, 2001, 23(2): 71 – 74. (in Chinese)

[2] Kim M C, Hu J, Viswanathan N, et al. ICCAD – 2014 CAD contest in incremental timing-driven placement and benchmark suite [C]//Proceedings of IEEE/ACM International Conference on Computer-Aide, 2014: 361 – 366.

[3] Kahng A B, Lienig J, Markov I L, et al. VLSI physical design: from graph partitioning to timing closure [M]. Dordrecht, NLD: Springer Publishing Company, Incorporated, 2011.

[4] Agashiwala N, Upadhyay S P, Bazargan K. t-QuadPlace: timing driven quadratic placement using quadrisection partitioning for FPGAs (abstact only) [C]//Proceedings of ACM/Sigda International Symposium on Field-Programmable Gate Arrays, 2016: 284.

[5] Brenner U. VLSI legalization with minimum perturbation by iterative augmentation [C]//Proceedings of Conference on Design, Automation and Test in Europe, EDA Consortium, 2012: 1385 – 1390.

[6] Dutt S, Ren H X. Discretized network flow techniques for timing and wire-length driven incremental placement with white-space satisfaction[J]. IEEE Transactions on Very Large Scale Integration Systems, 2011, 19(7): 1277 – 1290.

[7] Flach G, Fogaça M, Monteiro J, et al. Drive strength aware cell movement techniques for timing driven placement[C]// Proceedings of the 2016 on International Symposium on Physical Design, 2016: 73 – 80.

[8] Ren H X, Pan D Z, Kung D S. Sensitivity guided net weighting for placement-driven synthesis [J]. IEEE Transactions on Computer-Aided Design of INTE, 2005, 24(5): 711 – 721.

[9] Guth C, Livramento V, Netto R, et al. Timing-driven placement based on dynamic net-weighting for efficient slack histogram compression [C]//Proceedings of the 2015 Symposium on International Symposium on Physical Design, 2015: 141 – 148.

[10] Viswanathan N, Huang S H, Lin R B, et al. Overview of the 2015 CAD contest at ICCAD [C]//Proceedings of IEEE/ACM International Conference on Computer-Aided Design, 2015: 910 – 911.

[11] Ajami A H, Pedram M. Post-layout timing-driven cell placement using an accurate net length model with movable Steiner points [C]//Design Automation Conference, Proceedings of the ASP-DAC 2001, Asia and South Pacific, 2001: 595 – 600.

[12] Choi W, Bazargan K. Incremental placement for timing optimization[C]//Proceedings of International Conference on Computer Aided Design, 2003: 463 – 466.

[13] Jung J, Nam G J, Reddy L, et al. OWARU: free space-aware timing-driven incremental placement[C]//Proceedings of the 35th International Conference, 2016.

[14] Marquardt A, Betz V, Rose J. Timing-driven placement for FPGAs [C]// Proceedings of the ACM/SIGDA eighth International symposium on field programmable gate arrays, 2000: 203 – 213.

[15] Viswanathan N, Nam G J, Roy J A, et al. ITOP: integrating timing optimization within placement [C]//Proceedings of International Symposium on Physical Design, 2010: 83 – 90.

[16] Huang C C, Liu Y C, Lu Y S, et al. Timing-driven cell placement optimization for early slack histogram compression[C]// Proceedings of the 53rd Annual Design Automation Conference, 2016: 81.

[17] Elmore W C. The transient response of damped linear networks with particular regard to wideband amplifiers[J]. Journal of Applied Physics, 1948, 19(1): 55 – 63.

[18] Kim M C, Jin H. IEEE CEDA / Taiwan MOE, ICCAD 2015 Contest [EB/OL]. [2016 – 10 – 01]. http://cad-contest.el.cycu.edu.tw/problem_C/default.html.