FPGA最小延时工艺映射理论及算法*

彭宇行 陈福接

(长沙工学院计算机研究所 长沙 410073)

摘要 本文以动态规划和网络流理论为基础,提出一种新的求解基于LUT结构的FPGA工艺映射算法,并证明了它能获得最小延时目标电路,算法计算复杂度低.

关键词 电子CAD,FPGA,工艺映射,延时分析,算法.

自从1985年美国Xilinx公司研制的基于LUT结构的FPGA芯片问世以来[1],以其设计周期短、制造代价低等特点赢得广大用户的欢迎.随之,面向FPGA的CAD技术,特别是工艺映射技术,得到广泛的研究.

传统的工艺映射技术是建立在标准单元库的基础上,它可描述如下:给定一个单元库和一个表示组合电路的有向无环图(DAG),用库单元构造目标电路来实现DAG的逻辑,并使电路面积最小或延时最优等.求解过程采用覆盖技术或动态规划等方法.[2,3]与其类似,现有的FPGA工艺映射主要目标分为以下几类:减少芯片中所使用的CLB个数,减小FPGA芯片延时,增大可布线性[4~8]等.然而,在基于LUT结构的FPGA芯片中,一个K入端的基本可编程块(CLB)能构造K个变量以内的任何布尔函数.因此,按照传统的方法求解FPGA工艺映射是行不通的.从图论的角度看问题,求DAG中某节点v_i的匹配,即是求分离v_i以及与v_i有关的原始输入节点的一个节点割集.由此观点出发,FlowMap[5]将DAG中的每个节点变为由一条容量为1的边相连的2个节点,从而将求节点割集化为求割集的问题,并运用网络最大流算法,求解单位延时模型下的延时最小化工艺映射.在一般延时模型中,为使割集最大延时弧最小,它先粗略估计延时上下界,然后在界内用二分法求解.[6]从目前的理论结果及实验数据来看,FlowMap是最成功的.

然而,FlowMap存在以下问题:①将DAG变化后,节点数增加一倍,计算量增大;②求解割集次数太多,当节点数为n时,需求解$\log n$次.本文给出一般延时模型下求解最短延时FPGA工艺映射算法.它首先根据动态规划的思想,证明了在求得DAG中每个节点的最短延时匹配后,通过从原始输入到原始输出的选择,可求得最短延时目标电路.为求节点的最短延时匹配,我们依据网络流理论,先直接求解DAG最小容量节点割集,然后逐步求优,获

* 本文研究得到"8.5同构型多处理机系统结构"经费资助.作者彭宇行,1963年生,讲师,主要研究领域为高性能机系统结构,VLSI设计,CAD算法.陈福接,1935年生,教授,主要研究领域为高性能计算机系统结构,VLSI设计,多媒体技术.

本文通讯联系人:彭宇行,长沙 410073,长沙工学院计算机研究所

本文1995-09-05收到修改稿

得延时最短节点割集.

1 延时最小化理论

1.1 延时模型

M 为目标电路的一个 CLB,它匹配 DAG 中节点 v. 设信号在 CLB 内部的传输延时为 D_0,Net_v 为 v 的扇出网络,网络延时为 $F(Net_v)$,$M_k,k=1,2,\ldots,i$, 为 M 的扇入单元,若 M_k 输出端的到达时间值为 $At(M_k)$,则 M 的到达时间计算如下[9]:

$$A_t(M)=\max\{A_t(M_k)\,|\,k=1,2,\ldots,i\}+D_0+F(Net_v)$$

不妨用 $D(v)$表示 $D_0+F(Net_v)$.

1.2 最小化算法

定义 1. 设 PI 为 DAG 的一个原始输入端,v 为节点,M 为 v 的一个匹配单元,则 v 及 M 的最小传输延时 $Lab(v)$,$Lab(M)$定义如下:

(1)$Lab(PI)=A_t(PI)=D(PI)$;

(2)设 $v_k,k=1,2,\ldots,i$,为 M 的扇入节点,则 $Lab(M)=\max\{Lab(v_k)\,|\,k=1,2,\ldots,i\}+D(v)$;

(3)若 v 有 j 个匹配单元 $M_1,M_2,\ldots,M_j$,则 $Lab(v)=\min\{Lab(M_k)\,|\,k=1,2,\ldots,i\}$称 $Lab(v)$,$Lab(M)$分别为 v,M 的标记.

引理 1. 若 M 为目标电路的组成单元,则 $A_t(PI)\geqslant Lab(M)\geqslant Lab(v)$

证明:用归纳法证明.

(1)$A_t(PI)=Lab(PI)=D(PI)$

(2)若 $M_k,k=1,2,\ldots,i$,为目标电路中 M 的扇入单元,且 M_k 匹配节点 v_k,假设 $A_t(M_k)\geqslant Lab(M_k)$,则

$$\begin{aligned}A_t(M)&=\max\{A_t(M_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&\geqslant\max\{Lab(M_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&\geqslant\max\{Lab(v_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&=Lab(M)\\&\geqslant Lab(v)\end{aligned}$$

定义 2. 若 M 是 v 的匹配单元,且 $Lab(M)=Lab(v)$,称 M 为 v 的最佳匹配单元.

引理 2. 若目标电路中的每一组成单元都是其匹配节点的最佳匹配单元,则对电路中的任一单元 M 及其所匹配的节点 v 来说,$A_t(M)=Lab(v)$.

证明:用归纳法证明.

(1)$A_t(PI)=Lab(PI)=D(PI)$

(2)若 $M_k,k=1,2,\ldots,i$,为目标电路中 M 的扇入单元,且 M_k 匹配节点 v_k,假设 $A_t(M_k)=Lab(M_k)$,则

$$\begin{aligned}A_t(M)&=\max\{A_t(M_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&=\max\{Lab(M_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&=\max\{Lab(v_k)\,|\,k=1,2,\ldots,i\}+D(v)\\&=Lab(M)\\&=Lab(v)\end{aligned}$$

算法 1. 基于 LUT 结构的 FPGA 工艺映射过程：

(1)从原始输入端开始，从下至上逐个求解 DAG 中每个节点的最佳匹配单元，直至原始输出端.

(2)从原始输出端开始，选择其最佳匹配；然后选择每个被选匹配单元的扇入节点的最佳匹配，直至原始输入端.

定理 1. 算法 1 能获得基于 LUT 结构的 FPGA 最小延时工艺映射目标电路.

证明：由引理 1，对任一原始输出端 PO，$Lab(PO)$是其最小到达时间；由引理 2，在目标电路中 $A_t(PO)=Lab(PO)$.

1.3 节点的最佳匹配单元

由定理 1，算法 1 的关键在于求 DAG 中每个节点的最佳匹配.

定义 3. 在 $DAG=(V,A)$中，设 $v_t\in V$，$G_t=(V_t,A_t)$定义如下：

(1)$v_t\in V_t$；

(2)若 $v_j\in V_t$，$(v_i,v_j)\in A$，则 $v_i\in V_t$，$(v_i,v_j)\in A_t$；

(3)若 $v_j\in V_t$，v_j 为原始输入端，则构造一个新节点 v_s，$v_s\in V_t\backslash V$，$(v_s,v_j)\in A_t$.

设 CLB 的输入端数为 K，则求节点 v_t 的一个匹配单元，即在 G_t 中求一个分离 v_s 与 v_t 的节点个数不多于 K 的节点割集；而求 v_t 的最佳匹配单元，即寻找一个延时最小的节点割集. 我们以网络流理论为基础，求解 v_t 的最佳匹配.

2 网络最大流与最小容量节点割集

2.1 网络与流

定义 4. 在有向无环图 $D=(V,A)$中，$v_s\in V$ 称为发点，$v_t\in V$ 称为收点，$V\backslash\{v_s,v_t\}$中任一节点 v_i 对应一个数 $I(v_i)=1$ 称为 v_i 的容量. 称 D 为节点单位容量网，记作 $D_I=(V,A,I)$.

定义 5. D_I 上的流是定义在弧集上的函数 $f=\{f(v_i,v_j)\mid(v_i,v_j)\in A,f(v_i,v_j)\geqslant 0\}$，称 $f(v_i,v_j)$为(v_i,v_j)上的流量.

定义 6. 满足下述条件的流称为可行流：

(1)容量限制条件：$\Sigma_{(v_i,v_j)\in A}f(v_i,v_j)\leqslant 1$，$v_i\in V\backslash\{v_s,v_t\}$；

(2)平衡条件：$\Sigma_{(v_i,v_j)\in A}f(v_i,v_j)-\Sigma_{(v_j,v_i)\in A}f(v_j,v_i)=0$，$v_i\in V\backslash\{v_s,v_t\}$

$\Sigma_{(v_s,v_j)\in A}f(v_s,v_j)=\Sigma_{(v_j,v_t)\in A}f(v_j,v_t)=v(f)$，称 $v(f)$为 f 的流量.

定义 7. D_I 中 $v(f)=0$ 的流称为零流，记为 f_0；$v(f)$达到最大的可行流称为最大流，记为 f_M.

2.2 增广链与节点割集

定义 8. 设 μ 是连接 v_s 和 v_t 的一条链，定义链的方向从 v_s 到 v_t. 若在 μ 中，弧(v_i,v_j)与链的方向一致，记 $g(v_j)=1$；若弧(v_i,v_j)与链的方向相反，记 $g(v_j)=0$.

定义 9. 在 $D_I=(V,A,I)$中，给定一可行流 f，对 $v_i\in V_{v_s,v_t}$，记 $h(v_i)=\Sigma_{(v_i,v_j)\in A}f(v_i,v_j)$. 设 μ 是从 v_s 到 v_t 的一条链，若 μ 中的任两相邻节点 v_i，v_j 满足下列条件，则称 μ 为关于 f 的一条增广链：

(1)若 $g(v_i)=1,h(v_i)=0$,则$(v_i,v_j)\in A$;

(2)若 $g(v_i)=1,h(v_i)=1$,则$(v_j,v_i)\in A,f(v_j,v_i)=1$;

(3)若 $g(v_i)=0,(v_i,v_j)\in A$,则 $h(v_i)=1,f(v_i,v_j)=0$;

(4)若 $g(v_i)=0,(v_j,v_i)\in A$,则 $h(v_i)=1,f(v_j,v_i)=1$.

定义 10. 给定 $D_I=(V,A,I)$,若点集 v 被分割为 2 个非空集合 $X,\sim X$,使得 $v_s\in X,v_t\in\sim X$,则称弧集$(X,\sim X)=\{(v_i,v_j)|v_s\in X,v_t\in\sim X,(v_i,v_j)\notin A\}$为分离 v_s 和 v_t 的割集;称节点集 $Y=\{v_i|(v_i,v_j)\in(X,\sim X)\}$为$(X,\sim X)$对应的节点割集,$Y$ 中节点个数$|Y|$为 Y 的容量.

定义 11. 在所有分割 v_s,v_t 的节点割集中,容量最小的一个节点割集称为最小容量节点割集,记为 Y_m.

定义 12. 设 f 是 D_I 的任一可行流,Y 为任一节点割集,称 $H(Y)=h(v_i)$为 Y 的流量.由网络流理论[10],我们可以得到下面的引理.

引理 3. $v(f)=H(Y)$.

引理 4. $v(f)\leqslant|Y|$.

引理 5. 若 $v(f)=|Y|$,则 $f=f_M,Y=Y_m$.

2.3　增广链的调整

定义 13. 设 f 是 D_I 的一个可行流,μ 是关于 f 的增广链,则对 μ 中的任一弧(v_i,v_j),下述过程称为对 μ 的调整过程:

(1)若 $g(v_j)=1$,令 $f^*(v_i,v_j)=1$;

(2)若 $g(v_j)=0$,令 $f^*(v_j,v_i)=0$;

(3)对 μ 以外的弧(v_i,v_j),令 $f^*(v_i,v_j)=f(v_i,v_j)$.

同样,由网络流理论[10],可得下述引理.

引理 6. 对 μ 进行调整后得一新流 f^*,则 f^* 仍为可行流,且 $v(f^*)=v(f)+1$.

定理 2. f 是 f_M 的充要条件是不存在关于 f 的增广链.

证明:若 $f=f_M$,假设存在关于 f 的增广链 μ,由引理 6,对 μ 进行调整后,得到可行流 f^*,且 $v(f^*)=v(f)+1$,与 f 是 f_M 矛盾.

若 D 中不存在关于 f 的增广链,我们用下面的方法来构造节点集 X:令 $v_s\in X,g(v_s)=1$;对 $v_i\in X$,分 3 种情况分别处理:

(1)若 $g(v_i)=1,h(v_i)=0$,则对所有$(v_i,v_j)\in A$,令 $v_j\in X,g(v_j)=1$;

(2)若 $g(v_i)=1,h(v_i)=1$,则对$(v_j,v_i)\in A$ 且 $f(v_j,v_i)=1$ 的节点 v_j,令 $v_j\in X,g(v_j)=0$;

(3)若 $g(v_i)=0$,则对所有$(v_i,v_j)\in A$,令 $v_j\in X,g(v_j)=1$;对$(v_j,v_i)\in A$ 且 $f(v_j,v_i)=1$的节点 v_j,令 $v_j\in X,g(v_j)=0$.

因为不存在关于 f 的增广链,故 $v_t\notin X$,于是得到割集$(X,\sim X)$及其对应的节点割集 Y,且对 $v_j\in Y$,有 $h(v_i)=1$. 由引理 3,$v(f)=H(Y)=|Y|$,由引理 5,f 为 f_M.

定理 3. $v(f_M)=|Y_m|$

证明:由定理 2,D_I 中不存在关于 f_M 的增广链,因而可构造一个节点割集 Y,使得 $v(f_M)=|Y|$. 由引理 5,$Y=Y_m$.

定理 4. 从 f_0 出发，经过 $|Y_m|$ 次调整过程可得 f_M.

证明：从 f_0 出发经一次调整过程后得 f_1. 则 $v(f_1)=v(f_0)+1=1$；从 f_1 出发经一次调整过程后得 f_2，则 $v(f_2)=v(f_1)+1=2$；……；若 f_{i-1}，f_i 分别为第 $i-1$ 次和第 i 次调整后所得的可行流，则 $v(f_i)=v(f_{i-1})+1=i$. 由于 $|Y_m|$ 为一个有限数，经过 $|Y_m|$ 次调整后可得可行流 $f_{|Y_m|}$，则 $v(f_{|Y_m|})=|Y_m|$，即 $f_{|Y_m|}$ 为最大流.

2.4 求最小容量节点割集

算法 2. 求 D_I 的 Y_m：

(1) $f\leftarrow f_0$；

(2)若不存在关于 f 的增广链，转 v；

(3)求一条关于 f 的增广链，调整后得 f^*；

(4) $f\leftarrow f^*$，转(2)；

(5)采用定理 2 证明中的构造法求 Y_m，结束.

设 D_I 的弧集 A 中包含 m 条弧，求一条增广链的计算复杂度为 $O(m)$，即每条弧最多扫描一次. 算法 2 中求增广链的次数为 $|Y_m|+1$，因此，求最小容量节点割集的计算复杂度仍为 $O(m)$.

3 最小标记 K 可行节点割集

3.1 D_L 及其变换网

定义 14. 在 D_I 中，若每个 $v_i\in V\backslash\{v_s,v_t\}$ 上赋一标记值 $Lab(v_i)$，则记 D_I 为 $D_L=(V,A,I,Lab)$. 设 D 为 D_L 的一个节点割集，称 $LAB(Y)=\max\{Lab(v_i)|v_i\in Y\}$ 为 Y 的标记.

定义 15. 在 D_L 中，$v_i\in V\backslash\{v_s,v_t\}$，若 $Lab(v_i)\geqslant LAB(Y_m)$，下述操作称为 v_i 的合并运算.

(1)从 D_L 中删除 v_i 以及与 v_i 相连的弧；

(2)将所有扇入到 v_i 的弧，连到 v_i 的所有扇出节点上.

引理 7. 在 D_L 中，对 v_i 进行合并运算后得 $D_L'=(V',A',I,Lab)$，若 Y' 为 D_L' 的任一节点割集，则 Y' 为 D_L 的节点割集.

证明：设 D_L 中 $(v_1,v_i),(v_2,v_i),\cdots,(v_m,v_i)$ 为 v_i 的扇入弧，$(v_i,u_1),(v_i,u_2),\cdots,(v_i,u_n)$ 为 v_i 的扇出弧，则在 D_L' 中，$(v_{k1},u_{k2})\in A,k_1=1,2,\ldots,m,k_2=1,2,\ldots,n$. 假设 Y' 不是 D_L 的节点割集，则在 D_L 中去掉 Y' 及其相连的弧后，仍有 v_s 到 v_t 的路径，不妨设为 $(v_s,\ldots,v_{k_1},v_i,u_{k_2},\ldots,v_t),k_1\in\{1,2,\ldots,m\},k_2\in\{1,2,\ldots,n\}$. 这样，在 D_L' 中去掉 Y' 及其相连的弧后，亦存在 v_s 到 v_t 的路径 $(v_s,\ldots,v_{k_1},u_{k_2},\ldots,v_t)$ 与 Y' 是 D_L' 的节点割集矛盾.

定义 16. 设 Y 为 D_L 的节点割集，给定一整数值 K，若 $|Y|\leqslant K$，称 Y 为 D_L 的 K 可行节点割集.

引理 8. D_L 中存在 K 可行节点割集的充要条件是 $v(f_M)\leqslant K$.

证明：若 $v(f_M)\leqslant K$，则 $|Y_m|=v(f_M)\leqslant K$，即 Y_m 为 D_L 的一个 K 可行节点割集；若 D_L 中存在 K 可行节点割集 Y，则 $|Y|\leqslant K$，即 $v(f_M)=|Y_m|\leqslant|Y|\leqslant K$.

定义 17. D_L 经下述变换后得到 $D_L^*=(V^*,A^*,I,Lab)$，称 D_L^* 为 D_L 的变换网：

(1)$V'\leftarrow V, A'\leftarrow A$;

(2)若 V' 中不存在节点 v_i,满足 $Lab(v_i)\leqslant LAB(Y_m)$,则

(2.1)$V^*\leftarrow V', A^*\leftarrow A'$;

(2.2)结束;

(3)选择一个满足 $Lab(v_i)\geqslant LAB(Y_m)$ 的节点 v_i,进行合并操作得 $D_L'=(V', A', I, Lab)$;

(4)转(2).

引理 9. 设 Y_m^* 为 D_L^* 的最小容量节点割集,若 $|Y_m^*|\leqslant K$,则 Y_m^* 为 D_L 的一个 K 行节点割集.

证明:在 D_L 中选择一个满足 $Lab(v_i)\geqslant LAB(Y_m)$ 的节点 v_i 进行合并运算后得 $D_L^{(1)}=(V^{(1)}, A^{(1)}, I, Lab)$,若 $Y_m^{(1)}$ 为 $D_L^{(1)}$ 的最小容量节点割集,由引理 7,$Y_m^{(1)}$ 为 D_L 的节点割集.

在 $D_L^{(1)}$ 中选择一个满足 $Lab(v_i)\geqslant LAB(Y_m)$ 的节点 v_i 进行合并运算后得 $D_L^{(2)}=(V^{(2)}, A^{(2)}, I, Lab)$,若 $Y_m^{(2)}$ 为 $D_L^{(2)}$ 的最小容量节点割集,由引理 7,$Y_m^{(2)}$ 为 $D_L^{(1)}$ 的节点割集,亦为 D_L 的节点割集.

由于 D_L 中满足 $Lab(v_i)\geqslant LAB(Y_m)$ 的节点个数有限,进行有限次合并运算后有 $D_L^{(j)}=D_L^*$. 由引理 7,Y_m^* 为 $D_L^{(j-1)}$ 的节点割集,Y_m^* 为 $D_L^{(j-2)}$ 的节点割集,……,Y_m^* 为 $D_L^{(1)}$ 的节点割集,Y_m^* 为 D_L 的节点割集. 又因 $|Y_m^*|\leqslant K$,因此它为 D_L 的 K 可行节点割集.

引理 10. 在 D_L 中选择任一节点 v_i 进行合并运算后得 $D_L'=(V', A', I, Lab)$,若 Y 为 D_L 中的任一节点割集,且 $v_i\overline{\in}Y$,则 Y 为 D_L' 的节点割集.

证明:若 Y 所对立的割集为 $(X, \sim X)$,当 $v_i\in X$ 时,令 $X'=X_{v_i}, \sim X'=\sim X$;当 $v_i\in \sim X$ 时,令 $X'=X, \sim X'=\sim X\backslash\{v_i\}$. 假设 Y 不为 D_L' 的节点割集,则存在 $v'\in \sim X'_Y, u'\in \sim X'$,使得 $(v', u')\in A'$. 由于 Y 为 D_L 的节点割集,则 $(v', u')\overline{\in}A$,即在 D_L 中 $(v', v_i)\in A$,$(v_i, u')\in A$,亦在 D_L 中存在路径 $(v_s, \ldots, v', v_i, u', \ldots, v_t)$,且该路径上的节点不包含在 Y 中,与 Y 是 D_L 的节点割集矛盾.

引理 11. 设 Y_m^* 为 D_L^* 的最小容量节点割集,若 $|Y_m^*|>K$,则 Y_m 为 D_L 最小标记 K 可行节点割集.

证明:假设 D_L 中存在节点割集 Y,满足 $|Y|\leqslant K$ 且 $LAB(Y)<LAB(Y_m)$,则因 D_L 中包含有限个 $Lab(v_i)\geqslant LAB(Y_m)$ 的节点 v_i,由引理 10,Y 亦为 D_L^* 的节点割集,即 $|Y_m^*|\leqslant|Y|\leqslant K$,矛盾.

3.2 求 D_L 的最小标记 K 可行节点割集

算法 3.

(1)$Y_m^*\leftarrow\Phi$;

(2)求 D_L 的最大流 f_M;

(3)若 $v(f_M)>K$,结束;

(4)求 Y_m 以及 $LAB(Y_m)$;

(5)求 D_L 的变换网 D_L^*;

(6)$Y_m^*\leftarrow Y_m$;

(7)$D_L \leftarrow D_L^*$;

(8)转(2).

定理 5. 算法 3 结束时,若 $Y_m^* = \Phi$,D_L 中无 K 可行节点割集;若 $Y_m^* \neq \Phi$,Y_m^* 即为 D_L 的最小标记 K 可行节点割集.

证明:由于 D_L 节点集 V 中的节点数有限,每求一次 D_L^*,至少从 V 中删去一个节点.因此,当 V 中节点数为 n 时,求 D_L^* 的次数不会超过 n,即算法能结束.

算法结束时,若 $Y_m^* = \Phi$,由引理 8,D_L 无 K 可行节点割集;若 $Y_m^* \neq \Phi$,由于 D_L 中节点数有限,且每个节点的标记值为有限数,由引理 9、11,Y_m^* 为 D_L 的最小标记 K 可行节点割集.

算法 4.

(1)求 D_L 的最小容量节点割集 Y_m;

(2)若 $|Y_m| > K$,结束;

(3)设 Y_m 对立的割集为 $(X_m, \sim X_m)$,将 $\sim X_m$ 合并为一个节点,形成新网,也记为 D_L;

(4)转 i.

3.3 复杂性分析

设 D_L 中包含 n 个节点,m 条弧,每求一个 K 可行节点割集,计算复杂度为 $O(m)$.又因每求一次 K 可行节点割集,V 中至少删去一个节点,所以算法 3 最多求 $n-K$ 次 K 可行节点割集,计算复杂度为 $O(mn)$.

根据网络流理论,每次所求的最小容量节点割集都是离 v_t 较近者.因此算法 4 中每求一次最小容量节点割集,$|Y_m|$ 至少增 1,即最多循环 K 次.一般来说,$K=5$,远小于 n.故其计算复杂度仍为 $O(m)$.我们也可以采用二分法求解,计算复杂度为 $O(m\log n)$.

从最坏的情况上看,二分法优于算法 3.但二分法求解,每次都要进行 $\log n$ 次求最小容量节点割集的运算.而对 *FPGA* 工艺映射来说,*DAG* 中各节点的标记值不大于其后继节点的标记值.因此,算法 4 在一般情况下能求得最小标记 K 可行节点割集.在特别的情况下,在离 v_t 较近的节点中,其标记值较小,则可以在算法 4 的基础上进行算法 3 的过程.当 $K=5$ 时,平均只求 2~3 次最小容量节点割集.

4 小 结

综上所述,求基于 LUT 结构的 FPGA 最小延时工艺映射过程如下:从原始输入节点开始,自下至上对每个节点 v_t 求 G_t,并在 G_t 中求 v_t 的最小标记 K 可行节点割集,即其最佳匹配单元,v_t 的标记值为 $Lab(v_t) = LAB(Y_m) + D(v_t)$.该过程直至计算出每个原始输出的最佳匹配单元.在此基础上,设置一队列 Q,先将所有原始输出端排入队列.当 Q 不空时,将队头节点的最佳匹配单元作为目标电路的组成单元,并将扇入到该匹配单元的非原始输入节点排入队尾.该过程直至 Q 为空.

我们用 C 语言实现了该算法,命名为 D_map,并用其运行了 ex1~ex4 等 4 种电路,它们分别为:双九位奇偶校验发生器、先行进位电路、快速六位加法器、类柱形乘法器电路.同时,我们也用 Xilinx 的 xdm 设计系统对这 4 个电路进行工艺映射.xdm 为目前我国所引进的最新的 FPGA 设计系统.相比之下,D_map 的设计结果优于 xdm 设计结果,实验数据如

表 1 所示.

表 1

电路	延时(ns)	
	xdm	D_map
ex1	30.2	25.3
ex2	34.3	30.2
ex3	34.3	30.2
ex4	54.6	44.8

参考文献

1 Carter W S. A user programmable reconfigurable logic array. In: Proc. IEEE CICC, 1986. 233～235.

2 Keudzer K. DAGON: technology binding and logic optimization by DAG matching. In: Proc. 24th ACM/IEEE DAC, 1987. 341～347.

3 Touati H J *et al*. Performance—oriented technology mapping. In: Proc. 6th MIT Conf. Advanced Research in VLSI, 1990. 79～97.

4 Francis R J *et al*. Chortle—erf: fast technology mapping for lookup table—based FPGA's. In: Proc. 28th ACM /IEEE DAC, 1991. 613～619.

5 Cong J, Ding Y. FlowMap: an optimal technology optimization in lookup—table based FPGA designs. IEEE Trans. on CAD of Integ. Cir. & Syst., 1994,13(1):1～12.

6 Cong J *et al*. An optimal performance—driven technology mapping algorithm for LUT based FPGA's under arbitrary net—delay models. In: Proc. Int. Conf. on CAD/Graphics'93, 1993. 599～603.

7 Woo N S. A heuristic method for FPGA technology mapping based on the edge visibility. In: Proc. 28th ACM /IEEE DAC, 1991. 248～251.

8 Schlag M *et al*. Routability—driven technology mapping for lookup table—based FPGS's. IEEE Trans. on CAD of Integ. Cir. & Syst., 1994,13(1):13～26.

9 Hitechcock R B *et al*. Timing analysis of computer hardware. IBM Journal of Research and Development, 1982,26 (1):100～105.

10 甘应爱等. 运筹学. 北京:清华大学出版社,1990.

THE TECHNOLOGY MAPPING THEORY ALGORITHM FOR MINIMAL DELAY IN LUT—BASED FPGA DESING

Peng Yuxing　Chen Fujie

(*Institute of Computer Science　Changsha Institute of Technology　Changsha*　410073)

Abstract　Based on the dynamic programming and the theorem of network flow, this paper presents a new technology mapping algorithm for delay optimization in LUT—based FPGA design in polynomial time. It is proved that the algorithm can minimize the delay in the target circuit. Its theoretical results are better.

Key words　ICCAD, FPGA, technology mapping, timing analysis, algorithm.