General Modeling and Technology-Mapping Technique for LUT-based FPGAs[1]

Amit Chowdhary and John P. Hayes
Advanced Computer Architecture Laboratory
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122
{*amitc,jhayes*}*@eecs.umich.edu*

Abstract

*We present a general approach to the FPGA technology mapping problem that applies to any logic block composed of lookup tables (LUTs) and can yield optimal solutions. The connections between LUTs of a logic block are modeled by virtual switches, which define a set of multiple-LUT blocks (MLBs) called an MLB-basis. We identify the MLB-bases for various commercial logic blocks. Given an MLB-basis, we formulate FPGA mapping as a mixed integer linear programming (MILP) problem to achieve both the generality and the optimality objectives. We solve the MILP models using a general-purpose MILP solver, and present the results of mapping some ISCAS-*85 *benchmark circuits with a variety of commercial FPGAs. Circuits of a few hundred gates can be mapped in reasonable time using the MILP approach directly. Larger circuits can be handled by partitioning them prior to technology mapping. We show that optimal or provably near-optimal solutions can be obtained for the large ISCAS-*85 *benchmark circuits using partitions defined by their high-level functions.*

1 Introduction

Current FPGAs fall into two main types based on their logic block structure: lookup table-based [1, 2, 11] and multiplexer-based. An m-input single-output *lookup table (LUT)* is a static RAM of 2^m bits which can programmed to implement any combinational logic function of at most m inputs. Figure 1 illustrates a few LUT-based logic blocks which serve as examples in this paper. The multiplexers appearing in some logic blocks are used only for selecting different LUT configurations; they are not used for mapping purposes. FPGA logic blocks can also contain flip-flops, which are bypassed when implementing combinational logic. In this paper, we will be concerned with mapping combinational circuits using logic blocks composed of one or more LUTs.

The *technology mapping problem* for FPGAs is to transform a gate-level logic circuit into an equivalent circuit of logic blocks. The goal is to minimize area, which is measured by the total number of logic blocks. Figure 2 shows a mapping of a small circuit using three 4-input LUTs. Existing mapping algorithms were designed specifically for the XC3000 shown in Fig. 1a [10]. This logic block can be configured either as a 5-input LUT, or as two 4-input LUTs where the total number of inputs is no more than 5. Most algorithms were designed for the single-LUT configuration of the XC3000. Some, such as Chortle-crf [8], also handle the two-LUT configuration, but only as a post-processing step. These algorithms employ heuristics which generate solutions quickly, but whose results can be far from optimal.

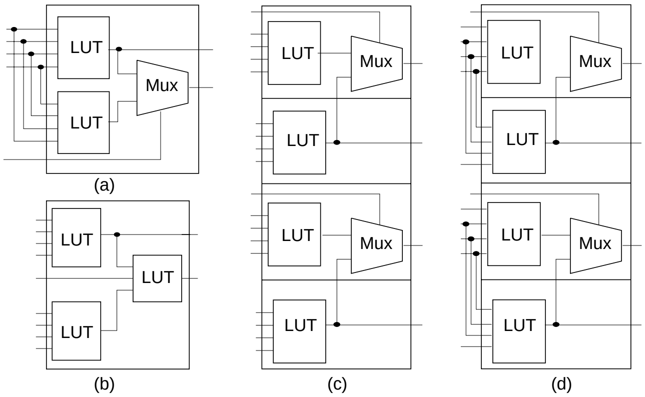

Figure 1: Combinational models of various FPGA logic blocks: (a)-(d) the LUT-based Xilinx XC3000, XC4000, XC5200, and AT&T ORCA, respectively.

We previously designed a technology mapping algorithm for single-LUT logic blocks [4], which is exact when the LUT size is monotone [5]. This paper extends the method of [4] to general non-monotone circuits and to multiple-LUT logic blocks. To achieve optimality, the FPGA mapping problem is formulated as a mixed integer linear program (MILP). The MILP problem is to minimize or maximize a linear objective function of a set of integer or real variables while satisfying a system of linear constraints. It can be stated in the matrix notation as: Minimize $\mathbf{cx} + \mathbf{dy}$ subject to $\mathbf{Ax} + \mathbf{Dy} \leq \mathbf{b}$, and $\mathbf{x} \geq \mathbf{0}$, $\mathbf{y} \geq \mathbf{0}$. Here $\mathbf{c}$ and $\mathbf{d}$ are cost vectors; $\mathbf{A}$ and $\mathbf{D}$ are constraint matrices; $\mathbf{x}$ is a vector of integer variables; and $\mathbf{y}$ is a vector of real variables.

We propose here a new and more general FPGA technology-mapping approach which is MILP-based and applies to any logic block that is a well-formed combinational

[1] This research was supported in part by the National Science Foundation under Grant No. MIP–9503463.

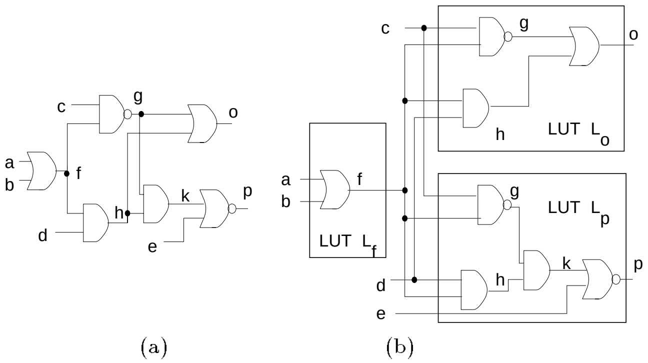

Figure 2: Technology mapping: (a) input circuit C; (b) optimal mapping using three 4-input LUTs.

circuit of LUTs. We introduce the concepts of virtual switch, multiple LUT block (MLB), and MLB-basis to represent the possible configurations of a logic block. This makes it possible to formulate technology mapping as an exact MILP optimization problem. The MILP formulation is extremely flexible, and can be solved in reasonable time using any standard MILP solver [3, 6]. We focus on area minimization, but the approach can be easily modified to optimize delay, or a combination of area and delay, by changing the objective and adding some constraints [4]. The method applies to most types of commercial logic blocks. Furthermore, it generates near-optimal mappings for very large circuits via partitioning the circuits and mapping each partition individually. Previous techniques are incapable of addressing such a wide range of logic blocks and mapping objectives.

2 Modeling Logic Blocks

Prior to technology mapping, the input logic circuit C is synthesized by a technology-independent logic minimization step performed using a logic synthesis package such as *mis* [7]. The gates in C are usually decomposed into smaller gates that can be mapped to the smallest component in the FPGA logic block. We model C by a directed graph $G(V, I, O, E)$ called its *circuit graph*, where V and E are nodes and edges that correspond to the gates and the interconnections of C, respectively. The nodes in I correspond to the primary inputs of C, and the nodes in O correspond to the gates at primary outputs of C. Note that O is a subset of V, while I and V are disjoint. Figure 3a shows the circuit graph of Fig. 2, where $V = \{f, g, h, k, o, p\}$, $I = \{a, b, c, d, e\}$ and $O = \{o, p\}$. The functionality of the nodes in G is ignored, since an m-input LUT can implement any function of at most m inputs.

An m-input LUT can map a connected subgraph $L(V_L, I_L, O_L, E_L)$ of a circuit graph G if and only if *(i)* $|I_L| \leq m$; *(ii)* $|O_L| = 1$; and *(iii)* for all $v \in V_L$ and $(u, v) \in E$, then $u \in V_L \cup I_L$. (A LUT is assumed to have a single output, unless stated otherwise.) We call such a subgraph an *L-graph* of G, and denote it by $L_{v_1,\ldots,v_n}$ if $O_L = \{v_1, \ldots, v_n\}$. Figure 3$b$ shows three L-graphs, L_f, L_o and L_p, of the circuit graph of Fig. 3a. Here $I_{L_f} = \{a, b\}$, $I_{L_o} = \{f, c, d\}$, $I_{L_p} = \{f, c, d, e\}$, $V_{L_f} = \{f\}$, $V_{L_o} = \{g, h, o\}$ and $V_{L_p} = \{g, h, k, p\}$. For a circuit graph G and a LUT L, technology mapping finds a set of L-graphs which are structurally equivalent to, or cover, G.

Definition 1 A *LUT cover* or *L-cover* of a circuit graph $G(V, I, O, E)$ by an m-input LUT L is a set $C_{L,G} = \{L_1, \ldots, L_q\}$ of L-graphs of G that satisfies the following conditions:

1. *Graph connectivity*: Every L-graph input is either a primary input of G or an output of some other L-graph. Similarly, every L-graph output is either a primary output of G or an input to some other L-graph.
2. *Node covering*: Every node belongs to at least one L-graph.
3. *Output uniqueness*: Every node is the output of at most one L-graph.

The area $A(C_{L,G})$ of $C_{L,G}$ is q, since every L-graph contributes one to the total area. □

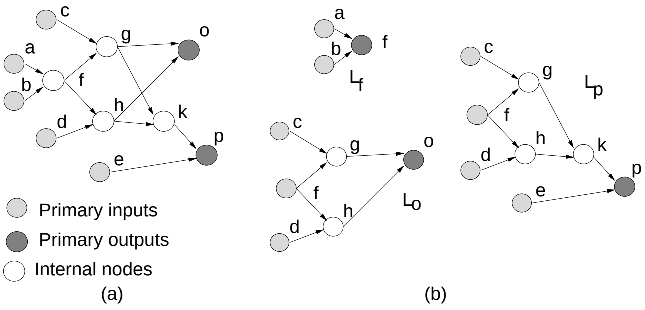

Figure 3: Technology mapping of the circuit of Fig. 2 using 4-input LUTs: (a) the circuit graph G; (b) an L-cover of G consisting of 3 L-graphs.

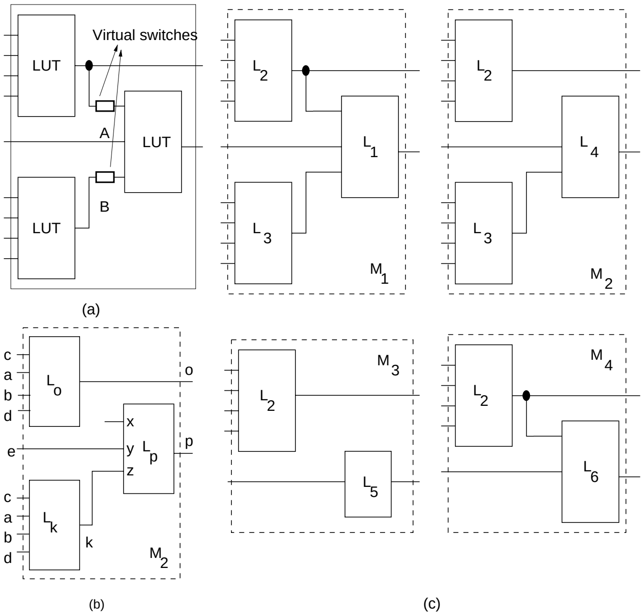

Figure 4: (a) XC4000 with its two virtual switches; (b) a single XC4000 mapping the graph of Fig. 3a; and (c) four MLBs obtained by programming the virtual switches. The MLB-basis of the XC4000 is $\{M_1, M_2\}$.

An area-optimal L-cover of the graph in Fig. 3 using 4-input LUTs is given by $\{L_f, L_o, L_p\}$. Nodes can be replicated in different L-graphs if this leads to a smaller L-cover. For example, the L-cover of Fig. 3b replicates nodes g and h in L_o and L_p. An L-cover corresponds directly to an FPGA implementation.

A LUT-based logic block can be used in various modes (circuit configurations) depending on its programmability

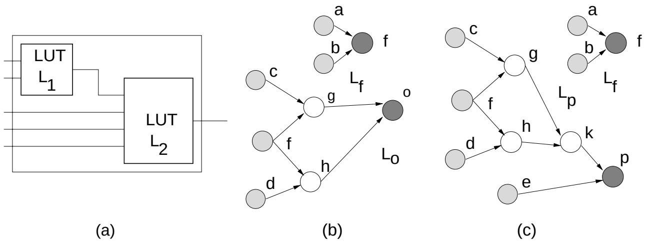

Figure 5: M-cover of the circuit graph of Fig. 3*a*: (a) MLB M; (b) M-graph M_o; (c) M-graph M_p.

[11, 2, 1]. Existing mapping techniques [10] were designed for one specific configuration, usually a single LUT, and so are not fully applicable to many commercial designs. There is thus a need for a general model to capture the various modes of a multiple LUT logic block that is a small circuit of LUTs. To address this need, we introduce the notion of a *virtual switch* which, in effect, simulates an external programmable connection to a LUT. Any connection between two LUTs or from a fanout stem to a LUT can be a virtual switch. For example, in the XC4000 logic block of Fig. 4*a*, A denotes a virtual switch between a fanout stem and a LUT, while B is virtual switch between two LUTs. A virtual switch is turned on/off by programming the internal switches of the LUT. The circuit graph of Fig. 3*a* can be mapped using a single XC4000 logic block as shown in Fig. 4*a*. Here virtual switch A is turned off by programming L_p to be independent of input x, which means that $p(0, y, z) = p(1, y, z)$ for all values of y and z.

There are 2^s ways of configuring the s virtual switches of a logic block, but s is usually small. Each such circuit is called a *multiple-LUT block* (MLB) M_i. Figure 4*b* shows the four MLBs of the XC4000 obtained by programming its virtual switches A and B; LUTs whose outputs are disconnected are deleted. Consider an MLB M with k LUTs of fanin $m_1, \ldots, m_k$. An *M-graph* of G by M is a set $(L_1, \ldots, L_k; I_M, O_M)$ of k L-graphs, where *(i)* $I_M \subseteq I_1 \cup \ldots \cup I_k$ is the set of inputs external to M; *(ii)* $O_M \subseteq O_1 \cup \ldots \cup O_k$ is the set of outputs; *(iii)* if L_i feeds L_j, then $O_i \subset I_j$ and $1 \leq |I_j - O_i| \leq m_j - 1$; and *(iv)* if L_i and L_j share d inputs, then $|I_i \cup I_j| \leq m_i + m_j - d$ and $1 \leq |I_i \cap I_j| \leq d$. Figures 5*b* and 5*c* show two M-graphs M_o and M_p for the MLB of Fig. 5*a*, where $M_o = \{L_f, L_o; \{a, b, c, d\}, \{o\}\}$ and $M_p = \{L_f, L_p; \{a, b, c, d, e\}, \{p\}\}$.

Two MLBs M_1 and M_2 are *independent* iff there exists at least one M-graph of M_1 that is not an M-graph of M_2, and vice-versa. For example, a 2-input LUT and a 3-input LUT are not independent. We model an FPGA logic block by its independent MLBs, which we call an *MLB-basis* $B = \{M_1, \ldots, M_p\}$, $p \leq 2^s$. If M_1 and M_2 are not independent and M_1 is subsumed by M_2, then M_1 is excluded from the MLB-basis. As is evident in Fig. 4*c*, M_3 and M_4 are subsumed by M_2 and M_1, respectively. Thus the MLB-basis of XC4000 is $\{M_1, M_2\}$.

Figure 6 shows the MLB-bases of some other logic blocks. The XC3000 logic block is used in two modes [11]; one is a 5-input LUT and the other is a pair of 4-input LUTs, these modes correspond to M_1 and M_2 of Fig. 6*a*, respectively. The two-output mode of the XC3000 has 6 virtual switches in its primary inputs. If the appropriate virtual switches are turned off, the two LUTs of M_2 can map independent functions, resulting in MLB M_3, also shown in Fig. 6*a*. Even though there are 6 virtual switches, only two of the resulting MLBs M_2 and M_3 are independent. Thus, the MLB basis for the XC3000 is $\{M_1, M_2, M_3\}$. The logic blocks in all current commercial FPGAs have very few LUTs, so their MLB-bases are small. XC4000E is identical to XC4000 except that the three LUTs are independent as shown in Fig. 6*b*. XC5200 has two identical half-blocks, where the basis of a half-block is shown in Fig. 6*c*.

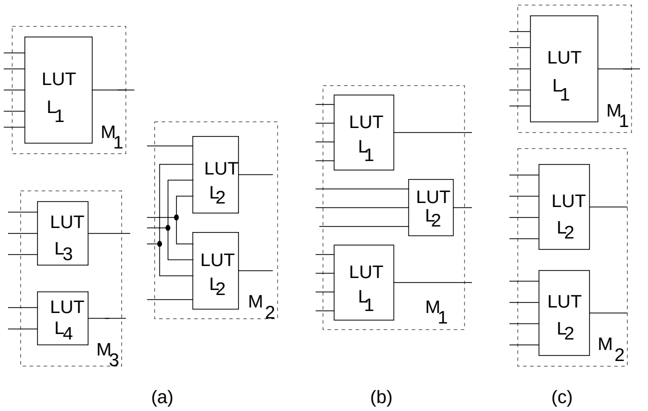

Figure 6: MLB-bases of other logic blocks: (a) XC3000, (b) XC4000E, and (c) XC5200.

Definition 2 An *MLB cover* or *M-cover* of $G(V, I, O, E)$ by a k-LUT MLB M is a set $C_{M,G} = \{M_1, \ldots, M_q\}$ of M-graphs of G, that satisfies the following conditions:
1. *Graph connectivity*: Every input of an M-graph is either a primary input of G or an output of another M-graph, and every output of an M-graph is either a primary output of G or an input of another M-graph.
2. *Node covering*: Every node belongs to at least one M-graph.
3. *Output uniqueness*: Every node is the output of at most one M-graph.
The area $A(C_{M,G})$ of $C_{M,G}$ is q. □

An area-optimal M-cover $\{M_o, M_p\}$ of the circuit graph of Fig. 3*a* by the MLB of Fig. 5*a* is shown in Figs. 5*b* and 5*c*. L-graphs can be replicated in different M-graphs of an M-cover. For example, the L-graph L_f is duplicated in the M-graphs M_o and M_p, as shown in Figs. 5*b* and 5*c*.

Next we define a B-cover as a generalization of M-cover where every M-graph corresponds to one of the MLB types in the basis.

Definition 3 An *MLB-basis-cover* or *B-cover* of $G(V, I, O, E)$ by an MLB-basis B with p MLBs is a set $C_{B,G} = \{M_{1,1}, \ldots, M_{1,q_1}, \ldots, M_{p,1} \ldots, M_{p,q_p}\}$ of M-graphs of G ($M_{i,j}$ is the jth M-graph of type i, $1 \leq i \leq p$) which satisfies the conditions from Definition 2 of an M-cover. The area $A(C_{B,G})$ of $C_{B,G}$ is $\sum_{i=1}^{p} q_i$. □

Finally, we state the general technology mapping problem for LUT-based FPGAs in a form suitable for MILP solution.

FPGA Technology Mapping Problem: Given an FPGA logic block modeled as an MLB-basis B, find a B-cover of G that optimizes area, where every MLB contributes unit area.

3 MILP Formulations

The MILP mapping technique for a general, multiple-LUT logic block is presented in stages, starting with the single-LUT case.

Two LUTs of an MLB M are of the same *type* i, and are denoted by L_i, if they have identical fanin and fanout. The

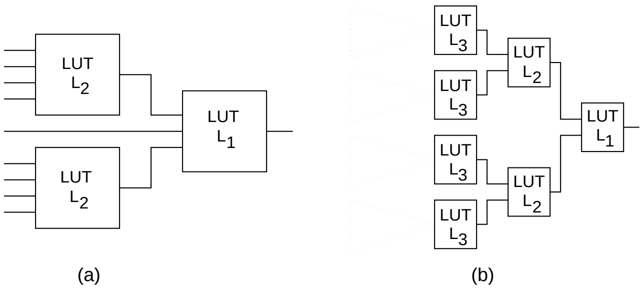

Figure 7: Tree MLBs: (a) a single-output version of the XC4000, and (b) a balanced tree with k levels.

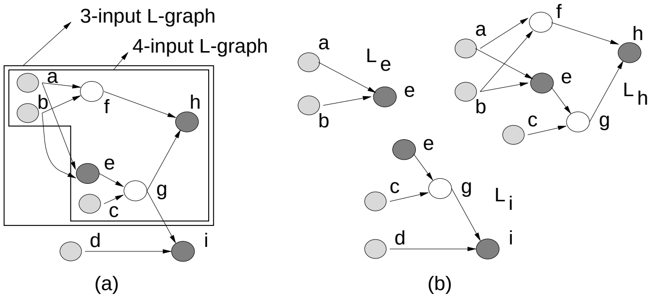

Figure 8: (a) A non-monotone graph G, (b) an optimal L-cover of G using 3-input LUTs obtained by solving MILP Model 1.

single-output version of the XC4000 shown in Fig. 7*a* has three LUTs, two of which are of type 2. For the balanced binary tree with $2^k - 1$ LUTs shown in Fig. 7*b*, the number of LUT types is equal to the number of levels. Similarly, two virtual switches are of the same type, if they lie between two LUTs of the same type. For example, in the MLB in Fig. 7*a*, the virtual switches between L_1 and each L_2 are of the same type, and are both on. Let k and s be the number of types of LUTs and virtual switches in M, respectively. A routing switch is associated with every primary input and output of M, resulting in a total of $s+1$ types of switches. An MLB can be characterized by a fanin matrix $[m_{i,j}]$ of size $(s+1) \times k$, where $m_{i,j}$ is the maximum number of switches of type i which can feed a LUT of type j. For the MLB of Fig. 7*a*, $F = \begin{bmatrix} 1 & 4 \\ 2 & 0 \end{bmatrix}$, where the switches of type 1 and 2 are the routing and virtual switches, respectively. For an MLB with one LUT of fanin m, F reduces to $[m]$. The MILP formulation of the mapping problem depends on the circuit graph G and the fanin matrix F of the logic block.

Single LUT: An MILP-based solution to the FPGA technology mapping problem is developed in [4] for the special case where the logic block is a single LUT. This model defines a binary *external* variable for every node of G. If *external*$[v]$ = 1 in the solution, then the node v is either a primary input or the output of an L-graph, otherwise it is mapped inside an L-graph. To ensure proper fanin for every L-graph, we define a *size* variable for every node v in G which counts the number of L-graphs or primary inputs feeding it. By definition, if v is an input of an L-graph, then *size*$[v] = 1$, otherwise *size*$[v]$ is obtained by summing the *size* variables of its fanin nodes.

For every node-pair (s,t) with multiple paths from s to t, called a *reconvergent node-pair* or RN-pair, we define a *reconvergence* variable to avoid multiple counting of *size*$[s]$ while evaluating *size*$[t]$. The circuit graph of Fig. 3*a* has two RN-pairs (f,o) and (f,p). Here *reconvergence*$[s,t]$ is set to *size*$[s]$ if the paths from s to t lie in the same LUT, and to 0 otherwise. Here we assume that there are two paths from s to t. In case of multiple paths from s to t, we decompose t at the preprocessing stage which does not affect optimality of the L-cover. The number of RN-pairs depends on the interconnection structure of G. A fanout-free circuit has no RN-pairs; at the other extreme, some circuits such as a grid have $O(|V|^2)$ RN-pairs, so the number of such pairs can be quadratic in $|V|$ (the number of RN-pairs was incorrectly stated in [4] to be $|E|$). We assign a *reconvergence* variable to an RN-pair only if the pair can be merged in a single LUT. Thus the number of *reconvergence* variables is reduced significantly by considering only the mergeable RN-pairs. For the ISCAS-85 benchmark circuits, for example, we show the number of mergeable RN-pairs to be less than 10% of the number of gates present. Our preprocessing procedure to find if an RN-pair (s,t) can be merged in an m-input LUT combines all the nodes between s and t in the subgraph of G rooted at t to form a composite node, and checks for a cut of size m in the modified subgraph. This procedure is similar to the step of finding a minimum-depth cut at every node of G in the Flowmap algorithm [5].

An m-input L-graph of G is monotone, iff it has no subgraph which is an L-graph with more than m inputs [5]. For example, the 3-input L-graph in Fig.8*a* is non-monotone, since it has a subgraph which is a 4-input L-graph. G is called *monotone* iff it has no non-monotone L-subgraph. We have found that the number of non-monotone L-graphs in the ISCAS benchmarks is less than 5% of the number of gates. The MILP model, described in [4], applies to all circuit graphs, but is guaranteed to be optimal only for monotone circuit graphs. Here, we extend that model to generate optimal L-cover for any non-monotone circuit graph.

If a node s is internal in an L-cover, then the subgraph rooted at s is implicitly replicated for all its fanout nodes. However, in the case of a non-monotone graph, if s is external, then L_s might have to be duplicated for a node t in its transitive fanout. For example, in the area-optimal L-cover of Fig.8*b* using 3-input LUTs, e is external, but L_e is duplicated for L_h. We call such a node-pair (e,h) a *duplication node-pair* or a DN-pair. We define a binary *duplication* variable for every DN-pair (s,t), where *duplication*$[s,t]$ is 1, only if both s and t are external (constraints (1)-(2)), and L_s has to be duplicated for L_t. If the *duplication* variable of Fig.8*a* is ignored, then the smallest L-cover has four LUTs, which is one more than optimal. Let *input-size*$[s]$ be the sum of the *size* variables of the fanin nodes of s, taking into account the *reconvergence* and *duplication* variables. (As defined earlier, *size*$[s]$ is equal to *input-size*$[s]$ if s is internal, otherwise it is set to 1.) For a general non-monotone G, *input-size*$[t]$ is calculated by the constraint set (3) given below.

$$\begin{aligned} duplication[s,t] &\leq external[s] && (1) \\ duplication[s,t] &\leq external[t] && (2) \end{aligned}$$

$$\begin{aligned} input\text{-}size[s] = \; & \textstyle\sum_{(u,t)\in E} size[u] \\ - & \sum_{(s,t)\in RN} reconvergence[s,t] \\ + & \sum_{(s,t)\in DN} duplication[s,t] \cdot (input\text{-}size[s]-1) \quad (3) \end{aligned}$$

Here, RN and DN stand for the set of RN-pairs and DN-pairs of G respectively. Note that the last term in constraint

(3) is non-linear, but it can be linearized using the fact that $input\text{-}size[s]$ is bounded by m.

The preprocessing procedure to find the DN-pairs of G uses a maxflow-mincut network flow technique similar to the Flowmap algorithm [5]. Like the RN-pairs, the number of DN-pairs is a very small fraction of the total number of nodes for the ISCAS benchmarks. The problem of finding an area-optimal L-cover can be formulated as follows.

MILP Model 1 Assign the *external* variables of G to $\{0,1\}$ to satisfy the constraints described in [4] along with the constraints (1)-(3). The resulting assignment defines an L-cover $C_{L,G}$ of G by L. An assignment with the minimum value of $\sum_{v \in V} external[v]$ is an area-optimal $C_{L,G}$. □

Tree MLB: We next consider an MLB M_t whose LUTs form a tree structure with no internal fanout. There is a one-to-one correspondence between the LUT types and the switch types of M_t, *i.e.* $k = s+1$, since every internal LUT has a single output which corresponds to a specific virtual switch, while the root LUT connects to LUTs in other logic blocks by routing switches.

The MILP formulation defines a binary $external_i[v]$ variable for every node v of G and every LUT type i. If $external_i[v] = 1$ in the solution, then the node v is the output of an L-graph which belongs to an M-graph of the M-cover corresponding to the solution. The root LUT is assumed to be of type 1. If $external_1[v] = 1$, then v is either a primary input of G or output of an M-graph. As in the single-LUT case, we define a $size_i[v]$ variable to count the number of switches of type i feeding v. The MILP constraints closely follow the conditions on an M-cover, its M-graphs, and their L-graphs.

Conditions on M-cover: $external_1[v] = 1$ for all $v \in I \cup O$ (node covering). The following constraint ensures that if a node v is external of type 1, then $size_1[v] = 1$ (graph connectivity).

$$external_i[v] \le size_i[v] \le external_i[v] + (max_{j=1}^{k} m_{i,j}) \cdot (1 - \sum_{j=1}^{k} external_j[v]) \qquad (4)$$

Conditions on M-graphs in the M-cover: For an M-graph M_i, $O_{M_i} = O_{i,1}$ since the root LUT is of type 1. If $m_{i,j}$ switches of type i feed a LUT of type j, then $m_{i,j}$ is used in calculating $size_i$ as described below by constraint (7) corresponding to condition *(iii)* of the M-graph definition.

Conditions on L-graphs of an M-graph: We define a variable $reconvergence_i[s,t]$ for every RN-pair (s,t) of G, which is set to $size_i[s]$ if s and t are in the same LUT, and 0 otherwise. Here we assume that there are at most two paths between any pair of nodes, similar to the assumption in MILP Model 1. If a node v in a path from s to t is external, then s and t are in different LUTs, which gives the following set of constraints.

$$0 \le reconvergence_i[s,t] \le (max_{j=1}^{k} m_{i,j}) \cdot (1 - \sum_{j=1}^{k} external_j[v]) \qquad (5)$$

$$reconvergence_i[s,t] \le size_i[s] \qquad (6)$$

Minimize $ext_1[f] + ext_1[g] + ext_1[h] + ext_1[k] + ext_1[o] + ext_1[p]$ **subject to :**

Boundary conditions:
$ext_1[v] = 1$, for all $v \in \{a,b,c,d,e,o,p\}$;

LUT fanin constraints: For all $v \in \{f,g,h,o,k,p\}$,
$ext_1[v] \le size_1[v] \le ext_1[v] + 4 \cdot (1 - ext_1[v] - ext_2[v])$;
$ext_2[v] \le size_2[v] \le ext_2[v] + 2 \cdot (1 - ext_1[v] - ext_2[v])$;

$size_1[a] + size_1[b] \le size_1[f] + 4 \cdot ext_2[f]$;
$size_2[a] + size_2[b] \le size_2[f] + 2 \cdot ext_1[f] - ext_2[f]$;
$size_1[c] + size_1[f] \le size_1[g] + 4 \cdot ext_2[g]$;
$size_2[c] + size_2[f] \le size_2[g] + ext_1[g] - ext_2[g]$;
$size_1[d] + size_1[f] \le size_1[h] + 4 \cdot ext_2[h]$;
$size_2[d] + size_2[f] \le size_2[h] + ext_1[h] - ext_2[h]$;
$size_1[g] + size_1[h] - recon_1[f,o] \le size_1[o] + 4 \cdot ext_2[o]$;
$size_2[g] + size_2[h] - recon_2[f,o] \le size_2[o] + 2 \cdot ext_1[o] - ext_2[o]$;
$size_1[g] + size_1[h] - recon_1[f,k] \le size_1[k] + 4 \cdot ext_2[k]$;
$size_2[g] + size_2[h] - recon_2[f,k] \le size_2[k] + 2 \cdot ext_1[k] - ext_2[k]$;
$size_1[k] + size_1[e] \le size_1[p] + 4 \cdot ext_2[p]$;
$size_2[k] + size_2[e] \le size_2[p] + 2 \cdot ext_1[p] - ext_2[p]$;

Reconvergence constraints: For $t \in \{o,k\}$ and $v \in \{g,h\}$,
$recon_1[f,t] \le 4 \cdot (1 - ext_1[v] - ext_2[v])$;
$recon_2[f,t] \le 2 \cdot (1 - ext_1[v] - ext_2[v])$;
$0 \le recon_i[f,t] \le size_i[f]$, for all $i \in \{1,2\}$;

Integrality constraints: For $i = 1..2$, $ext_i[v] \in \{0,1\}$, for all $v \in \{f,g,h,k\}$;

Figure 9: MILP Model 2 to find an area-optimal cover of the graph in Fig. 3a by the tree MLB of Fig. 7a. (Here *ext* and *recon* stand for *external* and *reconvergence* variables.)

The fanin constraint of an L-graph is ensured by the following constraint set for all $v \in V$ and for every LUT type i. Here, *input-size* is evaluated using constraint (3) defined for every LUT type i.

$$\sum_{(u,v) \in E} size_i[u] - \sum_{(w,v) \in RN} reconvergence_i[w,v] - \sum_{(w,v) \in DN} duplication[w,v] \cdot (input\text{-}size_i[w] - 1) \le size_i[v] - external_i[v] + \sum_{j=1}^{k} m_{i,j} \cdot external_j[v] \qquad (7)$$

If $external_i[v] = 1$, then $size_i[v] = 1$ from constraint (4), which implies that the fanin of the L-graph is no more than $m_{i,i}$. Similarly, if $external_j[v] = 1$ and $i \ne j$, then the fanin of L_v of type i is no more than $m_{i,j}$. If $external_j[v] = 0$ for all j, then $size_i[v]$ is bounded by the size of its fanin nodes, which implies that if $v \in V_L, (u,v) \in E$, then $u \in V_L \cup I_L$.

MILP Model 2 Assign the $external_i$, $1 \le i \le k$, variables of G to $\{0,1\}$ to satisfy the constraints of MILP Model 1 (defined for every LUT type i) along with the constraints (4)-(7). The resulting assignment defines an M-cover $C_{M_t,G}$ of G by the tree MLB M_t. An assignment with the minimum value of $\sum_{v \in V} external_1[v]$ is an area-optimal M-cover of G by M_t. □

The MILP model of an optimal cover of the graph in Fig. 3a by the tree MLB of Fig. 7a is given in Fig. 9. Note that the graph in Fig. 3a does not have any DN-pairs.

General MLB: We now extend the MILP formulation to apply to any MLB M which can have LUTs that fan out to one or more lines. The sub-block rooted at every LUT with multiple fanout is replicated to obtain a corresponding tree

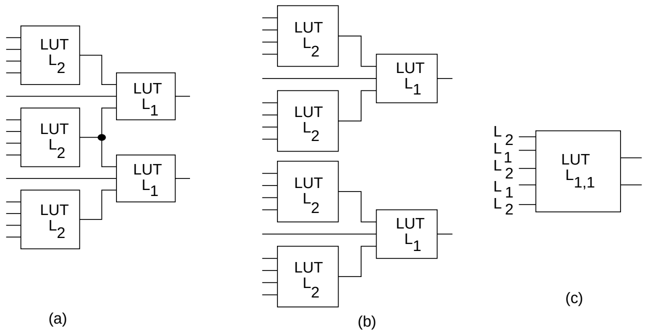

Figure 10: An MLB example: (a) logic block M with two outputs, (b) the tree network M_t rooted at the two outputs, and (c) the 2-output LUT $L_{1,1}$ in the set M_m.

MLB M_t. Consider a LUT L_i which fans out to r LUTs of types j_1, j_2, …, j_r, which need not be distinct. These r LUTs are combined to produce an r-output LUT $L_{(j_1,\ldots,j_r)}$. Let M_m be the set of all such multi-output LUTs of M. Thus, M is the union of M_t and M_m. For the 2-output logic block of Fig. 10a, M_t shown in Fig. 10b is similar to the tree MLB of Fig. 7a, while M_m consists of the 2-output LUT of type $(1,1)$ shown in Fig. 10c. The MILP model for M combines the constraints of M_t and every multi-output LUT of M_m.

We now describe the constraints for an r-output LUT of type $(j_1, j_2, \ldots, j_r)$, which corresponds to condition *(iv)* in the definition of M-graph. Let $m_{i,(j_1,\ldots,j_r)}$ be the number of switches of type i feeding the r-output LUT. A binary variable $external_{j_1,\ldots,j_r}$ is defined for every ordered group, $V_r = (v_1, \ldots, v_r)$, of r nodes of G which can be mapped to $L_{(j_1,\ldots,j_r)}$. Here $external_{j_1,\ldots,j_r}[V_r] = 1$, only if the nodes $v_1, \ldots, v_r$ are mapped to the LUTs of types j_1, …, j_r, respectively, which is ensured by the following set of constraints.

$$external_{j_q}[v_q] \leq external_{j_1,\ldots,j_r}[V_r] \text{ for } 1 \leq q \leq r \qquad (8)$$

We also define *reconvergence* variables for every pair consisting of a source node s and a sink node group T. The LUT fanin constraints for mapping a node group V_r to $L_{(j_1,\ldots,j_r)}$ are as follows.

$$\begin{aligned} & \sum_{q=1}^{r} size_i[v_q] - \sum_{(s,V_r)\in RN} reconvergence_i[s,V_r] \\ & - \sum_{(s,V_r)\in DN} duplication[s,V_r] \cdot (input\text{-}size_i[s] - 1) \\ \leq \;& m_{i,(j_1,\ldots,j_r)} \cdot external_{j_1,\ldots,j_r}[V_r] \\ + \;& (\sum_{q=1}^{r} m_{i,j_q}) \cdot (1 - external_{j_1,\ldots,j_r}[V_r]) \qquad (9) \end{aligned}$$

MILP Model 3 Assign the $external_i$ variables for the tree M_t and the $external_{j_1,\ldots,j_r}$ variables for every multi-output LUT of M_m to $\{0,1\}$ to satisfy all constraints of MILP Model 2 and constraints (8)-(9) for multiple-output LUTs. The resulting assignment is an M-cover $C_{M,G}$ of G by M. □

The MILP model for the MLB cover of the graph of Fig. 3a using the MLB in Fig. 10a, which has internal fanout, is obtained by adding the constraints for the LUT of type $(1,1)$ to the tree model of Fig. 9. The resulting MILP Model 3 for the MLB cover is presented in Fig. 11.

Minimize $ext_1[f] + ext_1[g] + ext_1[h] + ext_1[k] + ext_1[o]$
$+ ext_1[p] - ext_{1,1}[o,p] - ext_{1,1}[o,k] - ext_{1,1}[g,h]$
subject to
Constraints for LUT tree M_t: see Fig. 9;

LUT fanin constraints:
$size_1[o] + size_1[p] - recon_1[g,(o,p)] - recon_1[h,(o,p)]$
$\leq 2 \cdot ext_{1,1}[o,p] + 2 \cdot (1 - ext_{1,1}[o,p])$;
$size_2[o] + size_2[p] - recon_2[g,(o,p)] - recon_2[h,(o,p)]$
$\leq 3 \cdot ext_{1,1}[o,p] + 4 \cdot (1 - ext_{1,1}[o,p])$;
$size_1[o] + size_1[k] - recon_1[g,(o,k)] - recon_1[h,(o,k)]$
$\leq 2 \cdot ext_{1,1}[o,k] + 2 \cdot (1 - ext_{1,1}[o,k])$;
$size_2[o] + size_2[k] - recon_2[g,(o,k)] - recon_2[h,(o,k)]$
$\leq 3 \cdot ext_{1,1}[o,k] + 4 \cdot (1 - ext_{1,1}[o,k])$;
$size_1[g] + size_1[h] - recon_1[f,(g,h)]$
$\leq 2 \cdot ext_{1,1}[g,h] + 2 \cdot (1 - ext_{1,1}[g,h])$;
$size_2[g] + size_2[h] - recon_2[f,(g,h)]$
$\leq 3 \cdot ext_{1,1}[g,h] + 4 \cdot (1 - ext_{1,1}[g,h])$;

Reconvergence constraints: For all $s \in \{g,h\}$,
$recon_1[s,(o,p)] \leq 4 \cdot (1 - ext_1[k] - ext_2[k])$;
$recon_2[s,(o,p)] \leq 2 \cdot (1 - ext_1[k] - ext_2[k])$;
$0 \leq recon_i[s,(o,p)] \leq size_i[s]$, for all $i \in \{1,2\}$;
$0 \leq recon_i[s,(o,k)] \leq size_i[s]$, for all $i \in \{1,2\}$;
$0 \leq recon_i[f,(g,h)] \leq size_i[f]$, for all $i \in \{1,2\}$;

Integrality constraints: For $(u,v) \in \{(o,p),(o,k),(g,h)\}$,
$ext_{1,1}[u,v] \in \{0,1\}$;
$ext_1[u] \leq ext_{1,1}[u,v]$; $ext_1[v] \leq ext_{1,1}[u,v]$;

Figure 11: MILP Model 3 to find an optimal M-cover of the graph in Fig. 3a by the 2-output MLB of Fig. 10.

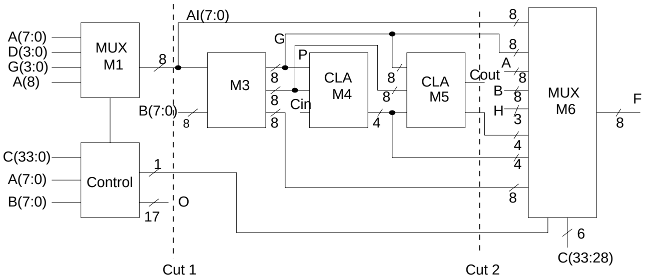

Figure 12: High-level model of c880 benchmark circuit [9] showing its two cuts.

MLB-Basis: The modeling technique from the previous section can be used to model any logic block by an MLB-basis B, which contains p independent MLBs. The MILP model for B will then consist of p different sets of *external*, *size* and *reconvergence* variables. Since the p MLBs of B are independent, the MILP Model for B is obtained by combining the constraints for the MILP Models 3 for the p MLBs of B.

4 Experimental Results

The preceding MILP models cover the logic blocks of Fig. 1, several of which are modeled as MLB-bases in Figs. 4b and 6. We have implemented the MILP models described in the previous section for general monotone circuits and a variety of MLB-bases. We have solved these models using various MILP solvers [6, 3], and found that *cplex* is the most efficient. Note that all these packages can generate optimal solutions, but their execution times vary considerably.

Prior to technology mapping, the circuit nodes are decomposed such that every node can be mapped to at least one LUT in the MLB, unlike our prior work [4] and that of a few others [10], where every node is decomposed into 2-input nodes. This has the advantage of reducing the time to solve the MILP model, since the time roughly depends on the number of binary variables in the formulation which in turn depends on $|V|$. As a preprocessing step, we have developed algorithms for finding the mergeable RN-pairs of G as well as the DN-pairs of G, both of which have a complexity of $O(m \cdot |V|^2 \cdot |E|)$.

We have mapped some representative ISCAS-85 benchmark circuits with the logic blocks of Fig. 1 using the MILP approach. The benchmarks c432, c499, c880 and c6288 have 196, 392, 347 and 2406 gates, respectively. These benchmarks are partitioned using their high-level models from [9]; the same partitions are also used in [4]. Most practical circuits have well-defined high-level models, which can produce suitable partitions. The cuts used for c880 benchmark are shown in Fig. 12. The number of cuts needed for a logic block depends on the number of LUTs and interconnections it contains. For example, more cuts are required to map a circuit by the XC3000, XC4000 and ORCA logic blocks than by the XC4000E and XC5200 logic blocks. The number of cuts to partition the selected benchmarks are shown in Table I along with the number of gates they contain. For every circuit and logic block pair, the MILP solution obtained is given in Table II. An important feature of our method is that it also gives tight lower bounds on the optimum area, which are shown in Table III. The lower bound is the smallest unexplored LP solution in the branch-and-bound search tree, since an unexplored LP solution can lead to an MILP solution.

Since prior work on FPGA technology mapping is restricted to the single-LUT case, we cannot compare the MILP results obtained here with prior results. Instead we compare the various logic blocks with one another based on the MILP results for the selected benchmarks. Using the MLB-basis of the XC3000 shown in Fig. 6a, area is reduced by about 20-50% compared to using only its single-LUT MLB M_1. The benchmarks require 15-20% fewer XC4000 logic blocks compared to the XC3000; this can be attributed to XC4000's larger fanin. The XC4000E gives a further 25-33% improvement for these benchmarks, since it has an additional output. Results for the XC5200 and ORCA logic blocks are similar, except that the ORCA has lower fanin since its two 4-input LUTs share three inputs. Our results for the XC5200 are better than for the ORCA by up to 30%. The lower bounds obtained are quite close for simple logic blocks such as the 5-input LUT, XC4000E and XC5200; this is not the case for other complex logic blocks such as XC3000 and XC4000. The difference between the MILP solutions and the lower bounds depends on the cut-size of the partition. The lower bounds are within one-third of the optimal solutions. Since the XC3000 and XC4000 require more cuts, we obtain loose lower bounds in these cases compared to other logic blocks.

5 Conclusions

We have designed and implemented a general FPGA technology mapping methodology that applies to any combinational LUT-based logic block. The method first models a logic block by a complete set of independent MLBs, and then constructs an MILP formulation for the mapping problem. We have generated MILP formulations for various commercial logic blocks. The MILP-based approach can optimally map circuits with hundreds of gates. Much larger circuits can be partitioned into sub-circuits of a few hundred gates to facilitate mapping.

The MILP method has been applied here to minimize area only. It can be extended to minimize any combination of area and delay, where the delay can be measured by the number of topological levels in the logic block cover. It is also useful for evaluating the relative mapping efficiencies of different FPGA families. Thus our methodology is quite general in that it can be applied to any current or projected LUT-based logic block to address a wide range of design objectives.

	5-input	Xilinx				AT&T
Cct.	LUT	3000	4000	4000E	5200	ORCA
c432	1	2	2	1	1	2
c499	1	2	1	2	2	2
c880	1	2	2	1	1	2
c6288	4	7	7	4	4	7

Table I: Number of cuts used for various logic blocks.

	5-input	Xilinx				AT&T
Cct.	LUT	3000	4000	4000E	5200	ORCA
c432	77	54	48	35	25	27
c499	66	48	41	31	19	24
c880	104	83	67	45	34	42
c6288	479	248	249	201	120	124

Table II: Number of logic blocks (area) obtained.

	5-input	Xilinx				AT&T
Cct.	LUT	3000	4000	4000E	5200	ORCA
c432	72	44	36	30	23	22
c499	61	44	35	27	16	22
c880	86	56	42	37	23	28
c6288	355	216	184	179	92	108

Table III: Lower bounds on area from MILP approach.

References

[1] Altera Inc. *The Altera FPGA Data Book*, Sunnyvale, Calif., 1993.

[2] AT&T Inc. *The AT&T FPGA Data Book*, Allentown, Pa., 1995.

[3] P. Barth. *Logic Based 0-1 Constraint Programming*. Kluwer, Boston, 1995.

[4] A. Chowdhary and J. P. Hayes. Technology mapping for field-programmable gate arrays using integer programming. *Proc. Int'l Conf. on CAD*, pp. 346–352, 1995.

[5] J. Cong and Y. Ding. FlowMap: An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Designs. *IEEE Trans. on CAD*, 13(1):1–11, Jan. 1994.

[6] CPLEX Optimization Inc. *CPLEX documentation*, 1990.

[7] E. Detjens *et al.* Technology mapping in MIS. *Proc. Int'l Conf. on CAD*, pp. 116–119, 1987.

[8] R. J. Francis, J. Rose, and Z. Vranesic. Chortle-crf: Fast technology mapping for lookup table based field programmable gate arrays. *Proc. 28th Design Automation Conf.*, pp. 613–619, 1991.

[9] M. C. Hansen and J. P. Hayes. High-level test generation using physically-induced faults. *Proc. of VLSI Test Symposium*, pp. 20–28, 1995.

[10] A. Sangiovanni-Vincentelli, A. El Gamal, and J. Rose. Synthesis methods for field programmable gate arrays. *Proc. of IEEE*, pp. 1057–1083, July 1993.

[11] Xilinx Inc. *The Programmable Logic Data Book*, Santa Clara, Calif., 1994.