Functional Decision Diagrams
for Technology Mapping to Lookup-Table FPGA

E. Schubert  U. Kebschull  W. Rosenstiel
Lehrstuhl für Technische Informatik
Universität Tübingen
Sand 13, 72076 Tübingen, Germany
Phone: ++49-7071-295482
Fax: ++49-7071-610399
EMail: <es|uk|rosen>@peanuts.informatik.uni-tuebingen.de

Abstract

This paper addresses the problem of technology mapping to lookup-table based field-programmable gate arrays. It uses a different approach than others for the decomposition of the network into feasible nodes, the Davio expansion. To perform this it uses an efficient representation of the decomposed nodes, Functional Decision Diagrams (FDD). The influence of FDD optimisation methods to the mapped solution is shown. Certain methods for the covering phase into lookup tables were discussed, which are targeted to area, delay and routability optimisation.

1 Introduction

In the ASIC market the use of FPGA increases, especially the use of reconfigurable lookup-table (LUT) based types. Since the user is able to program (and re-program) these circuits himself, the time-to-market can be extremely reduced and fast prototyping becomes possible. However, this results in the need of efficient CAD tools [19, 7]. The final phase of digital systems design for FPGA, the technology mapping phase, is not yet fully solved.

In recent years lots of research in methods for technology mapping has been done. The main task of these methods is to optimise the area [9, 13, 14, 15, 20, 23, 24, 25, 29], i.e. the number of CLBs required for the final solution. Also the performance of the final solution is extremely important, a fact that still limits the practical use of LUT based FPGA. Such methods for delay optimisation in technology mapping to LUT based FPGAs are described in [11, 10, 16, 26, 29, 30]. Since there are restricted resources for the wiring on the chip, routability, comes more and more into view of mapping tools [26, 4, 31].

Several different approaches have been established. Nearly all of them start from a minimised multilevel boolean network, obtained from MIS [6, 5] or others. Then the network is decomposed into feasible or sometimes even smaller nodes. In the covering phase the nodes are collapsed into lookup tables applying several heuristics, depending on the optimisation target.

The methods split into tree and DAG mapping approaches [14] taking also into account reconvergent paths in the network and doing replication of logic for area optimisation [15] or are targeted to delay optimisation [16]. Others start with a decomposition of the multilevel network into feasible nodes, therefore using kernel extraction techniques, cube packing, cofactoring (i.e. the Shannon expansion) or Roth-Karp disjoint decomposition [24, 25]. For delay optimisation a level numbering heuristic is used in the covering phase, which minimises the length of the critical path [26]. In [13] the covering takes account of a special feature of the Xilinx FPGA of 3000 series [34], that
CLBs can also implement two-output functions under special constraints. In [11] another method is used for the decomposition, the DMIG algorithm, which does not increase the length of the critical path during decomposition. This delay oriented property is further used in [10] to allow a smooth migration inside the area/delay tradeoff. For the area optimisation in the covering phase the concept of maximal fanout free cones is introduced there.

Reduced Ordered Binary Decision Diagrams (BDDs) are used in [23] and [9] to perform a functional decomposition of the nodes. Their basic idea is to cut the ROBDD level-wise into smaller OBDDs, representing feasible nodes, or to substitute levels of the BDDs using intermediate variables. In these approaches the two phases, decomposition and covering are no more separated.

An alternative representation to BDDs are the Functional Decision Diagrams (FDDs). They are a graph-based representation of the Reed-Muller Expansion (RME), a representation which is more efficient [3, 27] and can lead more easily to testable [28] solutions than the well known sum-of-product form. FDDs were introduced in [22, 32] for the use in multilevel logic synthesis. In [21] the full set of efficient manipulation algorithms for FDDs were given. In [1] an efficient implementation of an FDD-package was shown. There it could be shown that for special kinds of Boolean functions (e.g. arithmetic or symmetric functions) the size of FDDs is polynomial but the size of BDDs is exponential [2].

In this paper FDDs were used to represent the net nodes efficiently. This approach implies a different method for the decomposition of the multilevel network into feasible nodes: the application of the Davio expansion [12], an alternative to the well known Shannon expansion. Methods for the optimisation of FDDs were shown to optimise the mapped solution towards area, delay and routability.

This paper is organised as follows: Section 2 gives some basic definitions and the theoretical background of the Reed-Muller Expansion and Binary as well as Functional Decision Diagrams. In section 3 the properties of LUT based FPGA are described. After formulating the mapping problem, the FDD based mapping method is described in section 4. Section 5 gives results of a comparison with other methods for technology mapping to LUT based FPGAs. Finally section 6 gives concluding remarks and an outlook for future research directions.

2 Theoretical Background

In the following we use the basic definitions of [5] for multilevel networks, "Boolean networks", and network node representations. For two-level representations the well known sum-of-products form (SOP form) is used.

With the following shorthand of the cofactors: \( f_0 = f(x_i = 0) \); \( f_1 = f(x_i = 1) \); \( f_2 = f(x_i = 0) \oplus f(x_i = 1) \) the Shannon decomposition (or expansion) of a Boolean function \( f(x) \) in variable \( x_i \) is defined as:

\[
  f(x) = x_i \cdot f_1 + \bar{x}_i \cdot f_0, \text{ where } \bar{x} = (x_0, x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{n-1}),
\]

The Davio decomposition (or expansion) is a somehow modified Shannon decomposition, with the OR operation exchanged to EX-OR (this is possible, since the two terms are disjoint). It uses the equivalence of \( \bar{x} = x \oplus 1 \). Two Davio decomposition exist: one in positive variable \( x_i \) (eq. 1) and one in complemented variable \( \bar{x}_i \) (eq. 2):

\[
  f(x) = x_i \cdot f_2 \oplus f_0
  = \bar{x}_i \cdot f_2 \oplus f_1
\]

A function is called feasible, if it can be directly implemented by one single basic block of an LUT FPGA.

The level of a node of the multilevel network is recursively defined as the maximum of the levels of its fan-in nodes plus one. Primary inputs are defined having level 0.
2.1 Reed-Muller Expansion

The Reed-Muller expansion (RME) goes back to studies of Reed and Muller and to very early studies of Zhegalkin. In the RME every Boolean function can be written as an Ex-OR-polynomial:

\[
\begin{align*}
f(x_0, \ldots, x_{n-1}) &= a_0 \oplus a_1 \cdot x_0 \oplus a_2 \cdot x_1 \oplus a_3 \cdot x_0 \cdot x_1 \oplus \cdots \oplus a_{2^n-1} \cdot x_0 \cdot \cdots \cdot x_{n-1} \\
f(x_0, \ldots, x_{n-1}) &= \bigoplus_{t=0}^{2^n-1} a_t \cdot \pi_t
\end{align*}
\]

where

- \( \pi_t \) is the pi-term (the equivalent to the product-term of the SOP form)
- \( a_t \) is the pi-term coefficient
- \( \oplus \) is the Modulo-2 addition which is in the Boolean case equal to the Ex-OR-operation

This notation is called the zero polarity RME, since each variable is used only in its positive case, i.e. not complemented. The zero polarity RME is a canonical representation. We obtain the Generalised RME by dropping the constraint that each variable is used only in its positive case. Two kinds of Generalised RME exist:

- The fixed-polarity RME
  Each variable is used either positive or complemented (but not both). As there are \( n \) variables we have \( 2^n \) different RME, each with a different polarity value and different sets of coefficients. The fixed-polarity RME is also a canonical form.

- The mixed-polarity RME
  In this case a variable can be used in its positive and its complemented case at the same time, so the mixed-polarity RME is the more general but no more canonical representation.

The RME can be obtained by performing a Reed-Muller Transformation (RMT) of the SOP form. Each cube of the SOP form is directly transformed into one pi-term of the RME, as a prerequisite the cubes must be disjoint [21]. The RME does not directly represent the behaviour of a Boolean function like the SOP form. It rather represents the so-called spectral parts of the function. So the RME is suited to the functional domain, in opposite to the SOP form, which is suited to the operational domain [18]. The RME is conjectured to be a less complex two-level representation for some kinds of functions [3] than the SOP form (recently this conjecture was proven in [27]). It also can lead to more easily testable realizations for stuck-at faults [28]. The RME is not widely used due to the extra costs of EX-OR gates in CMOS technology, and since the behaviour cannot be obtained easily for manual design. But these drawbacks are overcome when the RME is used for Boolean function representation inside CAD tools.

2.2 Binary Decision Diagrams

For the complete definition and properties of Binary Decision Diagrams (BDDs) we refer to publications of Bryant [8]. In the sequel, if we talk about Decision Diagrams (BDD or FDD), we always mean the shared, reduced, ordered variant of it (SROBDD or SROFDD, resp.), but only use the shorthand.

In [8] Binary Decision Diagrams were defined using the Shannon decomposition in all the input variables of the Boolean function. Another view of BDD is, that they are the reduced tree, with the leaves of this tree as the function values (minterm coefficients). The reduction is done by

- eliminating nodes with isomorphic children, and
- combining isomorphic subtrees.
As a main drawback, the complexity of a BDD depends on the variable ordering: For one order the BDD can explode, for another the function can be represented [8]. The calculation of a good ordering is known to be NP complete, so heuristics were used instead of exact methods. A polarity transformation of the input variables (e.g. instead of using \(x_i, \overline{x}_i\) is used) has no influence on the complexity of the BDD. It only permutes the minterm coefficients in a special way without changing them. It results in swapping the 0- with the 1-successors of the nodes of this level without further reductions.

2.3 Functional Decision Diagrams

A Functional Decision Diagram (FDD) is a graph-based representation of the coefficients of the fixed-polarity RME. The pi-term coefficients of the two-level RME can be represented by a binary tree. For example, the binary tree of fig.1 represents the function \(f = d \oplus db \oplus cb \oplus dc \oplus dca \oplus dca\).

![Binary Tree and its Functional Decision Diagram](image)

By applying the same reduction methods as done for BDD, we obtain a graph-based representation for the two-level RME (fig. 1). We call this representation Functional Decision Diagrams, since this DAG bases on the RME, which is suited to the functional domain. So the FDD does not represent the behaviour of the function, but the spectral part the function consists of.

In the same way like FDD are a representation of a two-level RME, they also can be seen as a multilevel representation in a factored form [22]. This is due to the fact, that each node of an FDD has a sub-function \(S_i\), which depends on the sub-functions of its two successor nodes, \(S_{i+1}\) and \(S_{i-1}\) resp. These sub-functions were defined by the Davio expansion, they are the cofactors \(f_2\) and \(f_0\) (or \(f_1\), resp.). Since eliminated nodes are nodes having isomorphic successors, their sub-function only depends on one successor node. (This is in opposite to BDD, where eliminated nodes mean don't-care.) So the function of the FDD of fig. 1 can also be written by the factored form \(f = a(b \oplus 1)cd \oplus b(c(d \oplus 1) \oplus d) \oplus d\).

The computation of FDDs via reduced representations and not via matrix transformations solves the RMT problem and makes the RME (and so FDDs) applicable. The efficient transformation and manipulation algorithms for FDD were given in [21], there also the RMT problem is solved. The optimisation of these manipulation algorithms in [1] shows similar complexity as Bryant's algorithms for the BDD case. Additional complexity studies of [2] also show polynomial boundaries for FDD vs. exponential boundary for BDD for some kinds of Boolean functions, and vice versa.

2.4 Optimisation of FDD

In the same manner as BDD, the complexity of FDD depends on the variable ordering. A further degree of freedom is a polarity transformation. This results from the fact, that the coefficients of the Reed-Muller expansion change for different polarities of the input variables, and so the whole FDD is changed too. So the variable polarity (the choice between the two possible Davio expansions) is a further degree of freedom for the optimisation, which is not available in the BDD case. Generally three classes of optimisation methods exist [33]:

---

4
2.4.1 Domain specific variable ordering methods

These methods calculate a variable ordering by the properties of the coefficients of the DAG representation, so BDD methods are not suited for FDD optimisation. In [33] a heuristic for calculating an initial ordering by placing "most unate variables first" is described, which was adapted from [17] from the BDD case.

2.4.2 Domain independent variable ordering methods

These are ordering methods which base only on the topology of the DAG. Therefore they are the same for the optimisation of BDD and of FDD. In fig. 2 an example from [17] shows the exchangement of variables of level $z_i$ with those of level $z_{i+1}$ in BDD and in FDD case.

![Diagram showing variable exchange rules](image)

Figure 2: Exchange rules of two variables $z_i$ and $z_{i+1}$

This method can be performed very efficiently, since fast pointer techniques can be used, when implemented in C.

2.4.3 Polarity optimisation methods

By choosing between the two Davio expansions the FDD can be reduced in the number of nodes. A method therefore is also described in [33]. The example of fig. 3 shows the principle of this reduction.

![Diagram showing polarity influence](image)

Figure 3: Influence of variable polarity to FDD complexity

In general, a reduction of the FDD does not correspond to the minimisation of the corresponding two-level RME.

3 LUT FPGA Architecture

The general architecture of Xilinx FPGAs consists of a two-dimensional array of Configurable Logic Blocks (CLBs) with horizontal and vertical routing channels [7].

Each CLB consists of a $m$-input RAM. In Xilinx 2000 series the CLB has 4 inputs, in Xilinx 3000 series $m = 5$ and in the newest architecture, Xilinx 4000, the CLB is a two-level net of RAMs, where the first stage consists of two 4-input and the last stage consists of one 3-input RAM. Additional D-flip-flops (one in XC2000 and two in XC3000
and XC4000) make the implementation of sequential logic possible. This architecture allows the realization of complex gates in one single CLB, no matter which function should be realized. The only constraint is the number of inputs, the function depends on.

For routeing in XC2000 and XC3000 series three types of connections are available: direct interconnect, general purpose and long lines. In 4000 series the routeing resources are significantly different: Among direct interconnect and long lines, the general purpose wiring consists of an increased number of single- and double-length lines. For a more precise description of LUT architectures refer to [7, 34].

Due to the restricted routeing resources in most practical designs the yield of the number of CLBs is limited and the delay of an implementation is often not predictable. In some cases designs are even not routable, so routability must be taken into account during the synthesis.

4 Mapping of Functional Decision Diagrams

The general idea is as follows: Since technology mapping means the optimisation of the structure of a Boolean network for a target technology, representing the Boolean functions in functional domain may lead to better results than a representation of the behaviour of the network. This means, that the application of the Davio expansion for the task of decomposing the net into feasible nodes should lead to better results, especially for symmetric or arithmetic functions. FDDs were used to overcome the problem of [29], where the application of the Shannon expansion was limited to 10 levels due to complexity problems.

After formulating the mapping problem, an overview over the mapping will be given. Then each mapping step will be described in detail. Methods targeted to area, delay and routability optimisations will be discussed.

4.1 The Mapping Problem

The mapping to LUT based FPGA is complex due to the architecture of the CLBs. Up to now no optimal approach is available, which exploits all the degrees of freedom, f.e. of the Xilinx XC3000 CLBs. So either non-deterministic methods were applied, most of them base on simulated annealing, or heuristics were applied, leading to a restriction of the search space. In most cases three heuristic criteria were used: the chip area, which is modelled by the number of required CLBs, the delay that is modelled by the length of the critical path of the multilevel network, and finally the routability which is modelled by the number of connections between the CLBs.

The mapping problem must cover these three facts: In the case of this paper the area is heuristically optimised by minimising the initial network using multilevel logic synthesis and by optimising the net node representation, i.e. the FDD. The delay is heuristically optimised by reducing the number of levels of the obtained multilevel net. The routability is heuristically taken into account by not collapsing the network into too complex nodes (f.e. nodes with more than 4 inputs).

4.2 Overview

Starting point is a multilevel representation as a multilevel Boolean network. With phase assignment the multilevel network is optimised by area and delay. Then on each node of the multilevel network an FDD decomposition is performed as described below. For area optimisation the FDD is optimised using the methods of section 2.4. The delay is optimised further by reducing the number of levels of the multilevel network. Using the delay optimisation algorithm the covering of the decomposed network into LUT is done simultaneously. To keep in mind the routability of the final solution, the connection complexity of the nodes is reduced.
4.3 FDD Based Node Decomposition

The idea of the FDD based decomposition bases on the view of FDDs as a multilevel representation in a factored form (see section 2.3). For each net node the FDD representing the on-set is generated. While generating the FDDs, the Davio decomposition is performed immediately, by creating net nodes out of the nodes of the FDDs. The structure of the resulting multilevel network is given by the structure of the former FDD. Remark that eliminated FDD nodes result in nodes of the net, though. They are FDD nodes with isomorphic subtrees \((f_0 = f_2 \text{ or } f_1 = f_3)\), so their corresponding net node has two inputs only. This fact is exploited in the area optimisation using phase assignment, described below.

The result of the decomposition is a highly decomposed multilevel net with nodes having at most 3 inputs (f.e. the FDD of fig. 1 results in the decomposed network of fig. 4). This is somewhat same to tech_decomp of MIS-PGA, the complexity of the decomposed network is also comparable to the complexity of the network after tech_decomp.

![Figure 4: FDD based node decomposition](image)

FDDs are used for efficient representation of this decomposed nets. So the complexity problem of \([29]\), which limits the iterated application of the Shannon expansion, is solved. The decomposed net nodes then were connected given by the topology of the old multilevel net. The result is a multi-multilevel net of feasible nodes. With the use of FDD for representing the decomposed net, reconvergent paths can be easily detected by the standard reduction mechanisms of the decision diagrams. Reconvergent paths are then given by shared sub-graphs in the FDD, i.e. FDD nodes having more than one predecessor (see fig. 4).

4.4 Area Optimisation

In this step area optimisation of the final solution is done by optimising each node of the network locally. This can be performed by optimising the structure of the FDD. Two methods were applied: phase assignment and FDD reduction.

![Figure 5: Influence of phase assignment on two net nodes](image)
Since LUT were used to implement the functions later, phase assignment will have no extra costs for inverters. In opposite to classical phase assignment methods [5] we optimise each net node taking advantage of the properties of the FDDs. Since the structure of the FDD of the function \( f \) differs from the FDD for the complemented function \( \bar{f} \), for each node in the network a heuristic decides, whether \( f \) is calculated or its complement \( \bar{f} \). Phase assignment is done before the FDD based decomposition. We traverse the whole network starting at the primary inputs and apply phase assignment to each net node. Fig. 5 shows this in an example, where two redundant inverters were added to the network, one to use function \( f_B \) in complemented case, one at its fanout node to retain the correct behaviour of function \( f_A \).

For further reduction of the FDD structure, the methods from section 2.4 were applied. The optimisations are targeted to perform common subexpression extraction, by producing multi-referenced FDD nodes (done by the FDD reduction mechanism, see above) or to obtain much eliminated FDD nodes, which have less input and therefore are easier to collapse in the covering phase. Remark, that neither variable ordering nor polarity optimisation changes the behaviour of the function of the FDD. It only optimises the graph-based representation.

4.5 Delay Optimisation

For delay optimisation in the covering phase, the number of levels of the multilevel network is reduced using the level numbering and collapsing approach of [26, 30].

![Figure 6: Covering into LUT by level reduction](image)

The sketch of the algorithm is as follows: In the decomposed multilevel network the level of each net node is computed in postorder. Starting at the outputs, the network is traversed in preorder. As long as it remains feasible, each net node is collapsed with its fan-in net node having the largest level. Then the levels were adjusted due to the level reduction. Fig. 6 shows an example of such covering by level reduction. In the example each net node has its level written on the upper right-hand corner.

4.6 Routing Optimisation

After the covering phase the LUT network is optimised targeted to routability. Each net node has the following properties, which affect the results of the placement and routing phase: it is a feasible node, and due to the FDD based decomposition each net node has at least one or mostly two primary inputs as fanin and the other fan-in nodes are given by the (collapsed) successor nodes of the former FDD.

Now, to enhance routability, the ideas of cellular approaches [28] were taken into account, making the following propositions for Xilinx FPGAs of 3000 series:

1. Primary inputs should be distributed on the chip via long lines where possible.
2. The wiring between the LUT should be done using direct interconnect. Otherwise the general purpose connect is used for the wiring.

Following the cellular approach the final multilevel network is tried to be placed on the chip preserving the topology of the former DAG. In case of Xilinx FPGA this can be done by giving each LUT an area location
property which will be a constraint for the Xilinx APR tool. So net nodes which are neighboured in the DAG representing the multilevel network, were placed as neighbours on the FPGA and can be routed using direct interconnect in most cases.

5 Results

To compare our method we took results from [30], the network we started from was optimised using MIS. The results of our mapping method were targeted to 5-input LUT. In table 5 the number of LUT (#LUT) and the length of the critical path (#lvl) of the FDD based mapping were compared with Techmap-D [30], MIS-PGA-d [26] and shortle-d [16].

<table>
<thead>
<tr>
<th>MCNC91</th>
<th>#in</th>
<th>#out</th>
<th>Techmap-D</th>
<th>MIS-PGA-d</th>
<th>shortle-d</th>
<th>FDD based</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#LUT</td>
<td>#lvl</td>
<td>#LUT</td>
<td>#lvl</td>
<td>#LUT</td>
<td>#lvl</td>
</tr>
<tr>
<td>5xpl</td>
<td>7</td>
<td>10</td>
<td>17</td>
<td>2</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>9sym</td>
<td>9</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>clip</td>
<td>9</td>
<td>5</td>
<td>40</td>
<td>3</td>
<td>54</td>
<td>3</td>
</tr>
<tr>
<td>C499</td>
<td>41</td>
<td>32</td>
<td>148</td>
<td>4</td>
<td>199</td>
<td>8</td>
</tr>
<tr>
<td>duke2</td>
<td>22</td>
<td>29</td>
<td>175</td>
<td>4</td>
<td>164</td>
<td>6</td>
</tr>
<tr>
<td>misex1</td>
<td>8</td>
<td>7</td>
<td>18</td>
<td>2</td>
<td>17</td>
<td>2</td>
</tr>
<tr>
<td>misex2</td>
<td>25</td>
<td>18</td>
<td>41</td>
<td>3</td>
<td>37</td>
<td>3</td>
</tr>
<tr>
<td>rd84</td>
<td>8</td>
<td>4</td>
<td>16</td>
<td>3</td>
<td>13</td>
<td>3</td>
</tr>
<tr>
<td>sao2</td>
<td>10</td>
<td>4</td>
<td>45</td>
<td>4</td>
<td>45</td>
<td>5</td>
</tr>
<tr>
<td>vg2</td>
<td>25</td>
<td>8</td>
<td>36</td>
<td>4</td>
<td>39</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 5: Comparison of FDD based mapping

The results show that symmetric functions or functions with lot of EX-OR gates like C499 can better be mapped using the Davio expansion. Other examples (like misex1 or misex2) better use the BDD representation than our approach.

6 Conclusion

A mapping method for lookup-table based FPGAs was shown. For the decomposition of the multilevel network into feasible nodes it uses the Davio expansion, which graphically can be represented using Functional Decision Diagrams (FDDs). For area reduction optimisation techniques for FDDs were applied. For delay reduction a level numbering approach is used which controls the covering phase. Routability is taken into account using a cellular approach.

Further research will be in the combination of BDDs and FDDs to mix the Shannon with the Davio expansion. Other techniques for the covering phase will be also implemented.

References


