Analysis and Evaluation of a Hybrid Interconnect Structure for FPGAs

Renqiu Huang and Ranga Vemuri
University of Cincinnati, Cincinnati, OH 45220
{huangr, ranga}@ececs.uc.edu

Abstract

In this paper, a cluster-based FPGA is proposed. The proposed FPGA has a hybrid interconnect structure which takes advantages of both mesh and tree topologies. We analyze the area and performance of proposed FPGA in terms of the needed switches by comparing with those of conventional FPGAs. We evaluate the proposed architecture on a series of benchmark designs. The experimental results show that the proposed model can significantly reduce the routing area, achieve high performance and admit more implementations of various designs at the price of a modest increase of switches required for that architecture.

1. Introduction

Field-programmable gate array (FPGA)-based implementations map designs onto an array of configurable logic blocks that are connected by programmable interconnects. Compared to the custom designs, this design style requires fewer design iterations and has the advantage of shorter time-to-market. However, the system performance and logic density in FPGA-based implementation are not as high as those of custom designs due to the area and performance overhead of programmable logic and interconnect.

The programmable interconnect is a key design element in FPGAs. Studies show the programmable routing structure occupies about 70−90% of the overall chips silicon real estate, and 60−80% of overall design delay are attributed to programmable interconnects [1, 2]. The current trend in the FPGA industry is to pack more and more transistors on a chip (till ten million gates in state-of-art FPGAs [3]). This trend puts an ever-increasing burden on the routing structure.

Two-dimensional (2-D) array FPGAs have good routing wiring utilization and are easy to be computed for computer aided design (CAD) tools, but its Manhattan connection scheme makes the delays of long wires grow linearly. In order to improve the performance of FPGAs, commercial designs provide fixed long interconnections. Works in [4, 5, 6, 7] describe hierarchical interconnects based on tree structures show that both density and performance can be further improved. However, their routing flexibility, are largely depended on the designs of connection topologies between any two consecutive levels in the tree network.

This paper presents a cluster-based FPGA with a hybrid interconnect architecture, which takes both advantages of mesh and tree interconnects. In the proposed architecture, several CLBs are formed as a cluster, and tree networks are imposed over the mesh arrays to stitch parts of clusters together. The routing is performed through a hierarchical fashion: mesh and tree routings.

The organization of the rest paper is as follows. We present our architecture model in the next section. In Section 3, we analyze and estimate the area and performance of proposed model in terms of switches needed. The experimental results based on the MCNC benchmarks are provided in Section 4. Finally, Section 5 summarizes this work.

2. Architecture Modeling

A typical two-dimension island-style FPGA architecture [8, 1] is illustrated in Fig.1. It consists of a 2-D array of configurable logic blocks (CLBs, labeled as “L”) and horizontal and vertical routing channels. The input and output pins of each CLB are connected to the connection blocks (labeled as “C”) in the adjacent channels. The connections between different connection blocks are performed through switch blocks (labeled as “SM”) located in the each channel intersection.

Channel width, symbolized as “W”, is defined as the number of wires or tracks in a channel. Those tracks are the fixed connections between connection blocks and switch blocks. W = 3 in Fig.1. In order to make a connection flexible, there are a number of switches for each track to route that track to CLB through connection block or to the tracks of other sides of the corresponding switch block. For example, in the Xilinx’s XC4000 switch block, each programmable point has 6 switches. It is reasonable to consider that SW_{clb}, the total number of switches required for one CLB, is at least related to W as: SW_{clb} = O(W). Note that the above is the lower bound of SW_{clb}. It would be approaching SW_{clb} = O(W^2) when the routing flexibility of each track is increasing. If the array size is N × N, the total switches of that FPGA are SW_{total} = O(N^2W^2).

Since N is fixed, W is a key role to measure the number of switches. Normally, W should be determined as the maximum number of needed tracks by routing a set of benchmark circuits completely within a given performance.

The inherent diameter of mesh structure is N, this means that the path will pass through 2N − 1 switches in the worst case. The performance of an FPGA can be increased by reducing the number of passing switches along signal path. Tree structure [5, 10] decreases the diameter from N to log N. All-
though this kind of hierarchical structure decreases the passing 
switches from $2N - 1$ to $2\log N - 1$ in the longest path, great 
attentions are needed to put on the switch block design and a 
larger number of trees are needed to increase the routing flexi-
bility. This can be illustrated by comparing the bisection width 
of their interconnect networks. The bisection width $[9]$ of a 
network is the minimum number of wires that have to be re-
moved in order to disconnect the network into two topologi-
cally identical halves. Consider the simplest case: the pure 
mesh network ($W = 1$ in Fig.1) and a binary tree network. The 
bisection width of mesh structure is $N$, while it is only $1$ in the 
binary tree.

The 2-D mesh interconnect structure works well for the 
shorter connections and is easy to be synthesized by the CAD 
tools. For the long wires, the tree interconnect is more at-
tractive. How does a combination of mesh and tree work? 
This motivates us that a hybrid interconnect structure may take 
both advantages of mesh and tree-like interconnect networks. 
Since most routings in FPGAs are short connections, we may 
control the connections between consecutive nodes of trees by 
grouping several leaves together as a cluster and performs all 
short connections within in that cluster. That is to limit the 
local routings within clusters and complete part or all external 
routings between clusters through trees. Hence a hierarchical 
interconnect is constructed in Fig.3. For the clarity, we only 
show the structure in one dimension. The other dimension has 
similar structure due to the symmetric. 

Let $k \times k$ denote the size of cluster, there are $k$ leaves (from 
henceforth, we use the terms leaf and CLB interchangeably) 
for one dimension of cluster. Each leaf in a cluster has two 
trees connected to the corresponding leaves in the other clus-
ters. Namely, in each dimension of an array FPGA, every two 
 corresponding CLBs located on two clusters with distance $k$ 
are formulated as two leaves of a binary tree. The same policy 
is applied to the root nodes from level 1 to level $\log N - \log k$ 
recursively. Therefore, instead of one, $k \times k$ mesh-of-trees 
(MoTs) are constructed and organized on a cluster-based ar-
ray.

The topology of MoT is shown in Fig.2. The leaves of the 
tree are exactly $N \times N$ nodes of mesh. In each row and each 
column, wires and additional nodes are connected to form a 
binary tree. MoT has polylogarithmic $\Theta(\log N)$ or $\Theta(\log^2 N)$ 
running time on a wide range problems described in [9], which 
is dramatically faster than the typical running times of $\Theta(N)$ or 
$\Theta(N^2)$ for algorithms on meshes or trees. Further, works in 
[11, 7] show that mesh of tree permits area-efficient layout. 

After construction, $N/k$ leaves are put together remotely 
through a tree network (shown in Fig.4, where $N = 16$ and 
$k = 4$). For the mesh array (level 0), we assume the same struc-
ture as the typical island-style FPGA. However, $k \times k$ CLBs 
are grouped as a cluster in which local routings are performed. 
The long connections between clusters are routed through tree 
networks if those tree routings are possible.

The switch blocks allow the needed inter- and intra-cluster 
connections to be realized. Since the same interconnect struc-
tures of island-style are applied at level 0, we assume that the 
typical connection topology such as disjoint or subset switch 
block is used. As to the tree switch block “$S_T$”, a simple 3-
side interconnect pattern is used. The number of parent pins 
is set to the summation of pins of two children. That is, every 
child pin can respectively be connected to a parent pin. For 
the connections between children, the flexibility of child pin is 
set to the number of pins of one CLB. By doing so, it provides 
the full connectivity at the beginning of each tree network. In 
practice, it is not necessary to guarantee that all connections 
would go through tree network since most routings are per-
formed within clusters in a mesh fashion. To understand how 
many long connections would go through the tree network, a 
stoastic instead of combinatorial analysis may reveal more 
associations, and we leave it as the future investigation. As 
we will see in next section, even at worst case, this connecti-
ion scheme will not significantly increase the number of switches 
needed in the proposed architecture as the sizes of trees are 
relatively small ($N/k$, instead of $N^2$).

Normally, in a mesh array, each CLB has four neighbors 
(east, south, west and north). By applying the proposed hybrid 
network, the neighbors have been expanded from four to six 
(Fig.5). Two more neighbors are the siblings of that leaf in its 
two trees. Note that all neighbor connections could be realized 
by only one or two switches. With those expanded neighbors 
and imposed tree networks, it is possible to support more short 
connections, hence achieving performance improvements. 

3. Model Analysis

Since switches dominate FPGA design, from FPGA archi-

tect’s view, it is highly expectable to achieve high perfor-
mance and high routability with the available switches in that FPGA. Routability describes the effectiveness in utilizing the programmable routing resources. Although there are lots of discussions about the issue of routing ability, unfortunately, there is no formal definition of routability. General understanding is that the routability of FPGAs depends strongly on the number of available connections. Bisection width is the vital factor which may measure the connections. The higher the bisection width, the larger the space of routing solutions and the easier the routing. We use the bisection width to evaluate the switch requirements between different architecture models.

3.1. Comparison of Switches Needed

Let’s consider two FPGA models: Type-A, the conventional array FPGA (Fig. 1), and Type-B, the proposed FPGA, which is a hierarchical combination of mesh and tree interconnects mentioned in the last section. For the further analysis, let \( m \) denote the number of pins of a CLB, and \( I \) is the number of inputs and outputs from one side of CLB to the channel. We have \( I = m/4 \) for mesh architecture. For example, \( m = 8, I = 2 \) in Fig.1. The sizes of two FPGAs are set to \( N \times N \). In order to be comparable, all bisection widths at highest level are set to be equal, which is \( BW = pmN^2 \). Here, \( mN^2 \) is the total number of possible signals. Since any signal may cross the middle line of chip several times or not pass it at all, we multiply it by a parameter \( p \). Let \( W_a \) be the channel widths of Type-A model. According to Fig.1, there are \( N \) connection blocks and \( N-1 \) switch blocks per row/column either along horizontal or vertical dimension. The bisection width can be counted as:

\[
BW_A = N \times I + 2 \times (N - 1) \times W_a
\]  (1)

Let \( BW_A = BW \) and substitute \( I \) with \( m \), we have:

\[
W_a = \frac{mN(4pN-1)}{8(N-1)}
\]  (2)

Following the definitions in [8], if we assume the flexibility of connection block is \( W_e \) and the flexibility of switch block is \( F_t \), then for the Type-A FPGA, the total number of switches is:

\[
SW_A = N^2 \times (2 \times 2 \times I \times W_a + \left( \frac{4}{2} \times F_t \times W_a \right))
\]

\[
= \frac{m(m+6F_t)(4pN-1)N^3}{8(N-1)}
\]  (3)

In the above calculation, we associate each logic block with one switch block and two connection blocks. Therefore, the total switches are the \( N^2 \) times of summation of switches in one switch and two connection blocks.

For the total number of switches of Type-B model, we approximate it as the summation of the mesh switches and tree switches. Let us estimate the switches used in the tree connections first. The total number of switches needed for the trees in the proposed interconnect is as following

\[
SW_T = 2kN \sum_{i=1}^{\log \frac{N}{k}} \left( 2 \times 2^{i-1} \times m + 2^i m \right) \times 2^{-i} \times \frac{N}{k}
\]

\[
= 2m(1+N)N^2 \log \frac{N}{k}
\]  (4)

We give some explanations for \( SW_T \). There are totally \( 2kN \) trees. In a tree network, at \( i \)th level, where \( 1 \leq i \leq \log \frac{N}{k} \), the switches for the connections between one child and the other child in one single block are \( 2^{i-1} \times m \), the switches for the connections between parent and children in that block are \( 2^i \times m \), and the numbers of switch blocks are \( 2^{-i} \times \frac{N}{k} \).

Next, we look at the switches used in the mesh connections. Recall that the proposed FPGA is cluster-based. This implies that there are two kinds of channels. One is the channel within a cluster, while the other is the channel between clusters. Two channels can be viewed as the results of different routings. The former is come from the local routing, while the latter is affected by the long routing. In order to keep the same bisection width at each partition level between those two models, obviously, the channel width within the cluster should be set to \( W_e \). The channel width (denoted as \( W_b \)) between clusters is calculated as following.

The bisection width of proposed model is \( NW + 2(N-1)W_e + mnN^2/2 \). Similar to \( BW_A \), the first two items are the mesh bisection widths of proposed FPGA. The last item is the tree cut, which is calculated as follow. The cut will bisect \( k \) trees per row or column, and there are totally \( N \) rows or columns. Since each logic block has \( m \) pins, the total cut of tree will be \( k \times m \times N \times 2^{\log \frac{N}{k}-1} \), which is \( mnN^2/2 \).

\[
W_b = \frac{mN(4pN-2N-1)}{8(N-1)}
\]  (5)

Finally, the totally switches needed for the proposed model would be approximated as the summation of all switches used in the clusters, between clusters, and in the tree network, which result in the following equation.

\[
SW_B = \frac{N^2}{k^2} (mk(k-1) + 6F_t(k-1)^2)W_e + 2(N-1)(m+6F_t)W_b + SW_T
\]  (6)

In the above equation, the first item is the number of switches needed for all clusters. It is calculated as following. Since there are totally \( \frac{N^2}{k^2} \) clusters, for each cluster, the numbers of connection and switch blocks are counted as \( k(k-1) \) and \( (k-1)^2 \) respectively. The second item is the total number of switches required for the connections between clusters. There are totally \( \frac{N}{k}-1 \) channels in one direction and the number of switch and connection blocks are approximated as \( N \) for each channel. By substitution, we have

\[
SW_B = \frac{m(k-1)(mk+6F_t(k-1))(4pN-1)N^3}{8k^2(N-1)} + \frac{m(m+6F_t)(4pN-4k-1)(N-k)N^2}{4k(N-1)} + 2m(1+N)N^2 \log \frac{N}{k}
\]  (7)
From Eq.3 and Eq.7, we may see that both models need $O(N^3)$ switches to achieve the same number of connections. We would like to see the effects of various parameters. In order to compare the switch requirements between those two models more clearly, we sketch the percentages of increased switches ($\frac{O(3s)−O(3W)}{O(3W)} \times 100\%$) for different values of $k$ with different $m$, $F_s$, $p$ in Fig.6, Fig.7, and Fig.8 respectively. During each calculation, $N$ is set to 128, since this number can cover the sizes of all commercial FPGAs currently available. For the non-changed parameters, we fix $F_s = 3$, $p = 0.5$, and $m = 40$ correspondingly. When compared with conventional architecture, the switch increments required in the proposed structure are largely depended on the size of cluster while kept in the same order.

Figure 6. Switch overhead for various values of $m$

![Figure 6](image1)

Figure 7. Switch overhead for various values of $F_s$

![Figure 7](image2)

Figure 8. Switch overhead for various values of $p$

![Figure 8](image3)

Obviously, in terms of switches overhead, the smallest $k$ is desirable. For the lowest point which corresponding to $k = 1$, the switches required for each cluster would be zero from the Eq.7. This implies that the tree switches will dominate the total number of needed switches. From Eq.3 and Eq.4, we know that the switches required for the mesh and tree are $O(N^3)$ and $O(N^2 \log N)$ respectively. This is why the large reductions are observed in those figures when $k$ is smaller. The increased switches in the tree network are balanced by the channel width reductions in the mesh array. In the later section, based on a series of benchmark studies, we will see this is exactly true. $k$ can not be too small, otherwise the dominated tree connections would be easily suffered from congestion problem due to the lack of alternative mesh connections. Therefore, we only look at the switch overhead for $k > 1$. The mesh switches begin to dominate when $k$ is slightly increased.

For the two architectural parameters $m$ and $F_s$, the increased switches are under 10% even in the worst cases in the above figures. For parameter $p$, which describes the connection richness in the implementation of a design, we observe that the switch overhead is much high when $p$ is small. From Eq.5 the channel width between clusters would be approaching to zero when $p$ is decreasing to certain value. This means that the tree networks can provide much more connections than those required for the design. Since the conventional array FPGAs need less channel tracks to implement that design (hence the less switches, see the beginning in Section 2) while the tree switches in the proposed model remain the same, the overhead will be high. When $p$ is growing larger, the overheads are still comparable with other two parameters.

One important property is that the switch overheads are decreasing quickly when parameter $p$ or $F_s$ is increasing. This would benefit FPGA design a lot since today both FPGAs and connections in a design are becoming larger and larger. We also observe that the switch overhead is decreasing when the ratio $\frac{1}{k}$ is increasing. However, a larger $k$ will undermine the performance improvement. Since most nets in FPGAs are short nets, the performances of those nets implemented in mesh routing are limited to $k$. Further, we will see next that the performance improvement of long connections due to tree networks is also weakened when $k$ is larger.

3.2. Performance Gain

The proposed architecture encourages the short implementations of signal paths, which can be either realized through mesh routings for the short connections, or through tree rout-
ings for the long connections. Since the performance of an FPGA can be improved by reducing the number of passing switches along signal path, it is reasonable to relate the number of switches to the wire length. We are looking at the average case and use the methodology [12] to approximate the average connection length $T$.

$$N(l) = q(l)D(l)$$  \hspace{1cm} (8)$$

Where $N(l)$ is the number of net of length $l$, $D(l)$ is the number of valid two-pin net placement sites and $q(l)$ is the probability that a placement site is occupied. Therefore, we have

$$T = \frac{\sum_{l=1}^{N} l N(l)}{\sum_{l=1}^{N} N(l)}$$  \hspace{1cm} (9)$$

Further, if we denote $T(k)$ as the average length of all connections whose lengths are greater than $k$, we have

$$T(k) = \frac{\sum_{l=k+1}^{N} l N(l)}{\sum_{l=k+1}^{N} N(l)}$$  \hspace{1cm} (10)$$

For a pure mesh model, the switches used for above average length are approximated as:

$$T_A = \sqrt{\frac{2}{l(k)}} + 1$$  \hspace{1cm} (11)$$

$\sqrt{2}$ is employed to measure the distance from Euclidean to Manhattan in the above equation. For the proposed model, although we expect all long connections are routed through tree networks, we use the normalized routing possibility to model the routing algorithm since those connections could be either realized through mesh or tree routings. Obviously, the performance of proposed FPGA is dependent on the size of cluster. When $k = 1$, most of nets are routed through the tree network and we may get the tree’s performance at the best case. On the other hand, when the size of cluster is large enough, such as $k = N$, nets are exactly routed through Manhattan scheme and the mesh’s performance is achieved.

$$T_B = \left( (2 \log(\sqrt{\frac{2}{l(k)}}/2) - 1) \frac{N - k}{N} + (\sqrt{\frac{2}{l(k)}} - 1) \frac{k}{N} \right)  \hspace{1cm} (12)$$

We sketch the above two equations and their ratios in Fig.9 for different lengths of long connections (namely, $k$). From that Figure, the best range of improvement is around $[4, 10]$. To see that improvement clearly, we redraw that gain in Fig.10 according to $\frac{N}{2N} \times 100\%$. The peak point is at $k = 9$ with achieving as much as 75% improvement in Fig.10. Note that the above improvement is only for the implementations of long nets, while the switch requirements for short connections are assumed to be comparable since the short connections would be implemented through mesh routings for both models.

4. Evaluation

We present the experimental studies in this section. We use T-VPack and VPR (v 4.30) [13] to map and place the design. We use sample architecture file 4-A-lut-sanitized.arch as our base mesh unit. Therefore, each CLB has 4 4-LUTs, 10 pins for input and 4 pins for outputs. We first use T-VPack to pack the netlist to support the sample architecture, and then use VPR to produce the placed design. By default, the VPR tries to place the circuit in the smallest rectangle in which the circuit fits. After placement has been done, we stretch that rectangle into a large mesh structure. According to the cluster size, the size of stretched rectangle is the least 2’s power which is not less than original size. Since array sizes has been changed, we run VPR again to update the placement with the newly stretched sizes.

For the conventional model, VPR (running at default settings) is used as the routing tool. For the proposed FPGA model, since it allows routing to be done in both mesh and hierarchical tree interconnects, we have developed a custom routing tool to take advantages of those features. We constrain the short nets (all terminals of a net are guarded within the same cluster and the neighboring CLBs of that cluster) are routed through mesh routing. For a net which has terminals scattering at different clusters, the routing process is as following.

At the global routing step, the terminals are grouped according to their corresponding clusters. The routings for the terminals of each group are performed within that cluster. The connections between groups are the long nets which will go through the tree networks as much as possible. Hence, our routing process consists of two routing procedures: tree and mesh routings. The tree router is running on the top of mesh router. The tree routing algorithm is based on [14] while the algorithm of mesh routing sticks to the breadth first search in VPR.

As to the tree routing, there are two main issues. One is the net ordering, and the other is the path search order. The ordering of all long nets is created to be fixed according to their lengths. The key of path order is to determine which gross tree (totally $2N$) is chosen as the viable route. If there exists two terminals which belong to two leaves of a tree, the path cross-product
ing that tree is only route for choice. Otherwise, we search the trees with the intermediate leaves which are available to be connected to both two terminals with less distance than that of Manhattan scheme. Further, in order to assure signal integrity and predictable delays, the wire switch of segments in the tree connections are assumed to be buffered while it is not in the mesh network.

Table 1. Array sizes and channel widths

<table>
<thead>
<tr>
<th>Design</th>
<th>Array size</th>
<th>VPR</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>nx, ny</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>alu4</td>
<td>32, 32</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td>apex2</td>
<td>32, 32</td>
<td>37</td>
</tr>
<tr>
<td>3</td>
<td>apex4</td>
<td>32, 32</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>bigkey</td>
<td>32, 32</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>clma</td>
<td>64, 64</td>
<td>56</td>
</tr>
<tr>
<td>6</td>
<td>des</td>
<td>32, 32</td>
<td>19</td>
</tr>
<tr>
<td>7</td>
<td>diqeq</td>
<td>32, 32</td>
<td>25</td>
</tr>
<tr>
<td>8</td>
<td>disp</td>
<td>32, 32</td>
<td>20</td>
</tr>
<tr>
<td>9</td>
<td>elliptic</td>
<td>32, 32</td>
<td>45</td>
</tr>
<tr>
<td>10</td>
<td>ex1010</td>
<td>64, 64</td>
<td>43</td>
</tr>
<tr>
<td>11</td>
<td>ex5p</td>
<td>32, 32</td>
<td>38</td>
</tr>
<tr>
<td>12</td>
<td>frisc</td>
<td>32, 32</td>
<td>47</td>
</tr>
<tr>
<td>13</td>
<td>musek3</td>
<td>32, 32</td>
<td>37</td>
</tr>
<tr>
<td>14</td>
<td>pdc</td>
<td>64, 64</td>
<td>67</td>
</tr>
<tr>
<td>15</td>
<td>s298</td>
<td>32, 32</td>
<td>30</td>
</tr>
<tr>
<td>16</td>
<td>s354811</td>
<td>64, 64</td>
<td>33</td>
</tr>
<tr>
<td>17</td>
<td>s358541</td>
<td>64, 64</td>
<td>32</td>
</tr>
<tr>
<td>18</td>
<td>seq</td>
<td>32, 32</td>
<td>39</td>
</tr>
<tr>
<td>19</td>
<td>spla</td>
<td>32, 32</td>
<td>41</td>
</tr>
<tr>
<td>20</td>
<td>tseng</td>
<td>32, 32</td>
<td>24</td>
</tr>
</tbody>
</table>

We mapped, placed and routed the 20 largest MCNC benchmark circuits on our proposed FPGA. We report the minimum number of channels required for VPR and different cluster sizes in Table 1. The third column shows the array sizes for implementations of all designs. The minimum track counts for VPR are listed in the 4th column. As can be seen from that table, channel widths have been reduced largely. Even for a larger k, up to 40% reductions have still been achieved for the larger designs, such as clma. We also observe that the increase of channel width is slow when the cluster size increases, and for each cluster size, the deviations of channel widths are relatively small. This is especially useful, because more designs would be admitted for a given FPGA with a fixed channel width.

We also plot the routing area and critical path delay for VPR and different clusters in Fig.11 and 12 respectively. The routing area is the summation of mesh routings and tree routings in terms of minimum-width transistor area [15]. The minimum routing area of each tree switch is calculated as the following. We assume each tree switch is implemented with a pass transistor, the number of minimum-width transistor required for that is 11.5. As mentioned early, the wire segment of tree connections is assumed to be buffered. We use the same buffer size as in Chapter 7 [15] and it occupies 19.7 minimum-width transistor areas. Therefore, the total area for each tree switch needed in the design is 31.2 minimum-width transistors. As to the delay, we use the same timing numbers in the sanitized architecture file. As explained in the VPR documentation, although they are not complying with the foundry process data, those numbers are still reasonable enough to allow CAD experimentation. From Fig.11 and 12, it is clear that considerable improvements are achieved. For the routing area, we can see as much as 30% reduction is achieved. This is because most of nets are routed through tree networks, resulting in a huge saving of switches needed. More significant achievements are observed for the performance improvements. Up to 50% reductions are possible for the large designs. Again, the improvements are due to our intentional introduction of the tree network. From those two figures, the larger the design, the more the improvement for both routing and performance. We notice that improvements increases when k decreases. While a small k is desirable, we also notice that the performances are degraded for the design ID 4, 7 and 14 when k is further reduced. This may be explained when k is too small, the routings will go through more mesh interconnects instead of tree connections due to the congestion problem. Since we enlarge the array size, much more tree networks are available for the routings of long nets and the congestions are largely relieved. This is one of limitations of our experiments. Therefore, considering those factors, a slightly large k is more implementable.

5. Summary

The routing area and performance in FPGAs is generally limited by the programmable interconnects. We proposed a cluster-based FPGA with a hybrid interconnect structure which takes advantages of both mesh and tree interconnect networks. The proposed architecture is developed with a combinatorial analysis which examines the number of switches needed. The benchmark studies show that our architectural model has less switch accrued effects due to the introduction of tree interconnects. By encouraging local routing and short implementations of long connections, significant reductions in the routing area and long path delay can be achieved. In additional, the CAD tools for the tradition island-style FPGAs, considering its immense popularity, can be easily adapted.

References


