Reliability Optimization of Reconfigurable FPGA Based on Second-Order Approximation Domain-Partition

Mi Zhou and Lihong Shang
School of Computer Science and Engineering
Beihang University
Beijing 100191, China
zooome@gmail.com; shanglh@buaa.edu.cn

Yu Hu
Key Lab. of Computer System and Architecture, ICT
Chinese Academy of Sciences
Beijing 100190, China
huyu@ict.ac.cn

Abstract—The problem of planning the overlaps of multiple alternative configurations is critical to maximize the reliability of a reconfigurable fault-tolerant system based on field programmable gate arrays. To address the problem, an unnecessary assumption made in previous work is removed and a second-order approximation domain-partition method is proposed. Experimental results on ITC99 benchmark circuits demonstrate the proposed technique outperforms previous work in failure rate with less memory cost to store configurations.

Keywords-reconfigurable system; fault tolerant; domain-partition

I. INTRODUCTION

Field Programmable Gate Array (FPGA) devices have been widely used due to their flexibility and functionality merits. However, along with fast shrinking feature size, FPGA devices are more vulnerable to electromigration. Especially, SRAM-type FPGAs and FPGAs with higher density of embedded SRAM blocks are more sensitive to Single Event Upsets (SEUs), which induces temporary errors.

Fortunately, there are always a number of “virtual” spare resources in FPGAs [2], which can be used to replace faulty cells and thus repair the system. By this means, the inherent redundancy of FPGAs can be exploited to implement fault recovery so that the reliability of FPGA based fault-tolerant (FT) systems can be improved [3].

Usually, an FPGA based system recovers from faults by means of reconfiguration, in other words, reloading an alternative configuration containing no faulty resource. Any defect on the overlaps of multiple configurations may make all configurations it involves fail. Thus, the problem of planning the overlaps of multiple alternative configurations is critical to maximize the reliability of a reconfigurable FT system.

To address the above problem, a second-order approximation domain-partition (SOADP) method is proposed in this work. The problem of planning the overlaps of configurations so as to maximize the reliability is approximately formulated and solved as a second-order planning problem.

II. RELATED WORK

The problem of assigning reconfigurable resources was converted into a pattern optimization under given area (POGA) problem in [4]. A domain-partition (DP) model was proposed. A first-order optimal solution (POGA solution for short) was conjectured to plan the overlaps of configurations so that reliability can be optimized under given redundancy and number of configurations. However the POGA solution is based on unnecessary assumptions, which results in limited solution space. Thus the POGA solution is not always global optimal.

A “k+m” method was proposed in [3]. In the work, each FPGA is partitioned into multiple tiles and each tile contains $k+m$ columns of configurable resources. In each configuration of a certain tile, $k$ columns are used to implement the original circuit and $m$ columns are left as spares. Other examples of “k+m” method can be found in [5] [6] [7]. In [8], one spare column is preserved for each occupied column. Such a solution can be considered as a special case of the “k+m” solution where $k=1$ and $m=1$. The “k+m” solution is simpler to be implemented but lacks in flexibility. Sometimes one defect may fail a whole column.

In our previous work [9], the problem of maximizing the reliability is converted into a second-order planning problem and solved. The proposed SOADP method is an extension of [9]. Different from our previous work, tiling is introduced to further improve the reliability. An unnecessary assumption is removed to widen the solution space. Experimental results show more reliable solutions can be found with less time and less memory than the previous method.
III. DOMAIN-PARTITION MODEL

Introduction of the DP model will be presented before detailed description of the proposed approach.

The DP model introduced in [4] is an abstraction of a reconfigurable device with several versions of circuit configurations as follows:

Let \( S \) be a fault-tolerant design implemented on a reconfigurable FPGA. The domain-partition of \( S \) is defined as a \( 2^p \) dimensional vector \( N \), whose components are \( N(i) \).

That is, \( N = (N(0,0),...,N(010),N(100),N(101),N(110),N(000)) \), where \( p \) is the number of alternative configurations, \( v \) is a \( p \) dimensional vector whose elements are in \{0, 1\}, \( N(v) \) is the relative area of the overlap of the configurations corresponding with \( v \). An example of using the DP model in a duple modular redundancy (DMR) system with 3 alternative configurations is shown in Fig. 1, where the shaded blocks denote the occupied resources (for application) and the blank blocks represent the unoccupied resources (for spare).

\[
\begin{align*}
\text{Cnf}_1 & \quad \text{Cnf}_2 \quad \text{Cnf}_3 \\
\text{FPGA} & \quad \text{N(100)} \quad \text{N(101)} \quad \text{N(011)} \quad \text{N(010)}
\end{align*}
\]

Figure 1. An example of DP in a DMR system.

As shown in Fig.1, there are 3 alternative configurations (i.e. \( \text{Cnf}_1, \text{Cnf}_2 \) and \( \text{Cnf}_3 \)), and each of them is denoted by a row of blocks. Thus \( p=3 \) and we have: \( N = (N(000), N(001), N(010), N(011), N(100), N(101), N(110), N(111)) \). According to the definition of \( N \), we have \( N(100)=1/6 \) since the relative area occupied only by \( \text{Cnf}_1 \) but not occupied by \( \text{Cnf}_2 \) and \( \text{Cnf}_3 \) is 1/6. Similarly, \( N(101)=1/3 \) since the relative area occupied by both \( \text{Cnf}_1 \) and \( \text{Cnf}_3 \) but not occupied by \( \text{Cnf}_2 \) is 1/3. Note that \( N(111)=0 \) since there is no area occupied by all of the three configurations. Thus the domain-partition \( N \) of the FT system will be \( N=(0, 0, 1/3, 1/6, 1/6, 1/3, 0, 0) \).

Partition \( N \) must satisfy the following relations:

\[
\begin{align*}
\sum_{v \in V} N(v) &= 1, \quad (2) \\
\sum_{v \in V, v_i=1} N(v) &= 1/r \quad (1 \leq i \leq p), \quad (3)
\end{align*}
\]

where \( V \) is the set of all the \( p \) dimensional vectors whose elements are in \{0, 1\}, i.e. \( V = \{0,1\}^p \), \( v_i \) represents the \( i \)-th component of \( v \), \( r \) represents the redundancy that equals to the ratio that the area of all available resources in \( S \) over the area occupied by a single configuration.

Equation (1) means the area of each overlap can not be less than 0. Equation (2) means the areas of all available resources are obtained by summing up all the overlapping areas. Equation (3) denotes that the relative area of each configuration is \( 1/r \), which is derived from the definition of \( r \).

For example, in the FT system shown in Fig. 1, \( r=2 \) since there are total 6 blocks and a half of which are occupied in each configuration. We have \( V = (000, 001, 010, 011, 100, 101, 110, 111) \). Obviously, (1) and (2) are satisfied since \( N=(0, 0, 1/3, 1/6, 1/6, 1/3, 0, 0) \). The value of \( \sum_{v_i=1} N(v) \) equals the relative area occupied by \( \text{Cnf}_1 \), thus \( \sum_{v_i=1} N(v) = 1/6 + 1/3 + 0 + 0 = 1/2 = 1/r \).

Calculations about \( \text{Cnf}_2 \) and \( \text{Cnf}_3 \) are similar.

The reliability of \( S \) on time \( t \) under partition \( N \) can be formulated as:

\[
R(N) = 1 - e^{-\lambda t} \sum_{v \in V} \left( -1 \right)^{|v|} \exp(\lambda t \sum_{v \in V, v_i=1} N(v)), (4)
\]

where \(|v|\) denotes the number of \( 1 \)'s in vector \( v \), \( \lambda \) is the failure rate of all available resources in \( S \).

IV. SECOND-ORDER APPROXIMATION DP METHOD

The problem of maximizing reliability subjected to (1), (2) and (3) was proposed in [4]. However the problem was not solved due to its complexity. A first-order optimal solution (the POGA solution) was conjectured based on two assumptions as follows:

Assumption 1: It is assumed that under the optimal partition the area of overlaps of alternative configurations must be minimized. Furthermore, it must be satisfied that \( A(q-1)=q-p/r \) and \( A(q)=1-q+p/r \) to minimize the overlap, where \( A(n)=\sum_{v \in V, |v|=n} N(v), 0 \leq n \leq p \), \( r \) denotes the redundancy and \( q \) denotes the smallest integer not less than \( p/r \).

Assumption 2: It is assumed that the optimal partition is symmetrical (i.e. \( N(v) \) depends only on \(|v|\)) because of the symmetry of the problem. In other words, \(|v|=|l| \Rightarrow N(v)=N(l)\), \( v, l \in V \).

However, experiments show Assumption 2 is not necessary. For example, the “\( k+m \)” solution which does not meet assumption 2 may provide more reliable partition under same redundancy than the POGA solution sometimes. Thus we removed the assumption 2 to widen the solution space.

To reduce the time cost of calculating \( R(N) \). A second-order approximate formula of \( R(N) \) is proposed as follows.

Expanding (4) in the neighborhood of \( x=x_0 \), we get the following formula:
\[ R_{2}(N) = 1 - e^{s_{0}} \left( (1 - \lambda t - x_{0}) A(p) + \lambda t^{2} Y / 2 \right), \]

where \( Y = \sum_{v \in V} (-1)^{\|v\|_{p}} \left( \sum_{\{v \in V' \mid v_{i} \leq l_{j} \}} N(v) \right)^{2}. \)

According to (2), when Assumption 1 is made, it is easy to prove that \( A(p)=0. \) Thus, \( Y \) must be minimized in order to maximize \( R_{2}(N). \)

The problem of maximizing reliability under given redundancy \( r \) and number of alternative configurations \( p \) can be approximately formulated as the following second-order planning problem.

**SOADP Problem:** Find domain-partition \( \tilde{N} : \)

\[
\text{Min } Y(\tilde{N}) = \sum_{\{v \in V' \mid \|v\|_{q} = 1 \}} (-1)^{\|v\|_{p}} \left( \sum_{\{v \in V' \mid v_{i} \leq l_{j} \}} \tilde{N}(v) \right)^{2},
\]

s.t. \( \tilde{N}(v) \geq 0 \ (v \in V'), \)

\[
\sum_{\{v \in V' \mid \|v\|_{q} = 1 \}} \tilde{N}(v) = q - p / r,
\]

\[
\sum_{\{v \in V' \mid \|v\|_{q} = 1 \}} \tilde{N}(v) = 1 - q + p / r,
\]

\[
\sum_{\{v \in V' \mid \|v\| = 1 \}} \tilde{N}(v) = 1 / r \quad (1 \leq i \leq p),
\]

where \( V' = \{ v \in V \mid \|v\| = q \lor \|v\| = q - 1 \}. \)

Each \( \tilde{N}(v), v \in V' \) can be considered as a variable, thus there are \( C_{p}^{q} + C_{q}^{q-1} \) variables to be solved. The SOADP problem can be easily solved with mathematical tools, since all equality constraints are linear.

The objective function \( Y(\tilde{N}) \) can also be formulated as multiplication of matrices. Further discussion can be found in [9].

To tolerate more faults, it is popular to divide the entire FPGA into several tiles. Such a process is called “tiling” [3]. The proposed method can also be integrated with tiling by treating each tile as an independent FT system and applying SOADP on it. The circuit to be implemented on the FPGA also needs to be divided into pieces and be mapped into the FPGA tiles, so that each tile contains resources to implement the circuit as well as resources for spare. The reliability of the entire design can be evaluated as the multiplication of the reliability of all tiles. More details can be found in the experiments.

**V. CASE STUDY**

To illustrate how to implement the proposed method to design a FT system, we apply SOADP to the FT design shown in Fig. 1 as an example.

As mentioned in section III, the parameters have been evaluated as \( p=3 \) and \( r=2 \). Thus, \( q=\text{ceil}(p/r)=2 \). As mentioned in section IV, \( V' \) is the set of 3-dimensional vectors each of which contains \( q \) or \( q-1 \) \( s. \) Thus, \( V'=(001, 010, 011, 100, 101, 110). \) And we have:

\[
\tilde{N} = (\tilde{N}(001), \tilde{N}(010), \tilde{N}(011), \tilde{N}(100), \tilde{N}(101), \tilde{N}(110)).
\]

Let \( x = \tilde{N}(x_{1}, x_{2}, \ldots, x_{6}) \) for short, i.e.

\[
x_{1} = \tilde{N}(001), \quad x_{2} = \tilde{N}(010), \quad \text{etc., we have:}
\]

\[
Y(\tilde{N}) = x_{1}^{2} + x_{2}^{2} + x_{3}^{2} - (x_{1} + x_{2} + x_{3})^{2} - (x_{4} + x_{5} + x_{6})^{2},
\]

\[
-(x_{4} + x_{5} + x_{6})^{2} + (x_{1} + x_{2} + x_{3} + x_{4} + x_{5} + x_{6})^{2}.
\]

So for this example, its SOADP problem is:

Find domain-partition \( \tilde{N} : \)

\[
\text{Min } Y(\tilde{N}),
\]

s.t. \( x_{i} \geq 0 \ (1 \leq i \leq 6), \)

\[
x_{1} + x_{2} + x_{4} = 0.5, \quad x_{3} + x_{5} + x_{6} = 0.5, \quad x_{1} + x_{3} + x_{5} = 0.5, \quad x_{2} + x_{3} + x_{6} = 0.5, \quad x_{1} + x_{4} + x_{6} = 0.5.
\]

The above problem can be easily solved by the MATLAB toolbox. The solved optimal solution is \( x=(1/6, 1/6, 1/6, 1/6, 1/6). \) The above solution is equivalent to the POGA solution in this example. The corresponding optimal SOADP design of the FT system is shown in Fig. 2.

![Figure 2. Optimal design of the FT system where \( p=3 \) and \( r=2 \).](image)

To tolerate more faults, the above system can be divided into several tiles, each of which can be protected with the SOADP method. An example is shown in Fig. 3 where the design is divided into two tiles.

![Figure 3. Optimal design of the FT system where \( p=3 \) and \( r=2 \).](image)
VI. EXPERIMENTS

To evaluate the effectiveness of the proposed SOADP technique, several ITC'99 benchmark circuits [10] were implemented on Xilinx’s FPGAs. For the purpose of comparisons, the chosen circuits were also implemented under the POAG partition and the “k+m” approach.

Failure rate of $S$ under partition $N$ is used as a scale to measure the ability of fault tolerance of each implementations, which is defined as: $F(N)=1−R(N)$.

Parameters assignment is shown in Table I. Redundancy $r$ was set to the ratio that the number of slices of FPGA divided by the number of slices occupied by the chosen circuit. In the experiments FPGA devices were always chosen so that $r<2$, since this article focuses on fault recovery systems. When $r>2$, passive redundancy techniques such as passive Duple Modular Redundancy or Triple Modular Redundancy [11] [12] are more popular than fault recovery.

<table>
<thead>
<tr>
<th>General parameters</th>
<th>Circuit</th>
<th>b22</th>
<th>b21</th>
<th>b14</th>
<th>b12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>2v1000</td>
<td>2v500</td>
<td>2v250</td>
<td>2v40</td>
<td></td>
</tr>
<tr>
<td>Used slices</td>
<td>3035</td>
<td>1927</td>
<td>942</td>
<td>200</td>
<td></td>
</tr>
<tr>
<td>Total slices</td>
<td>5120</td>
<td>3072</td>
<td>1536</td>
<td>256</td>
<td></td>
</tr>
<tr>
<td>Used columns</td>
<td>19</td>
<td>16</td>
<td>10</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Total columns</td>
<td>32</td>
<td>24</td>
<td>16</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>$r$</td>
<td>1.69</td>
<td>1.59</td>
<td>1.63</td>
<td>1.28</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters of “k+m” implementations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tile I $k$</td>
</tr>
<tr>
<td>$m$</td>
</tr>
<tr>
<td>Number of Tile I</td>
</tr>
<tr>
<td>Tile II $k$</td>
</tr>
<tr>
<td>$m$</td>
</tr>
<tr>
<td>Number of Tile II</td>
</tr>
<tr>
<td>Tile III $k$</td>
</tr>
<tr>
<td>$m$</td>
</tr>
<tr>
<td>Number of Tile III</td>
</tr>
<tr>
<td>Total tiles</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters of POAG and SOADP implementations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tile A $p$</td>
</tr>
<tr>
<td>Number of Tile A</td>
</tr>
<tr>
<td>Tile B $p$</td>
</tr>
<tr>
<td>Number of Tile B</td>
</tr>
<tr>
<td>Total tiles</td>
</tr>
</tbody>
</table>

In the “k+m” implementations, each design was divided into several tiles. The number of tiles and parameters $k$ and $m$ were evaluated according to the total number of CLB (Configurable Logic Block) columns in the FPGA and the number of columns occupied by the chosen circuit. For example, the circuit “b22” implemented on FPGA 2v1000 takes 19 columns while there are total 32 columns available in the FPGA. Thus the circuit was partitioned into 7 tiles. In each of the first 6 tiles, there were 3 columns used to implement “b22” and 2 columns used as spares (i.e. $k=3, m=2$). These 6 tiles were noted as “Tile I” in TABLE I. In the last tile, there were 1 column for “b22” and 1 column for spare (i.e. $k=1, m=1$). This tile was denoted as “Tile II”. Thus there were totally 19 columns for “b22” and 13 columns for spares.

In the POAG implementations and the proposed SOADP implementations, each design was also divided into tiles so that the number of tiles equaled that in the “k+m” implementations. Parameter $p$ was evaluated which satisfied that the memory cost used to store the alternative configurations was not more than that in the “k+m” implementations. For example, according to [3], in the “k+m” implementation of “b22” introduced above, the memory space needed to store all the alternative configurations is $9.5$ times of that needed to store a single configuration. Thus in the POAG or SOADP implementations, the design was partitioned into 7 tiles. Each tile contained $1/7$ of all available slices. In each of the first 3 tile, which were noted as “Tile A” in TABLE I, there were 10 alternative configurations. In each of the last 4 tile (noted as “Tile B”), there were 9 alternative configurations. Thus it took $9.428$ times memory space to store all the alternative configurations.

Note that Mean Time to Failure (MTTF) in the FPGA is $1/\lambda$, in which the failure rate of the FPGA device is $\lambda$. In the experiments, we took into account the data presented in [13] and [14], and set the defect rate of the FPGA device as $\lambda=10^{-6}/[h]$. The failure rate of the implementations at a certain time point $t$ is listed in TABLE II.

<table>
<thead>
<tr>
<th>Time</th>
<th>Solution</th>
<th>Failure Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b22</td>
<td>b21</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$t=1/(2\lambda)$</td>
<td>&quot;k+m&quot;</td>
<td>0.0005</td>
</tr>
<tr>
<td></td>
<td>POAG</td>
<td>0.0015</td>
</tr>
<tr>
<td></td>
<td>SOADP</td>
<td>0.0002</td>
</tr>
<tr>
<td>$t=1/\lambda$</td>
<td>&quot;k+m&quot;</td>
<td>0.0026</td>
</tr>
<tr>
<td></td>
<td>POAG</td>
<td>0.0065</td>
</tr>
<tr>
<td></td>
<td>SOADP</td>
<td>0.0015</td>
</tr>
<tr>
<td>$t=2/\lambda$</td>
<td>&quot;k+m&quot;</td>
<td>0.0157</td>
</tr>
<tr>
<td></td>
<td>POAG</td>
<td>0.0279</td>
</tr>
<tr>
<td></td>
<td>SOADP</td>
<td>0.0108</td>
</tr>
</tbody>
</table>
As shown in TABLE II, compared with the “k+m” implementations, the proposed SOADP implementations reduced the failure rate by 10.7% to 42.3% when \( t = \frac{1}{\lambda} \) (mean time when the first fault may occur in FPGA), and reduced the failure rate by 9.98% to 31.2% when \( t = \frac{2}{\lambda} \) (mean time when 2 faults may occur in FPGA).

Since the POGA solutions are first order optimal solutions that maximize reliability, in some cases (e.g. implementations of b12) SOADP solutions are equivalent to POGA solutions. But in other cases (e.g. implementations of b22), SOADP solutions are obviously better than POGA solutions.

The memory costs of alternative configurations of the experimental implementations were listed in Table III. The memory costs were measured as the times of the memory area needed to store a single configuration. For example, it was supposed that it took 1 memory unit to store a single configuration of b22, thus it would take 9.428 memory units to carry out the SOADP implementation of b22. As shown, less or equal memory space was needed in SOADP than previous work.

<table>
<thead>
<tr>
<th>Solution</th>
<th>Memory Cost (times of the memory area for a single configuration)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>b22</td>
</tr>
<tr>
<td>&quot;k+m&quot;</td>
<td>9.5</td>
</tr>
<tr>
<td>POGA</td>
<td>9.428</td>
</tr>
<tr>
<td>SOADP</td>
<td>9.428</td>
</tr>
</tbody>
</table>

As shown in Fig. 4, the "k+m" implementation is more reliable than the POGA implementation when \( t < 4/\lambda \), however the proposed SOADP solution is more reliable than the above two implementations.

As shown in above experimental results, the proposed SOADP approach reduced failure rate and thus increased reliability with less time and memory costs compared with previous work.

Moreover, it is observed that if we relax Assumption 1 to \( A(q)+A(q-1)=1 \), more reliable partitions may be found in some cases.

VII. CONCLUSIONS AND FUTURE WORK

A second-order approximation domain partition method SOADP has been presented in this paper to solve the problem of assigning multiple alternative configurations and maximizing reliability in a FPGA based reconfigurable fault tolerant system. The SOADP approach helped to increase the reliability under given redundancy and given number of configurations. Compared with previous work, the proposed approach could offer a higher reliability and a lower configuration storage cost.

A drawback of the proposed approach is the time cost which is needed to solve the corresponding second-order optimization problem. The SOADP method helps to reduce the time cost of optimization objective calculation. However, when \( p \) grows, the total time cost may increase rapidly since the search space will be extremely enlarged with \( p \). We plan...
to refine assumption 1 and try to reduce the search space so that to reduce time cost. Fortunately, the time cost only depends on \( p \) and \( r \). In other words, once the SOADP solution under given \( p \) and \( r \) is found, it can be stored into library and be used to implement any circuits.

ACKNOWLEDGMENT

This research is supported in part by the open research program of Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences and in part by the Hi-Tech Research and Development Program of China (863) under grant No. 2007AA01Z107.

REFERENCES


