ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.13, No.11, 1–9

An integer programming method for constructing tightly coupled VLSI subarrays

Junyan Qian[a)], **Zhide Zhou, Lingzhong Zhao, Tianlong Gu, and Liang Chang**[b)]

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

a) *qjy2000@gmail.com*

b) *changl@guet.edu.cn*

Abstract: Minimizing the interconnection length between the processing elements (PEs) of VLSI arrays is beneficial to reduce the capacitance, power dissipation and dynamic communication cost. In this paper, a novel method, based on integer programming, for constructing tightly-coupled subarrays from the degradable VLSI arrays is presented, such that the target array has the minimum interconnection length. Compared with the state-of-the-art algorithms, the proposed method can guarantee that the interconnection length of the target array is minimum in row and column directions simultaneously. The performances of the proposed method are compared with previous studies and it indicates that the proposed method achieves better results in terms of total interconnection length.

Keywords: reconfiguration, integer programming, VLSI array

Classification: Integrated circuits

References

[1] R. M. Negrini, *et al.*: in *Fault-tolerance through Reconfiguration of VLSI and WSI Awards* (MIT Press, Cambridge, 1989).

[2] S.-Y. Kuo and I.-Y. Chen: IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **11** (1992) 1289 (DOI: 10.1109/43.170991).

[3] C. P. Low and H. W. Leong: IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **16** (1997) 1213 (DOI: 10.1109/43.662684).

[4] C. P. Low: IEEE Trans. Comput. **49** (2000) 553 (DOI: 10.1109/12.862215).

[5] M. Fukushi and S. Horiguchi: DFT (2004) 496 (DOI: 10.1109/DFTVS.2004.1347875).

[6] J. Wu and T. Srikanthan: J. Comput. Sci. Technol. **20** (2005) 647 (DOI: 10.1007/s11390-005-0647-3).

[7] J. Wu and T. Srikanthan: IEEE Trans. Comput. **55** (2006) 243 (DOI: 10.1109/TC.2006.43).

[8] J. Wu and T. Srikanthan: IEE Proc. Circ. Devices Syst. **153** (2006) 292 (DOI: 10.1049/ip-cds:20050273).

[9] J. Wu, *et al.*: Front. Comput. Sci. China **3** (2009) 315 (DOI: 10.1007/s11704-009-0032-4).

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

[10] J. Wu, *et al.*: IEEE Trans. Parallel Distrib. Syst. **25** (2014) 929 (DOI: 10.1109/TPDS.2013.114).
[11] Gurobi Optimization, Inc.: Gurobi Optimizer Reference Manual (2015) http://www.gurobi.com.

1 Introduction

With the rapid development of the techniques in very large scale integration (VLSI), hundreds to thousands of processing elements (PEs) are integrated on a single chip in a tightly coupled fashion to process massive amount of information in parallel. However, it is nearly impossible to guarantee that all components are fault-free throughout their product lifetime. Thus, fault tolerant techniques must be required to maintain the dependability in the systems on use. One way to achieve dependability in VLSI arrays is the degradation approach. In this approach, all PEs are treated in a uniform way and the fault tolerance is achieved by constructing a fault-free subarray utilizing as many fault-free PEs as possible [1, 2]. An optimal algorithm, called GCR, was proposed in [3] to find the maximum target (logical) array (MTA) that contains the selected rows. Low [4] proposed an efficient heuristic reconfiguration algorithm under the row and column rerouting constraint using GCR algorithm. Fukushi and Horiguchi proposed a hardware oriented heuristic reconfiguration approach in [5] based on the simple schemes of row and column rerouting.

In the most recent works of degradation approach, the researchers payed more attention to the reconfiguration of tightly-coupled maximum target array [6, 7, 8, 9, 10], which is an MTA with the minimum total interconnection length. However, the state-of-the-art works all focused on reducing the number of long interconnects (*nlis*) in logical columns of the MTA to decrease the routing cost, capacitance and dynamic power dissipation. There does not exist an exact algorithm or model to reducing the total length in both row and column directions.

In this paper, we proposed a novel method for constructing the tightly-coupled subarray by using the integer programming technique. A mathematical model for the reconfiguration of VLSI array is presented, which separates the problem and the solving algorithm. Compared with the state-of-the-arts, the proposed method can significantly reduce the total interconnection length in both row and column directions of the target array.

2 Preliminary

Let H indicate the *host array* (*physical* array) that is the original processor array after manufacturing and it may contain faulty PEs. The *target array (logical array)* T is defined as the subarray of H after reconfiguring, which contains no faulty PEs. The rows (columns) in H and T are called *physical rows (columns)* and *logical rows (columns)*, respectively.

The architecture and routing manners of the host array connected by four-port switches are shown in Fig. 1. Assuming $e_{i,j}$ represents the PE located in (i, j) of the host array H. Generally, two basic reroute schemes are used to reconfiguration, namely, *row bypass* and *column reroute*. As shown in Fig. 1, in row bypass

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

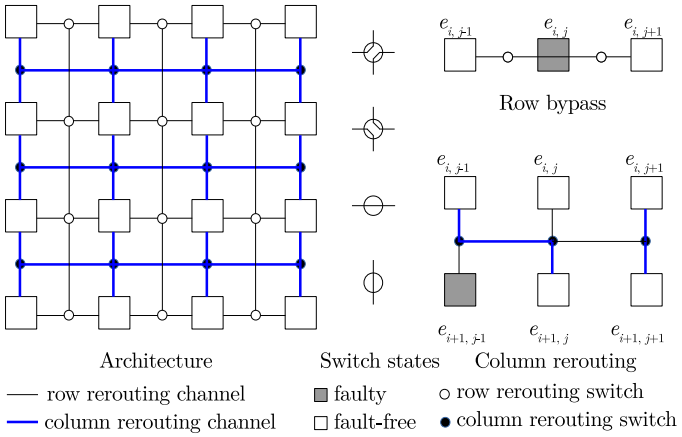


Fig. 1. Architecture and routing manners of a 4×4 array linked by switches.

scheme, $e_{i,j-1}$ can directly communicate with $e_{i,j+1}$ when $e_{i,j}$ is faulty. The data will bypass $e_{i,j}$ through an internal bypass link without being processed. In *the column rerouting* scheme, if $e_{i+1,j-1}$ is faulty, $e_{i,j}$ can directly connect to $e_{i+1,j'}$ with external switches, where $|j' - j| \leq d$, d is called *compensation distance* [3]. As same as in [7] and [10], d is also limited to 1. Throughout this paper, we use the *row bypass and column rerouting* constraint as the rerouting scheme.

As shown in Fig. 2, there are six possible types of link-ways for a target array [7], which can be classified into two classes based on the number of the switches used. One is called the *short interconnect*, which uses one switch to connect neighboring PEs; the other is called the *long interconnect*, which uses two switches. In Fig. 2, (a) and (d) are short interconnects, while the others are long interconnects. Suppose that the length of a short interconnect is d and the width of a PE is w. Thus, the length of a long interconnect is $d/2 + w/2 + d + w/2 + d/2$, i.e., $2d + w$ [10].

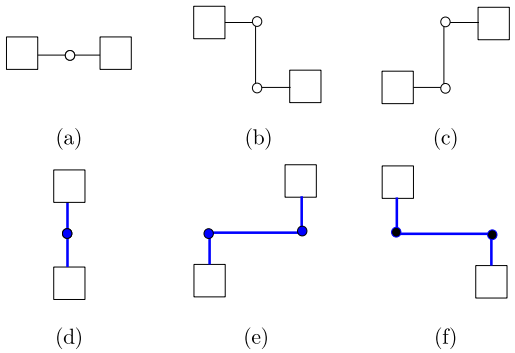


Fig. 2. Short and long interconnects.

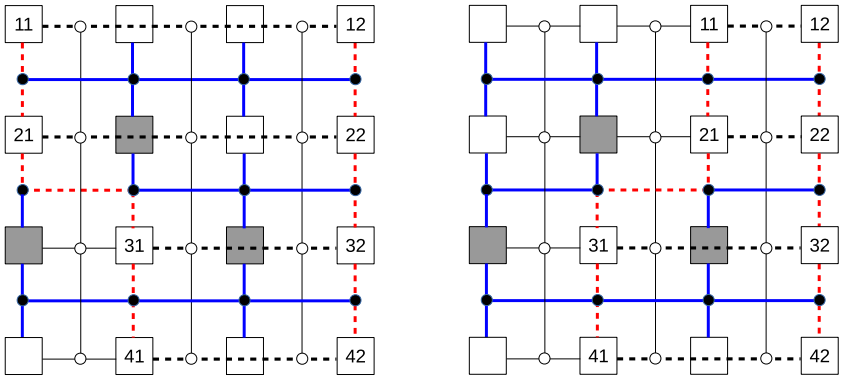
Definition 1. *An MTA with the minimum total interconnection length is called the tightly-coupled target array (TCTA).*

Problem $\mathcal{P}$. *Given an host array of size $m \times n$, find a TCTA that contains the selected rows under the constraint of row bypass and column rerouting scheme.*

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

Fig. 3 shows two 4×2 target arrays on 4×4 host array with 3 faulty PEs. The two digit number in a box indicates the location of a PE in the target array, the first and second digit represent the number of the logical rows and columns, respectively. The dotted red (black) line stands for the path that connects each PEs of a logical column (row).

Obviously, the total interconnection length of Fig. 3(b) is shorter than that of Fig. 3(a). In Fig. 3(a), the MTA has two logical columns and four logical rows with 15 short interconnects, one long interconnect and 6 bypass links, its total interconnection length is $17d + 7w$ including 15 short interconnects ($15d$), one long interconnect ($2d + w$) and 6 bypass links ($6w$) used to bypass 4 unused PEs and 2 faulty PEs. While the total interconnection length of the target array in Fig. 3(b) is $13d + 3w$ despite that it has the same number of logical columns and long interconnects, this is because the target array in Fig. 3(b) has shorter row interconnection length. Clearly, minimizing the total interconnection length in the target array without loss of harvest leads to lesser routing cost, capacitance, and dynamic power dissipation.



(a) MTA (b) TCTA

Fig. 3. Example of MTA and TCTA.

3 The proposed model

In this section, we propose an efficient method based on integer programming to construct a tightly-coupled MTA. Given an $m \times n$ host array, without loss of generality, we assume that the target array contains the selected rows $R_1, R_2, \cdots, R_m$. Let B_l and B_r be two logical columns passing through each physical row of the $m \times n$ host array. We say that $B_l < B_r$ (or $B_l = B_r$) if the ith PE in B_l lies to the left of (or is identical to) the ith PE in B_r, for $1 \leq i \leq m$. If there does not exist any identical PEs between B_l and B_r, we say that B_l and B_r are independent. In this paper, $\mathcal{A}[B_l, B_r]$ indicates the area that consists of the PEs bounded by B_l and B_r (including B_l and B_r). The logical columns B_l and B_r are called the left boundary and the right boundary of the area, respectively.

Definition 2. *A logical column is called the local optimal column related to B_l and B_r if and only if it is the logical column of the minimum number of the long interconnects in $\mathcal{A}[B_l, B_r]$* [5].

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

Suppose that B_l is the ith logical column generated by GCR [3] with the left-to-right manner, B_r is the $(k-i+1)$th logical column generated by GCR with the right-to-left manner (where k is the total number of logical column), and Y_i is the ith local optimal column, for $1 \le i \le k$. Thus, the following lemma indicates the relationship among B_l, B_r, and Y_i [7].

Lemma 1. *$B_l \le B_r$ and they are not independent of each other; $B_l \le Y_i \le B_r$ and the area $\mathcal{A}[B_l, B_r]$ is the largest area to produce Y_i for $1 \le i \le k$.*

Generally, an MTA consists of many logical columns $\mathcal{Y}_1, \mathcal{Y}_2, \cdots, \mathcal{Y}_k$. According to Lemma 1, it is easy to find that the length of a logical column $\mathcal{Y}_i$ can be regarded as a function $\mathcal{L}(\mathcal{Y}_i)$ in $\mathcal{A}[B_l, B_r]$. Assume that $P_{j,i}$ and $P_{j+1,i}$ are two PEs of the ith logical column $\mathcal{Y}_i$ in the jth row R_j and the $j+1$ row R_{j+1}, respectively, i.e., $P_{j,i} \in R_j \cap \mathcal{A}[B_l, B_r]$, $P_{j+1,i} \in R_{j+1} \cap \mathcal{A}[B_l, B_r]$, for $1 \le j \le m$. Let $col(P_{j,i})$ indicate the physical column index of $P_{j,i}$. Thus, if we set the length of a short interconnects and a long interconnects to 0 and 1, respectively; then the interconnection length between $P_{j,i}$ and $P_{j+1,i}$ can be defined as $L(P_{j,i}, P_{j+1,i}) = |col(P_{j,i}) - col(P_{j+1,i})|$ under the constraint that the compensation distance is set to 1 [7, 10], i.e., $|col(P_{j,i}) - col(P_{j+1,i})| \le 1$. Hence, the interconnection length of the ith logical column $\mathcal{Y}_i$ can be defined as $\mathcal{L}(\mathcal{Y}_i) = \sum_{j=1}^{m-1} L(P_{j,i}, P_{j+1,i})$, which represents the number of long interconnects of $\mathcal{Y}_i$.

Similarly, we can define the length of the logical row. Let $P_{j,i}$ and $P_{j,i+1}$ be two PEs of the logical column $\mathcal{Y}_i$ and $\mathcal{Y}_{i+1}$ in the jth row, respectively. The interconnection length between $P_{j,i}$ and $P_{j+1,i}$ is defined as $L(P_{j,i}, P_{j,i+1}) = |col(P_{j,i}) - col(P_{j,i+1})|$. Thus, the interconnection length of the jth logical row $\mathcal{R}_j$ is defined as $\mathcal{L}(\mathcal{R}_j) = \sum_{i=1}^{k-1} L(P_{j,i}, P_{j,i+1})$.

Assume that the boolean variable v represents the PE in $R_j \cap \mathcal{A}[B_l, B_r]$ as each fault-free PE is used to form a logical column, or not. Thus, $P_{j,i} = \sum v$ and $\sum v = 1$ as each logical column contains exactly one fault-free PE from each of the rows, for each $v \in R_j \cap \mathcal{A}[B_l, B_r]$. However, the areas of each logical columns always overlap each other such that a PE is represented by many variables. Thus, we need to ensure that at most one of these variables can be true. If a PE is represented by δ variables, i.e., $v_1, v_2, \cdots, v_\delta$, then $v_1 + v_2 + \cdots + v_\delta \le 1$. Meanwhile, in order to avoid that the logical columns cross each other, we add a constraint $P_{j,i+1} - P_{j,i} > 0$ such that the logical columns has a strictly partial order relation as $\mathcal{Y}_1 < \mathcal{Y}_2 < \cdots < \mathcal{Y}_k$.

Therefore, the objective function can be summarize as follows:

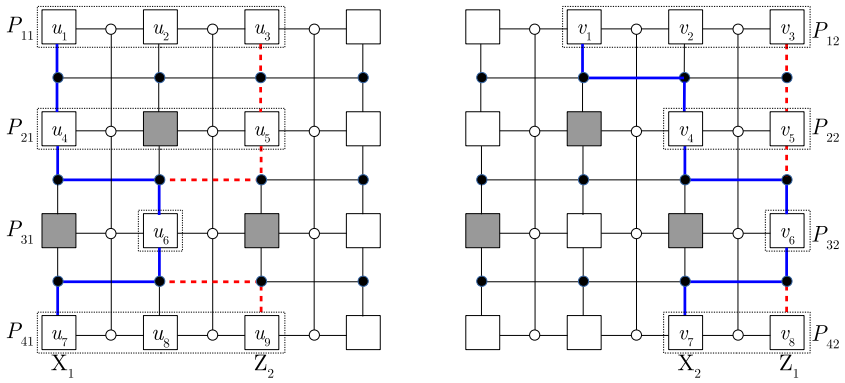
$$\min\left\{\sum_{i=1}^{k} \mathcal{L}(\mathcal{Y}_i) + \sum_{j=1}^{m} \mathcal{L}(\mathcal{R}_j)\right\}. \tag{1}$$

However, formula (1) will lead to increase the number of long interconnects as the value of $\sum_{j=1}^{m} \mathcal{L}(\mathcal{R}_j)$ is far greater than that of $\sum_{i=1}^{k} \mathcal{L}(\mathcal{Y}_i)$. Meanwhile, minimizing the number of long interconnects is more conducive to reduce routing cost, capacitance, and dynamic power dissipation [7]. Thus, we need to increase the weight of each logical column. As it is known, there are at most $m-1$ long interconnects of a

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

logical column, which makes the weight of each $\mathcal{L}(\mathcal{Y}_i)$ to be set as m. Therefor, formula (1) can be modified as follows:

$$\min\left\{\sum_{i=1}^{k} m \cdot \mathcal{L}(\mathcal{Y}_i) + \sum_{j=1}^{m} \mathcal{L}(\mathcal{R}_j)\right\}. \tag{2}$$



(a) First column. (b) Second column.

Fig. 4. Example of proposed method.

The idea of the proposed method is illustrated in Fig. 4, which is derived from Fig. 3. In this example, the target array produced by GCR with the left-to-right manner consists of the logical columns X_1, X_2 and the target array produced by GCR with the right-to-left manner consists of the logical columns Z_1, Z_2. The boolean variables in the box represent the current PE. According to the above analysis, the formulas of first column can be described as follows:

$$\begin{array}{l|r|l}
P_{11} = 1 * u_1 + 2 * u_2 + 3 * u_3 & u_1 + u_2 + u_3 = 1 & |P_{11} - P_{21}| \le 1 \\
P_{21} = 1 * u_4 + 3 * u_5 & u_4 + u_5 = 1 & |P_{21} - P_{31}| \le 1 \\
P_{31} = 2 * u_6 & u_6 = 1 & |P_{31} - P_{41}| \le 1. \\
P_{41} = 1 * u_7 + 2 * u_8 + 3 * u_9 & u_7 + u_8 + u_9 = 1 &
\end{array}$$

Notably, for conveniently expressing the length between $P_{j,i}$ and $P_{j+1,i}$, we encode $P_{j,i} = col(u_1) \cdot u_1 + col(u_2) \cdot u_2 + \cdots$, which is different with previous analysis (i.e., $P_{j,i} = \sum v$). Hence, the length of the first column can be encoded as follows:

$$\begin{aligned}
&L(P_{11}, P_{21}) = |P_{11} - P_{21}| \\
&L(P_{21}, P_{31}) = |P_{21} - P_{31}| \\
&L(P_{31}, P_{41}) = |P_{31} - P_{41}| \\
&\mathcal{L}(\mathcal{Y}_1) = L(P_{11}, P_{21}) + L(P_{21}, P_{31}) + L(P_{31}, P_{41}).
\end{aligned}$$

Similarly, the formulas of the second column can be described as follows.

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

$$\begin{array}{l|l|l}
P_{12} = 2 * v_1 + 3 * v_2 + 4 * u_3 & v_1 + v_2 + v_3 = 1 & |P_{12} - P_{22}| \le 1 \\
P_{22} = 3 * v_4 + 4 * v_5 & v_4 + v_5 = 1 & |P_{22} - P_{32}| \le 1 \\
P_{32} = 4 * v_6 & v_6 = 1 & |P_{32} - P_{42}| \le 1 \\
P_{42} = 2 * v_7 + 3 * v_8 & v_7 + v_8 = 1 &
\end{array}$$

$$L(P_{12}, P_{22}) = |P_{12} - P_{22}|$$
$$L(P_{22}, P_{32}) = |P_{22} - P_{32}|$$
$$L(P_{32}, P_{42}) = |P_{32} - P_{42}|$$
$$\mathcal{L}(\mathcal{Y}_2) = L(P_{12}, P_{22}) + L(P_{22}, P_{32}) + L(P_{32}, P_{42}).$$

For the PEs that are represented by many variables, we add the following constraints to ensure that each PE can only belong to exactly one logical column.

$$u_2 + v_1 \le 1,\ u_3 + v_2 \le 1,\ u_5 + v_4 \le 1,\ u_9 + v_7 \le 1.$$

The length of each logical row can be simply encode as $P_{j,i+1} - P_{j,i}$ as there are only two logical columns, i.e.,

$$\mathcal{L}(\mathcal{R}_1) = P_{12} - P_{11}, \quad \mathcal{L}(\mathcal{R}_2) = P_{22} - P_{21},$$
$$\mathcal{L}(\mathcal{R}_3) = P_{32} - P_{31}, \quad \mathcal{L}(\mathcal{R}_4) = P_{42} - P_{41}.$$

Thus, the objective is to minimize the total interconnect length of the target array utilized:

$$\min\{4 \cdot \mathcal{L}(\mathcal{Y}_1) + 4 \cdot \mathcal{L}(\mathcal{Y}_2) + \mathcal{L}(\mathcal{R}_1) + \mathcal{L}(\mathcal{R}_2) + \mathcal{L}(\mathcal{R}_3) + \mathcal{L}(\mathcal{R}_4)\}.$$

4 Experimental results

To evaluate the efficiency of the proposed method, we have developed a program for automatically generating the formulas of the proposed model and the Gurobi Optimizer [11] was called to solve it. Notably, all the options of Gurobi Optimizer were set as default. Meanwhile, the state-of-art algorithm reported in [10] (denote as ALG14 in this paper) has been implemented in *C* program language for performance comparison. In order to make a fair comparison, we maintain the same assumptions as in [10]. The fault density in the host arrays in varied from 1% to 5% for a comprehensive comparison. All of them were tested and compared with each other on the same random input instances. A Windows PC with an Intel(R) Xeon(R) E5607 2.27 GHz CPU and 4.0 GB of memory was used to run the experiments.

In Table I, the term *IP* indicates the proposed method and *row_len* represents the total row length of the target array. The data are collected for host arrays of different sizes from 24×24 to 40×40 and the values of *nlis* (the number of long interconnects) and *row_len* are with the decimal points rounded off to the nearest integer in all cases.

From Table I, it is clear that the proposed method significantly reduces the interconnect length of the target array. For example, for ALG14, the values of *nlis* and *row_len* are 210 and 1,545 for an array of 40×40 and fault densities 5%, respectively. But, for the proposed method, the values of *nlis* and *row_len* become

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

Table I. The comparison of the algorithms ALG14 and IP for random faults of uniform distribution, averaged over 20 random instances.

Host array		Target array	Performance							
Size	Fault	Size	*nlis*			*row_len*			*time*	
$m \times n$	(%)	$m \times k$	ALG14	IP	*imp* (%)	ALG14	IP	*imp* (%)	ALG14 (ms)	IP (s)
	1	24×23	25	25	0	548	546	0.28	0.036	0.133
24×24	3	24×21	49	46	7.65	546	544	0.42	0.036	0.898
	5	24×20	79	73	7.78	545	541	0.60	0.038	1.207
	1	32×30	58	53	8.74	991	988	0.30	0.065	1.001
32×32	3	32×28	112	98	12.88	981	977	0.39	0.069	9.682
	5	32×27	147	126	13.89	977	968	0.95	0.067	12.378
	1	40×38	90	86	4.56	1,552	1,548	0.28	0.105	4.693
40×40	3	40×36	181	157	13.77	1,547	1,539	0.48	0.093	12.516
	5	40×34	210	179	14.83	1,545	1,535	0.65	0.095	38.019

179 and 1,535, the improvements on *nlis* and *row_len* are 14.83% and 0.65%, respectively. As shown in Table I, both improvements on *nlis* and *row_len* are increased with the increase of the fault density and the size of host array. However, the improvement on *nlis* is more significant than that of *row_len*. This is because, on the one hand, it has the highest priority to reduce the number of long interconnects in this paper. On the other hand, in the low fault density, the row length of the target array is impossible to largely decrease compared with the host array, since the columns of the target array does not be dramatically reduced related to the host array. For a 40×40 host array with 1% faults, for example, the size of the target array is 40×38, only two columns are reduced and the target array is tightly-coupled compared with the host array.

However, the running time of the proposed method is too long compared with ALG14. This is because the proposed method heavily depends on the integer programming solver. In addition, the number of the variables and formulas increases rapidly with the increase of the fault density and the size of host array. From Table I, it is evident that the bigger the fault density and the size of host array, the longer the running time of the proposed method. Thus, the proposed method may not be able to meet the real-time requirement of the system, and the heuristic methods, such as the ALG14 algorithm, can be used to meet the system requirement. However, for the system that requires lower power dissipation and communication costs instead of the high real-time requirement, the proposed method should be the best choice as it can guarantee that the total interconnection length of the resultant logical array is minimum.

5 Conclusions

This paper proposed an integer programming method for the reconfiguration of degradable VLSI array, resulting in a tightly-coupled target array with the minimum interconnection length. The proposed method regards the length of each logical rows and columns as a function about the fixed integer variables, such that the

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016

tightly-coupled target array can be constructed by utilizing the efficient techniques of integer programming. Experimental results show that the proposed algorithm is capable to produce more tightly-coupled target array than the state-of-the-arts. The improvements in reducing the long interconnects and the length of logical rows are up to 14.83% and 0.68% for the target array derived from the 40×40 host array with 5% faults, respectively.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant (61262008, 61363030, 61562015, U1501252, 61572146), the High Level Innovation Team of Guangxi Colleges and Universities and Outstanding Scholars Fund, Guangxi Natural Science Foundation of China under grant (2014GXNSFAA118365, 2015GXNSFDA139038), Innovation Project of GUET Graduate Education (YJCXS201537), Guangxi Key Laboratory of Trusted Software Focus Fund, Program for Innovative Research Team of Guilin University of Electronic Technology.

DOI: 10.1587/elex.13.20160359
Received April 8, 2016
Accepted May 9, 2016
Publicized May 23, 2016
Copyedited June 10, 2016