A New K-Way Partitioning Approach for Multiple Types of FPGAs

Bernhard M. Riess, Heiko A. Giselbrecht, and Bernd Wurth
Institute of Electronic Design Automation
Technical University of Munich, 80290 Munich, Germany

Abstract

This paper considers the problem of partitioning a large, technology mapped circuit onto multiple FPGA devices of a specified device library. We propose an iterative three-step approach applying an analytical embedding technique, initial partitioning, and a k-way ratio cut improvement procedure. We successfully partitioned the ACM/SIGDA XILINX FPGA Benchmark circuits obtaining feasible design solutions with lower total dollar costs than previous methods. Moreover, our approach simultaneously assigns the FPGAs to physical locations on the FPGA board.

1 Introduction

In order to reduce time to market, FPGAs have become an important technology for system implementation and rapid prototyping. However, since high-level synthesis methods become widely accepted, the complexity of electronic systems is rapidly increasing. Thus, the entire design often cannot be realized on a single FPGA but the entire circuit has to be partitioned onto a set of single type or multiple type FPGAs. A partitioning approach for this application has to consider the special requirements of the given FPGA device library, which are constraints on the number of configurable logic blocks (CLBs) and on the number of input/output blocks (IOBs) of the devices.

K-way partitioning over multiple FPGAs can be performed before or after technology mapping. If the gate level netlist is split before technology mapping, estimation of chip utilization and routability is difficult without the exact count of the CLBs and IOBs. Thus, these estimations are made conservatively to ensure successful execution of placement and routing. After technology mapping, accurate CLB and IOB numbers are known, and the CLB and IOB count of each obtained partition can be compared to the CLB and IOB constraints of the given device library. Hence, we attack the problem of partitioning a technology mapped circuit.

Several approaches have been proposed to partition technology mapped circuits onto multiple type FPGAs. Kužnar et al. [1] proposed an algorithm to partition a mapped netlist onto multiple device types to minimize total device cost. This method recursively applies a modified min-cut technique [2]. In subsequent work, Kužnar et al. [3] introduced functional replication to reduce the number of required IOBs and therefore minimized total device cost. They presented excellent results. Woo and Kim [4] proposed another modification of the min-cut procedure for k-way partitioning, which tries to satisfy the constraints on the number of CLBs and IOBs of the FPGAs in the device library. Chou et al. [5] addressed the problem of partitioning a circuit onto a set of single type FPGAs such that the number of FPGAs is minimized. Their algorithm uses "local ratio-cut" clustering to reduce the circuit complexity and then derives a disjoint partition using a set covering approach. Recently, Huang and Kahng [6] proposed an approach that incorporates vertex ordering, clustering, and dynamic programming to achieve feasible design solutions.

In this paper, we address the problem of partitioning a circuit onto a set of multiple types of FPGAs of a given library of device types. Our approach iteratively bipartitions the given netlist until k subcircuits are obtained. The proposed k-way partitioning approach applies the following three steps in each iteration. First, we calculate a 2-dimensional embedding for all cells by minimizing a linear objective function. Next, each partition containing CLBs to be covered by more than one FPGA is initially partitioned by a rectangular cut which is based on the 2-dimensional embedding and uses a modified ratio cut metric. In the final step, the number of IOBs of all partitions are reduced applying an improved k-way ratio cut procedure. Reducing the number of IOBs of all partitions allows to use cheaper devices for the partitions and thus minimizes the total dollar costs required to implement the entire design. These three steps are repeated, until k partitions are obtained.

We successfully partitioned 13 circuits of the ACM/-SIGDA XILINX FPGA Benchmark Suite obtaining feasible design solutions with 2.3% lower total dollar costs on an average than previous methods. Compared to the absolute lower total cost bound [1] of this Benchmark Suite, our approach yields results on the average 21.5% closer to the absolute minimum than state of the art methods [3].

The remainder of our paper is organized as follows. The next section gives some basic preliminaries. Section 3 describes our new k-way partitioning approach. In Section 4, results of the ACM/SIGDA XILINX FPGA Benchmark circuits are presented and discussed.

2 Preliminaries

We model the circuit as a hypergraph $H = (V_{pio} \cup V_{clb}, E')$, where the vertices V_{pio} represent the set of primary inputs and primary outputs of the entire design, the vertices V_{clb} represent the set of CLBs, and the hyperedges E' represent the nets. A subcircuit is denoted by a subhypergraph $H_p = (V_{iob_p} \cup V_{clb_p}, E'_p)$ with the set of IOBs V_{iob_p}, the subset of CLBs $V_{clb_p} \subset V_{clb}$, and the set of hyperedges E'_p representing the nets in partition p. Partitioning without

cell replication implies an assignment of each vertex in V_{clb} to exactly one of a set of k subcircuits.

We address the problem of partitioning a circuit onto a set of multiple types of FPGAs, which means to create a feasible partitioning of the given circuit H onto a set of devices from a given FPGA library such that the total device cost is minimized.

A typical FPGA device library is shown in Table 1. It gives the XILINX XC3000 device family, which is used for the ACM/SIGDA XILINX FPGA Benchmark Suite and has been widely used in previous work [1,3,6].

i	Device X_i	#CLBs	#IOBs	$\frac{cost}{N\$}$	$\frac{cost}{\#CLBs}$
1	XC3020x-x	64	64	1.00	0.0156
2	XC3030x-x	100	80	1.36	0.0136
3	XC3042x-x	144	96	1.84	0.0128
4	XC3064x-x	224	120	3.15	0.0141
5	XC3090x-x	320	144	4.83	0.0151

Table 1: XILINX XC3000 device library

Table 1 presents for each device X_i the number of CLBs contained, the number of available IOBs, the price, normalized to the cost of the smallest device, and the cost of one CLB.

We call a subcircuit H_p feasible on a device X_i if $|V_{clb_p}| \leq \#\text{CLBs}(X_i)$ and $|V_{iob_p}| \leq \#\text{IOBs}(X_i)$ hold. A k-way partitioning is called feasible if each partition p is feasible on one device $X_i, i \in \{1, ..., 5\}$ of the given device library.

Considering the implementation of the entire circuit on a FPGA board, we focus on the following target architecture. The FPGA devices are regularly arranged in an $r \times c$ grid on the FPGA board with r rows and c columns of FPGA devices. We assume that the primary inputs and primary outputs have given locations on the board border. In Figure 1 a 2×3 grid target architecture is shown.

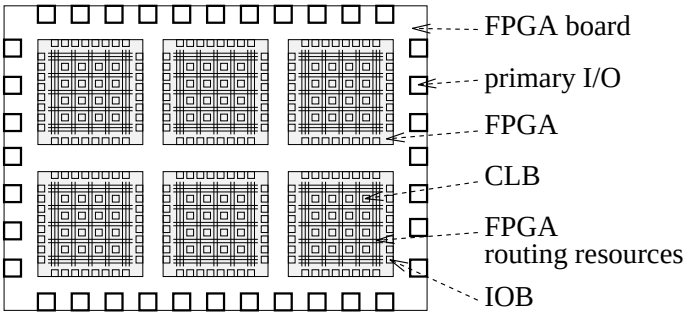

Figure 1: 2×3 grid FPGA board

The FPGAs used for one design don't have to be of the same type, but of one of the five types given in the device library. The number of required FPGAs $k = r \cdot c$ to determine an appropriate target architecture can be restricted to:

$$\left\lceil \frac{|V_{clb}|}{\#CLBs(X_5)} \right\rceil \leq k \leq \left\lceil \frac{|V_{clb}|}{\#CLBs(X_1)} \right\rceil \wedge \left\lceil \frac{|V_{pio}|}{\#IOBs(X_5)} \right\rceil \leq k \quad (1)$$

Considering this inequality constraint one can determine k and an appropriate target architecture with preferably quadratic grid ($r \approx c$). In the following, we will use the term *CLB* and *cell* interchangeably.

3 K-Way Partitioning for FPGAs

3.1 Outline of the Procedure

To solve the k-way partitioning problem, we propose an iterative three-step approach. Each iteration of this approach is called a level. On all levels, we calculate a 2-dimensional embedding for all cells and perform an initial 2-way partitioning of all existing partitions which have to be partitioned further. The number of required IOBs of all obtained partitions is reduced by a new k-way ratio cut procedure with an objective function specialized to FPGA partitioning.

After calculating the 2-dimensional embedding, the cells are initially partitioned by a rectangular cut. The possible cut directions (horizontal or vertical) and cut positions of this cut are selected according to the matrix grid of the addressed target architecture such that each partition contains CLBs to be covered by an integer number of FPGAs. Hence, our approach is driven by the grid of the addressed target architecture. All cells to the left (top) of a vertical (horizontal) cut line are assigned to the left (top) partition and all cells to the right (bottom) of the cut line are assigned to the right (bottom) partition. After this initial partitioning, the solution quality in terms of required IOBs is improved by applying the k-way ratio cut method. In doing this, cells can be moved from their initial partition to any adjacent partition.

On the subsequent levels, the 2-dimensional embedding is refined while the partitions of the previous levels are maintained. Subsequently, the remaining partitions are divided again and the obtained partitioning is improved again by the k-way ratio cut method. When the final level is finished, each partition corresponds to one device on the FPGA board.

3.2 Calculating the Embedding

Past partitioning approaches which are based on analytical techniques use one or several eigenvectors to calculate an embedding of the cells [7,8]. These spectral approaches minimize a quadratic objective function, which models squared wire length. Recently, it has been shown that minimizing a linear objective function yields improved results in placement [9,10] as well as in partitioning [11] since the linear model of the wire length is closer to reality. Therefore, we adopt this strategy and calculate a 2-dimensional embedding of the cells by minimizing a linear objective function.

To calculate the embedding, the hypergraph $H = (V_{pio} \cup V_{clb}, E')$ is transformed into a graph $G = (V_{pio} \cup V_{clb}, E)$ by mapping each hyperedge in the set E' into a set of binary edges. To perform this mapping, we apply the well-known clique model [12]. Each hyperedge consisting of h vertices is represented by a complete graph with edge weights equal to $1/h$. The obtained graph G is described by an $n \times n$ adjacency matrix $\mathbf{M} = [m_{ij}]$, where $n = |V_{pio}| + |V_{clb}|$. The matrix elements m_{ij} are calculated as the sum of the edge weights of all edges connecting the vertices v_i and v_j. If e_{ii} denotes the degree of vertex v_i (i.e., the sum of the weights of all edges incident to vertex v_i) and $e_{ij} = 0$ for all $i \neq j$, we obtain the $n \times n$ diagonal degree matrix $\mathbf{E}$. Now the Laplacian $\mathbf{C}$ is given by $\mathbf{C} = \mathbf{E} - \mathbf{M}$.

Since the objective function we want to minimize can be separated in x- and y-direction, we consider only the x-component in the following. The vector $\mathbf{x} = [x_1, \ldots, x_i, x_j, \ldots, x_n]^T \in \mathbb{R}^n$ contains a coordinate for each cell. We formulate a linear objective function:

$$\phi(\mathbf{x}) = \mathbf{x}^T \mathbf{C} \mathbf{x} \tag{2}$$

To minimize a linear objective function we adapt matrix $\mathbf{C}$ during the optimization process according to the adaption scheme of Sigl et al. [9]. Since fixed primary input and output cells on the FPGA board border shall be considered, the coordinates of the fixed primary input and primary output cells in the vector $\mathbf{x}$ are constant and some quadratic terms in equation (2) give linear or constant terms. Omitting the constant terms, we rewrite equation (2) as:

$$\phi_p(\mathbf{x}) = \mathbf{x}^T \mathbf{C}^* \mathbf{x} + \mathbf{d}^T \mathbf{x} \tag{3}$$

Matrix $\mathbf{C}^*$ is equivalent to matrix $\mathbf{C}$ except that the matrix elements of the linear and constant terms are removed. In the subsequent partitioning steps, the cells are assigned to partitions. To consider these partitions during the calculation of the next embedding, the center of gravity of all cells of each partition is fixed to the center of the partition. The centers of the partitions on the l^{th} level form the constraints $\mathbf{A}_l \mathbf{x} = \mathbf{b}_l$ on the logic cells, with the vector $\mathbf{b}_l$ of the center coordinates of the partitions and the matrix $\mathbf{A}_l = [a_{pi}]_l$ containing the assignment of cell v_i to partition p. The matrix elements a_{pi} are given by:

$$a_{pi} = \begin{cases} 1/|V_{clb_p}| & \text{if } v_i \in V_{clb_p} \\ 0 & \text{otherwise} \end{cases}$$

The objective function (3) and this constraint form our programming problem:

$$\underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \ \{\phi_p(\mathbf{x}) = \mathbf{x}^T \mathbf{C}^* \mathbf{x} + \mathbf{d}^T \mathbf{x} \mid \mathbf{A}_l \mathbf{x} = \mathbf{b}_l\} \tag{4}$$

By solving this problem for x- and y-coordinates, we obtain a 2-dimensional embedding of all cells. This problem formulation considers the connections of the CLBs to the fixed primary input and primary output cells and the assignment of cells to partitions while minimizing the linearly modeled wire length. Since the embedding of all cells is calculated simultaneously on all levels, even cells of different partitions can influence each other.

3.3 Initial Partitioning

After the embedding is calculated, each partition containing cells to be covered by more than one FPGA is divided into two parts. Driven by the matrix grid of the addressed FPGA board, this is done in the direction (horizontal or vertical) in which the partition has to be further partitioned, whereby both cut directions may be possible. Possible cut regions in a partition are located between the rows or columns of the devices on the FPGA board. The centers of these cut regions are defined by the cut lines which would lead to absolutely equisized partitions at the end of the procedure.

Since FPGA devices of different size are available, tolerance intervals allowing some deviation from equisized partitions can be specified. If $|V_{clb}|$ CLBs have to be mapped on $k = r \cdot c$ FPGAs, the average number (AVG) of CLBs per device equals:

$$AVG = \frac{|V_{clb}|}{k} \tag{5}$$

To ensure that a partition contains at most as many cells as available CLBs on the largest device X_5, the tolerance σ is set to the relative deviation of the number of available CLBs on a device of type X_5 from the average number of CLBs per device, limited by 1:

$$\sigma = \min\left(\frac{\#CLBs(X_5) - AVG}{AVG}, 1\right) \tag{6}$$

Multiplying the relative tolerance σ by the actual average number of CLBs of the FPGAs covering partition p $\frac{|V_{clb_p}|}{\#FPGAs(p)}$ we obtain the absolute width of the cut regions 2δ. The number of FPGAs covering partition p is denoted by $\#FPGAs(p)$.

In Figure 2 a 2×3 target architecture with 6 FPGAs is given. Imagine the set of cells spread out all over the entire FPGA board. On the first level, the circuit may either be initially partitioned in horizontal or vertical direction. There exist one horizontal cut region and two vertical cut regions. Overall, we obtain 3 cut regions shown in dark gray. Dashed lines denote the centers of the cut regions which would lead to absolutely equisized partitions.

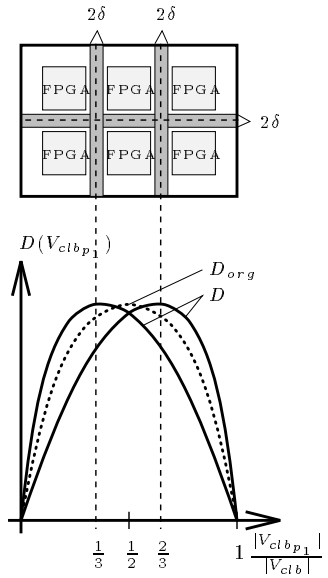

Figure 2: Cut regions and the new denominator functions

At one cut position in these cut regions the set of nodes V_{clb_p} of partition p with $G_p = (V_{iob_p} \cup V_{clb_p}, E_p)$ is divided into two disjoint subsets $V_{clb_{p_1}} \subset V_{clb_p}$, and $V_{clb_{p_2}} = V_{clb_p} - V_{clb_{p_1}}$, with $V_{clb_{p_1}} \neq \emptyset$, $V_{clb_{p_2}} \neq \emptyset$. (On the first level V_{clb_p} equals V_{clb}.) $C_{p_1 p_2}$ denotes the number of nets connecting the partitions p_1 and p_2. The determination of the actual cut position is based on the ratio cut metric by computing a ratio cut diagram for each cut region.

According to Wei and Cheng [13] the ratio cut (RC) is defined by:

$$RC = \frac{C_{p_1 p_2}}{|V_{clb_{p_1}}| \cdot |V_{clb_{p_2}}|} \tag{7}$$

This original ratio cut objective (7) favors balanced partition sizes. This is achieved by the parabola

$$D_{org}(V_{clb_{p_1}}) = |V_{clb_{p_1}}| \cdot |V_{clb_{p_2}}| = |V_{clb_{p_1}}| \cdot (|V_{clb_p}| - |V_{clb_{p_1}}|)$$
$$\text{with} |V_{clb_p}| = |V_{clb_{p_1}}| + |V_{clb_{p_2}}| = const. \tag{8}$$

in the denominator, which reaches its maximum when the partition sizes are absolutely equal. The dotted curve in Figure 2 gives this denominator function.

In our approach however, intermediate partitions do not have to meet this relation but may be quite unbalanced. Using the original ratio cut objective would favor cut regions with size relations close to 1:1. To provide equal chances for all possible cut regions, we move the vertex

of the parabola in the denominator to the point defined by the desired size ratio $\alpha_{12} = \frac{\#FPGAs(p_1)}{\#FPGAs(p)}$ for each cut region.

We construct a denominator function $D(V_{clb_{p_1}})$ consisting of 2 parabola branches with one common point and gradient in the vertex at $\alpha_{12} \cdot |V_{clb_p}|$. Considering these constraints, we get the following continuously differentiable denominator:

$$D(V_{clb_{p_1}}) = \begin{cases} \frac{|V_{clb_p}| \cdot |V_{clb_{p_1}}|}{2\alpha_{12}} - \frac{|V_{clb_{p_1}}|^2}{4\alpha_{12}^2} \\ \qquad \text{if} \quad 0 < |V_{clb_{p_1}}| \leq \alpha_{12} \cdot |V_{clb_p}| \\ \frac{|V_{clb_p}|^2(1-2\alpha_{12}) + 2\alpha_{12}|V_{clb_p}| \cdot |V_{clb_{p_1}}| - |V_{clb_{p_1}}|^2}{4(\alpha_{12}-1)^2} \\ \qquad \text{if} \quad \alpha_{12} \cdot |V_{clb_p}| < |V_{clb_{p_1}}| < |V_{clb_p}| \end{cases} \tag{9}$$

The denominator functions $D(V_{clb_{p_1}})$ for both vertical cut regions of the 2×3 example are shown in Figure 2. By substituting the original denominator by this denominator, we obtain the adapted ratio cut objective (ARC):

$$ARC = \frac{C_{p_1 p_2}}{D(V_{clb_{p_1}})} \tag{10}$$

Finally, we identify the minimum adapted ratio cut value within all cut regions of both possible cut directions and initially partition the set of cells V_{clb_p} in two subsets at this optimal cut. Figure 3 presents the cut regions, two adapted ratio cut diagrams, and the minimum adapted ratio cut defining the optimal cut position for the vertical cut regions of the example shown in Figure 2.

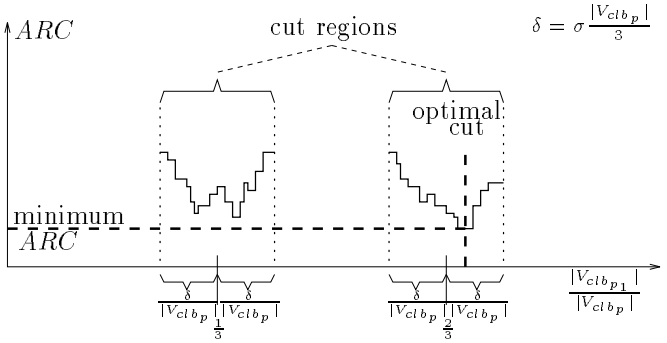

Figure 3: Cut regions, adapted ratio cut diagrams, minimum adapted ratio cut, and the optimal cut

3.4 Iterative Partitioning Improvement

3.4.1 The K-Way Ratio Cut Approach

On all levels of our approach, the number of required IOBs of each partition obtained after initial partitioning is reduced by applying our new k-way ratio cut algorithm. This approach combines the two-way ratio cut [13] and the Quadrisection algorithm [14] with special requirements for multiple FPGA partitioning.

The algorithm starts by selecting the partition with the largest number of required IOBs per device as central partition. All partitions adjacent to the central partition are called neighbor partitions. Cells can either be moved from the central partition to any neighbor partition or vice versa in the same sequence.

As proposed for the two-way ratio cut [13], we use a bucket list data structure [2] to maintain cell gains. The gain (GA) of a cell is the number of nets by which the cutsize would decrease if the cell is moved from its current partition to a destination partition. From all cells with the highest gain (GA) in the bucket lists, the cell with the highest ratio cut gain (RCG) is selected, preliminarily moved to its destination partition, and locked. The ratio cut gain is calculated by a special FPGA-specific objective function as described in Section 3.4.2. Updating the bucket lists and repeating the previous step, we obtain a sequence of cells to be moved from their source partition to a destination partition. When either all cells are locked or any further move would violate the partition size specifications, the cells of the partial sequence which achieves the minimum cutsize are actually moved to their destination partitions. This moving of cell groups is applied until no further improvement can be obtained [2]. Subsequently, of all remaining partitions, the partition with the highest cutsize is selected as the next central partition. The iterative improvement procedure terminates, when each partition has been selected as central partition exactly once. Finally, the center of gravity of all cells in each partition is assigned to the center of the partition to maintain the partitioning during the next embedding step. Figure 4 shows one step of the k-way ratio cut algorithm with the current central partition, all neighbor partitions, and the possible move directions for the 2×3 example.

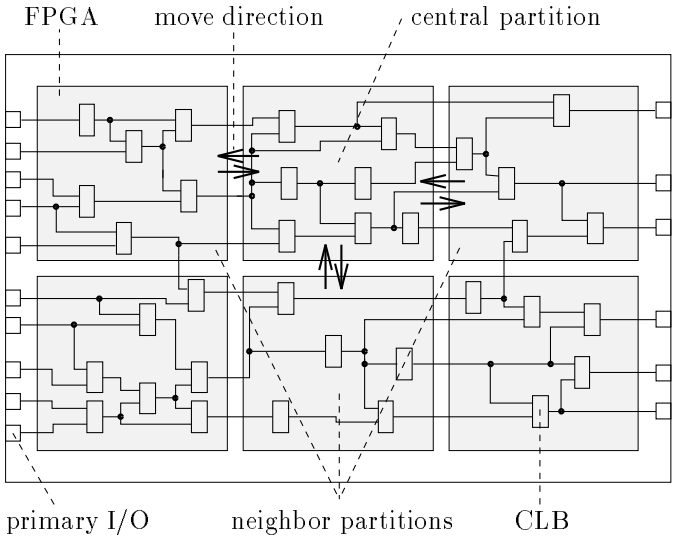

Figure 4: Overview of the k-way ratio cut

The main advantage of this approach is that not only the number of required IOBs of two partitions just obtained from one initial partitioning step can be reduced, as proposed by recursive min-cut, but also previously divided partitions can exchange cells if they are adjacent.

3.4.2 A New FPGA-Specific Objective Function

The ratio cut gain is calculated by a new FPGA-specific objective function, derived from the adapted ratio cut objective (10).

Since the total size $|V_{clb_{p_1}}| + |V_{clb_{p_2}}|$ of two adjacent partitions p_1 and p_2 may differ for each participating pair of partitions, the adapted ratio cut (ARC) will be different for a cell to be moved to partitions of different size. This effect is illustrated in Figure 5.

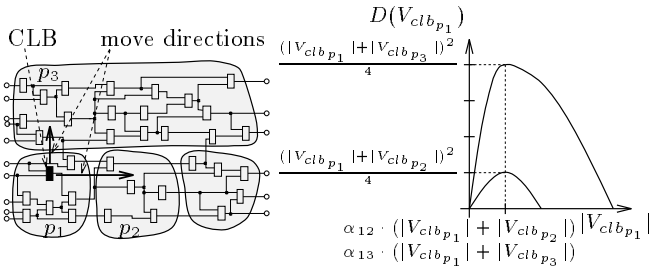

Figure 5: The effect of different partition sizes

Assume that the CLB filled black would yield exactly the same reduction of the cutsize and reach the maximum of the denominator $D(V_{clb_{p_1}})$ when either moved from partition p_1 to partition p_2 or moved to p_3. Then, the adapted ratio cut for a move of the CLB to partition p_3 will be smaller than the adapted ratio cut for the move to partition p_2, as the denominator $D(V_{clb_{p_1}})$ has different maximum values for the two move directions since the total size $|V_{clb_{p_1}}|+|V_{clb_{p_2}}|$ of the partitions p_1 and p_2 is much smaller than the total size $|V_{clb_{p_1}}|+|V_{clb_{p_3}}|$ of the partitions p_1 and p_3. To provide equal chances for all move directions, we scale the denominator with $|V_{clb_p}|^2$ and obtain the scaled ratio cut objective (SRC):

$$SRC = \frac{C_{p_1p_2}}{\frac{D(V_{clb_{p_1}})}{|V_{clb_p}|^2}} \tag{11}$$

Minimizing the cutsize does not necessarily lead to an even distribution of the IOBs. The difference between reducing the cutsize of two partitions ($C_{p_1p_2}$) and reducing the number of required IOBs is shown in Figure 6.

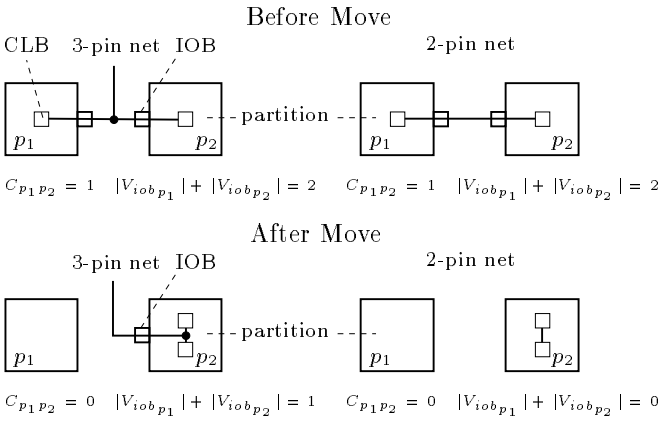

Figure 6: The difference between reducing the cutsize and the number of IOBs

In the left example, the cutsize decreases by one and the number of IOBs also decreases by one if one CLB is moved to the other partition. In contrast to that, in the right example, the cutsize also decreases by one if one of the CLBs is moved to the other partition, but the number of IOBs decreases by two. Thus, the reduction of the cutsize is the same in both cases, but the reduction of the number of required IOBs is higher in the right case. To favor a reduction of the number of IOBs, we substitute the cutsize in the numerator of equation (11) by the sum of the number of required IOBs in the source and destination partition scaled by the total number of FPGAs currently covering partition p and obtain the IOB count driven scaled ratio cut objective ($ISRC$):

$$ISRC = \frac{\frac{|V_{iob_{p_1}}|+|V_{iob_{p_2}}|}{\#FPGAs(p)}}{\frac{D(V_{clb_{p_1}})}{|V_{clb_p}|^2}} \tag{12}$$

Finally, we calculate the ratio cut gain (RCG) by subtracting the IOB count driven scaled ratio cut ($ISRC_A$) after the cell is moved from the IOB count driven scaled ratio cut ($ISRC_B$) before the cell is moved and obtain:

$$RCG = ISRC_B - ISRC_A \tag{13}$$

This objective is used by our approach to select the cell with the maximal RCG from all cells in the bucket list data structures with the highest gain (GA).

4 Experimental Results

The results of our new k-way FPGA partitioner called EBRC are compared to the results of the *k-way.x* approach of Kužnar et al. [3]. We compare to the best results obtained by this approach [3] setting the threshold replication potential T=2. This recursive min-cut approach minimizes the total dollar cost and interconnect allowing functional cell replication. Though we do not explicitly minimize the dollar cost of used devices and do not perform any cell replication techniques, we compare our approach to *k-way.x*. We compare both approaches applying the circuits of the ACM/SIGDA XILINX FPGA Benchmark Suite [15]. The characteristics of these circuits are summarized in Table 2.

In Table 2 the results of our EBRC approach are compared to the results of *k-way.x*. It shows the total dollar cost and computation time of the *k-way.x* approach as published in [3] as well as the FPGA board grid, the device distribution, the total dollar cost, and the computation time of our EBRC approach. Furthermore, it gives the relative improvement of our approach to *k-way.x* in terms of total dollar cost and computation time. All computations in [3] were executed on a SUN SparcStation 1+. Our platform was a DEC 3000 Model 600 AXP. All circuits where the results of *k-way.x* are not available are denoted with n.a.. Only the 9 benchmarks reported by Kužnar et al. [3] contribute to the shown total and average values.

We obtained feasible multiple FPGA partitionings for all 13 circuits and our approach outperforms *k-way.x* in all but four cases. The total dollar cost of the required devices for the circuits is reduced up to 20%. Although our approach does not explicitly minimize the total dollar cost of devices used and does not allow cell replication, we improved the cost by 2.3% on an average over *k-way.x*. Overall, we save \$3.72 total cost which equals a total reduction of 2.8%. Considering that the absolute lower total cost bound for this Benchmark Suite is \$113.72 [1] our method is 21.5% closer to the absolute minimum than *k-way.x*.

Since our approach assigns the partitions to FPGA devices during partitioning, we already obtain a placement of the FPGAs on the FPGA board. Thus, the task of assigning the FPGAs to positions on the FPGA board as required by previous approaches is already solved during partitioning. Moreover, the length of the inter-device connections is minimized during the embedding step. Thus, when considering the proposed FPGA placement, we expect that the signal delays of the inter-device signals will be small compared to previous approaches.

circuit	#CLBs	#PIOs	*k-way.x*		EBRC				Improvement	
			Total [$]	CPU [s]	Grid	Device-Distribution	Total [$]	CPU [s]	Total [%]	CPU [%]
c2670	150	221	n.a.	n.a.	2x2	{2,2,0,0,0}	**4.72**	**1.7**	–	–
c3540	283	72	4.56	32.0	3x1	{1,1,1,0,0}	**4.20**	**3.8**	7.9	88.1
c5315	377	301	**6.92**	102.2	5x1	{0,3,2,0,0}	7.76	**8.1**	-12.1	92.1
c6288	833	64	13.76	292.7	6x1	{0,0,6,0,0}	**11.04**	**14.6**	19.8	95.0
c7552	489	313	**7.36**	82.7	1x5	{0,2,3,0,0}	8.24	**10.1**	-12.0	87.8
s1238	158	30	n.a.	n.a.	2x1	{0,2,0,0,0}	**2.72**	**1.2**	–	–
s5378	381	86	**6.19**	142.4	2x1	{0,0,0,2,0}	6.30	**4.7**	-1.8	96.7
s9234	454	43	7.98	201.2	3x1	{0,1,1,1,0}	**6.35**	**16.9**	20.4	91.6
s13207	915	154	**18.12**	364.7	5x2	{0,5,3,1,1}	20.30	**172.6**	-12.0	52.7
s15850	842	102	14.97	402.3	2x3	{0,2,1,3,0}	**14.01**	**104.5**	6.4	74.0
s35932	2 153	357	n.a.	n.a.	7x4	{5,10,10,3,0}	**46.45**	**2 001.0**	–	–
s38417	2 221	136	n.a.	n.a.	2x10	{2,3,14,1,0}	**34.99**	**1 382.8**	–	–
s38584	2 901	292	51.19	3 832.0	6x4	{1,7,9,7,0}	**49.13**	**2 243.2**	4.0	41.5
$\sum$/avg			131.05	5 452.2			127.33	2 578.5	2.3	79.9

Table 2: Comparison of partitioning results of *k-way.x* and EBRC

5 Conclusions

We developed an efficient k-way partitioning method for multiple type FPGA partitioning. Our research leads to the following conclusions:

- The combination of the analytical embedding technique with the iterative improvement procedure is a promising approach for k-way FPGA partitioning.
- The k-way ratio cut method with our new FPGA-specific objective function allowing each partition to exchange CLBs with all adjacent partitions yields impressive partitioning quality.
- Partitioning results for a large set of benchmark circuits demonstrate that our approach outperforms previous methods.

Acknowledgment

The authors would like to thank K. Antreich and F. Johannes for continuous support as well as A. Schoene, M. Hermann, and C. Legl for many fruitful discussions. This work was supported in part by the German National Science Foundation (DFG) under Grant An 125/11-1.

References

[1] R. Kuznar, F. Brglez, and K. Kozminski, "Cost Minimization of Partitions into Multiple Devices," *Proceedings 30th ACM/IEEE Design Automation Conference*, pp. 315–320, 1993.

[2] C. M. Fiduccia and R. M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," *Proceedings 19th ACM/IEEE Design Automation Conference*, pp. 175–181, 1982.

[3] R. Kuznar, F. Brglez, and B. Zajc, "Multi-way Netlist Partitioning into Heterogeneous FPGAs and Minimization of Total Device Cost and Interconnect," *Proceedings 31th ACM/IEEE Design Automation Conference*, pp. 238–243, 1994.

[4] N.-S. Woo and J. Kim, "An Efficient Method of Partitioning Circuits for Multiple-FPGA Implementation," *Proceedings 30th ACM/IEEE Design Automation Conference*, pp. 202–207, 1993.

[5] N.-C. Chou, L.-T. Liu, C.-K. Cheng, W.-J. Dai, and R. Lindelof, "Circuit Partitioning for Huge Logic Emulation Systems," *Proceedings 31th ACM/IEEE Design Automation Conference*, pp. 244–249, 1994.

[6] D. J.-H. Huang and A. B. Kahng, "Multi-Way System Partitioning into a Single Type or Multiple Types of FPGAs," *Proceedings 1995 ACM Third International Symposium on Field-Programmable Gate Arrays*, 1995.

[7] P. K. Chan, D. F. Schlag, and J. Y. Zien, "Spectral K-Way Ratio Cut Partitioning and Clustering," *Proceedings 30th ACM/IEEE Design Automation Conference*, pp. 749–754, 1993.

[8] C. J. Alpert and A. B. Kahng, "Multi-Way Partitioning Via Spacefilling Curves and Dynamic Programming," *Proceedings 31th ACM/IEEE Design Automation Conference*, pp. 652–657, 1994.

[9] G. Sigl, K. Doll, and F. M. Johannes, "Analytical Placement: A Linear or a Quadratic Objective Function?," *Proceedings 28th ACM/IEEE Design Automation Conference*, pp. 427–432, 17.-21. jun 1991.

[10] L. Hagen and A. B. Kahng, "Improving the Quadratic Objective Function in Module Placement," *Fifth Annual IEEE International ASIC Conference and Exhibit*, pp. 42–45, 1992.

[11] B. M. Riess, K. Doll, and F. M. Johannes, "Partitioning Very Large Circuits Using Analytical Placement Techniques," *Proceedings 31st ACM/IEEE Design Automation Conference*, pp. 646–651, 1994.

[12] L. Hagen and A. B. Kahng, "Fast Spectral Methods for Ratio Cut Partitioning and Clustering," *Proceedings IEEE/ACM International Conference on Computer-Aided Design*, pp. 10–13, 1991.

[13] Y.-C. Wei and C.-K. Cheng, "Ratio Cut Partitioning for Hierarchical Designs," *IEEE Transactions on Computer-Aided Design*, vol. 10, pp. 911–921, July 1991.

[14] P. R. Suaris and G. Kedem, "An Algorithm for Quadrisection and Its Application to Standard Cell Placement," *IEEE Transactions On Circuits And Systems*, vol. 35, pp. 294–303, Mar. 1988.

[15] F. Brglez, D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *Proceedings IEEE International Synposium on Circuits and Systems ISCAS 1989*, vol. 3, no. 3, pp. 1929–1934, 1989.