Physical Information Driven Packing Method in FPGA[*]

Wentao Sui, Sheqin Dong, Jinian Bian，Xianlong Hong

Department of Computer Science and Technology, Tsinghua University, Beijing, China
suitwt05@mails.tsinghua.edu.cn, { dongsq, bianjn, hongxl }@mail.tsinghua.edu.cn

Abstract

Packing-integrating basic logic units into higher level logic unit-is an important step in cluster-based hierarchical FPGA placement. The physical information of logic block acquiring before packing has a real influence over both wirelength-driven and timing-driven packing algorithms. A new packing method with consideration with "physical" information is presented. To do this, we can not only pack basic logic units into higher level logic unit efficiently but also decrease the connection resource used by latter routing physical design stage. Experimental result show that this method performs well in island structure type FPGA placement.

Keywords: FPGA, Packing, Physical information

1. Introduction

The typical island FPGA architecture contains a two-level physical hierarchy: Basic Logic Element (BLE) and Configurable Logic Block (CLB) [1]. As described in the Fig. 1.

BLE is composed of two main elements: K-input Look-Up-Table (LUT) and DFF, the output of K-input LUT can be registered or transmitted directly to the BLE output. N BLEs form a higher level logic CLB. Each of the CLB inputs (I) can drive [1]all the BLEs, and each BLE drives an output. The output of a BLE can interiorly connects with other BLEs inputs in the same CLB. The interconnection delay between BLEs within the same CLB is usually much smaller than the delay between BLEs in different CLBs. Here K, N, and I are parameters described by the architecture file.

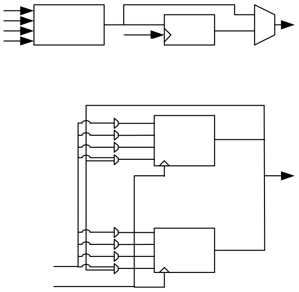

Fig. 1: FPGA two-level physical structure.

In the typical FPGA design flow, as showed in figure 2 [1], a circuit is first synthesized and mapped into a netlist of LUTs and Flip-Flops (FFs). Then it goes through the following three steps: packing, placement and routing. The packing step first integrates a LUT and a FF into BLE if and only if the output of the LUT is only directly connected with the FF. Otherwise, LUTs and FFs are regarded as BLEs respectively. Then the packing process arranges BLEs into CLBs according to the timing and connectivity of the mapped netlist. Since the local interconnect in CLBs can be make faster than the general-purpose interconnect between CLBs, this step can improve the final circuit speed. The placement step places the

[*]This work was supported by National Natural Science Foundation of China under grant NSFC-90607001

packed netlist onto the array of the FPGA on-chip CLBs and the routing routs all the wires in the netlist with the available routing resources on the device.

For the routing resource on the FPGA is fixed, the quality of packing plays an important role to the result of placement and routing. An FPGA in which every CLB contains several LUTs will need fewer placement and routing resource to implement a circuit than an FPGA in which each CLB with only one BLE. There are no accurate means available to estimate the distance between BLEs before placement, routability problem is difficult to deal with at packing stage. In this paper, we present a new method to consider routability at the packing stage. We apply our performance-driven method for packing, and finally place and route the circuit using VPR [9] a very famous FPGA design tool.

This paper is organized in several sections. Previous work on packing method will be present in section 2. Section 3 presents the details of our packing framework. Numerical results and conclusion are presented in section 4.

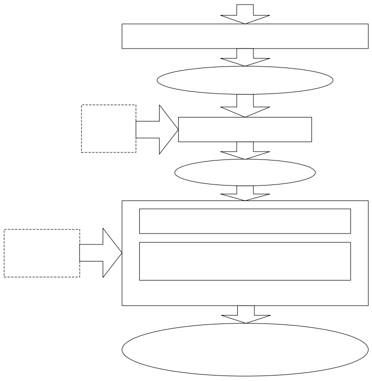

Fig. 2: FPGA design flow

2. Previous Work

Researchers have tried to capture interconnect requirements for routability of the circuit using a variety of metrics. VPack [3] focused mainly on the area-delay trade-off getting from the size and structure of the CLB. The main idea was to use a packing cost function to pack a technology-mapped circuit into CLBs with a given size and input/output pin constraints. The objective of VPack was to minimize the number of CLBs and the number of inputs per CLB. While in T-VPack [3], the author combined static timing information with VPack, which based on the idea of packing blocks on timing-critical paths to exploit faster circuit speed. The CLBs generated by T-VPack used an average of 12% fewer tracks than the CLBs generated by VPack for the same array size. RPack[4] presented a routability-driven packing algorithm. It showed that different inter-BLEs connection order could give different routability contribution, then prioritized these order into an improved packing cost function, and achieved fewer routing tracks than VPack but comparable to those generated by T-VPack.

TLC [5], MLC [6] are two packing algorithm focus on the timing parameter. They attempted to duplicate logics on the critical path during packing to obtain a set of CLBs with optimal depth. While effectively at reducing critical path delay, the process of logic duplication could be hard to control, leading to large increase in area.

3. Packing Method

Our packing method has two stages: physical information acquiring and performance-driven packing.

The top-down partition process can be viewed as a sequence of passes where each pass examines all bins and divides some of them into smaller bins. Min-cut partition has been employed successfully in ASIC placement. In these approaches [7], the relative positions of blocks can be obtained from fast initial placement. In

Synthesised
circuit

order to get the physical information before placement, we use hMetis [10] to recursively bi-partition the mapped netlist. Once the number of nodes in a partition is less than a predetermined number or the depth of the partitioning tree has exceeded a threshold, the partition stops. Then a deep-first and breadth-first search are performed respectively, after these two traversal, every partition gets two coordinate values which named x and y. Each BLE in the same partition has the same x and y values. The distance between two BLEs is defined as:

$$Dist_{ij} = | x_i - x_j | + | y_i - y_j | \qquad (1)$$

In T-VPack, if the CLB being packed is not full and there is no BLE with connection to it, in order to increase the utilization of the CLB, a BLE with more pins used will be selected into the current CLB without distance consideration which can bring downside influence on the timing performance of the circuit. This will be discussed below.

To improve the circuit performance, we use the same static timing analysis method like [1] to compute the criticality of each connection between BLEs and approximate the delay between CLBs as a constant value. The criticality of a connection i to be:

$$ConnectionCriticality(i) = 1 - \frac{slack(i)}{MaxSlack} \quad (2)$$

MaxSlack is the largest slack among all connections in the circuit. The BLE attached to the connection with the highest ConnectionCriticality was endowed with the highest criticality and will be selected as the seed. We sort the BLEs according static criticality, and select a seed top-down each time, Once a seed is chosen, the attraction function used to determine the next BLE, *B*, to be added to the current CLB, *C*, is:

$$Attraction(B) = \lambda.Criticality(B) + (1-\lambda)\frac{| Nets(B) \bigcap Nets(C) |}{MaxNets} \qquad (3)$$

There may be several BLEs have the same criticality value, the criticality of a BLE is slightly adjusted by several tie-breakers:

$$Criticality(B) = BaseBLECrit(B) + \varepsilon.TotalPathsAffected(B) + \varepsilon^2.D_{source}(B) \qquad (4)$$

We search the CLB not full in turn, if there are some blocks with connection to it, we packing the blocks into the current CLB according to (1). Otherwise, we select a block with the smallest cost function value:

$$Cost(Q) = \alpha.Criticality(b_j) + (1-\alpha)\frac{Dist(b_{ci}, b_j)}{MaxDis\tan ce} \qquad (5)$$

The Criticality(b_j) is the criticality of BLE(b_j) outside the CLB being packed, Dist(b_{ci},b_j) is the distance between BLE(b_{ci}) in the CLB and BLE(b_j) outside the CLB. The MaxDistance is the longest distance among all BLEs in the circuit. For computation convenience, we can set the value to the long dimension of the FPGA chip

The parameter α is used to control the two variables. If α equals to 1, the packing method is similar to T-VPack. We can adjust this value dynamically to get better result. The distance is computed using the physical information getting from early partition process.

We use the same hill-climbing operation like [2], which promotes the utilization of CLBs too. The key idea of hill-climbing is that adding a BLE to a CLB in which all of its inputs are already present, and in which the output of the BLE is used by some other BLE already in the

CLB decreases the number of distinct inputs to the CLB by one.

4. Experiment result and conclusion

In this section, we present the experiment results of our packing methodology and compare these with the results produced by the popular FPGA packing tool T-VPack. Table 1 shows the characteristics of nine benchmark circuits. The column ExtNets gives the average net number outside CLBs and WL is the wirelength of the circuit. The third column shows the criticality of the critical path. Numerical results show that with the size of the circuit increasing our method yielded a reduction of 8.58% in wire length compare to T-Vpack. At the same time, our algorithm decreases the average CLB external net number.

We got the physical information using a simple method and explored the use of physical information during packing. The experiment results show the effectiveness of the method.

5. References

[1] G. Chen and J. Cong, “Simultaneous timing driven clustering and placement for FPGAs,” *In Proc. FPL*, pp. 158-167, 2004.

[2] Vaughn Betz, Jonathan Rose, Alexander Marquardt, “Architecture and CAD for deep-submicron FPGA”, *KLUWER academic publishers.*

[3] A. Marquardt, V. Betz and J. Rose, “Using Cluster-Based Logic Blocks and Timing-Driven Packing to Improve FPGA Speed and Density”, *ACM/SIGDA International Symposium on FPGAs*, pp. 37-46, 1999.

[4] E. Bozorgadeh, S. Ogrenci-Memik, M. Sarrafzadeh, “RPack: Routability-Driven packing for cluster-based FPGAs,” *In Proc. ASPDAC*, pp. 629-634, 2001.

[5] J. Cong and M. Romesis, “Performance-driven multi-level clustering with application to hierarchical FPGA mapping”, *in Proc. DAC*, pp. 389-394, 2001.

[6] C. Sze, T.-C. Wang, L.-C. Wang, “Multilevel circuit clustering for delay minimization”, *in IEEE Trans. CAD,* pp. 1073-1085, 2004.

[7] K. Vorwerk and A. Kennings, “An improved multi-level framework for force-directed placement”, *In Proc. DATE,* pp. 902-907, 2005.

[8] http://www.cad.polito.it/tools/itc99.html

[9] V. Betz, and J. Rose, “VPR: A New packing, Placement and Routing Tool for FPGA Research,” *in 7th Intel. Workshop on Field-Programmable Logic and Applications*, pp. 213-222, 1997.

[10] http://www-users.cs.umn.edu/karypis/metis/hmetis/.

Table. 1: Packing results

	ours			T-VPack			Improvement Of ExtNet(%)	Decrease of WL(%)
benchmark	**ExtNets**	**WL**	**Crit**	**ExtNets**	**WL**	**Crit**		
Net1	1.191441	120	1.67344	1.953488	121	1.66295	2	0.8
Net2	1.721078	485	2.78346	1.789063	491	2.90638	3.8	1.24
Net3	1.893065	117	1.49665	1.933673	121	1.66295	2.1	3.4
Net4	3.049550	1313	4.097604	3.139640	1362	4.45392	2.8	3.7
Net5	1.266516	671	4.373158	1.890323	691	4.456838	3.3	2.98
Net6	1.886421	529	3.320465	1.965035	550	3.43378	4	3.97
Net7	1.834701	791	3.149447	1.93333	826	3.29785	5.1	4.42
Net8	1.800378	126455	18.0239	1.965478	137302	19.339	8.4	8.58
Net9	2.894435	209	1.999208	2.956522	213	2.0106	2.1	1.91
average							3.73	3.44