Combined Channel Segmentation and Buffer Insertion for Routability and Performance Improvement of Field Programmable Analog Arrays

Hu Huang, Joseph B. Bernstein, Martin Peckerar and Ji Luo
Dept.of ECE, University of Maryland, College Park, MD 20742
huanghu, joey, peckerar, jluo @eng.umd.edu

Abstract

In this paper, we propose a combined channel segmentation and buffer insertion approach, which minimizes the number of buffers inserted while satisfying the delay constraints for routing channels of field-programmable analog arrays. A segmented routing algorithm based on minimum-cost-bipartite-matching is improved with demand awareness and used to evaluate the various routing channels generated. Experiments show that, compared to a sequential segmenting-then-buffering design, our approach can significantly reduce the total number of buffers required, while achieving improved routability and minimum average interconnect delay. It is also shown that by increasing the number of long segment appropriately, the algorithm can dramatically improve the routability with a moderate increase on the number of buffers.

I. Introduction

Field programmable analog arrays (FPAAs) are composed of configurable analog blocks (CABs) and programmable interconnects that can be configured to implement analog circuits. These arrays have wide potential applications, both in academia and in industry. As modern IC processes scale to smaller size and millions of transistors are available on a single chip, a problem has emerged as how to use those resources effectively and realize their full potential. Interconnects and programmable switches will inevitably introduce parasitics. For large scale FPAAs, those parasitics will ultimately be the limiting factor of the system performance, other than the circuits. In addition, since interconnects don't scale as well as transistors, more area of an FPAA will be devoted to routing, which already occupied the largest portion of most commercial programmable devices. Therefore, routing architecture has becoming an essential part in FPAA design. It would be impossible for such systems to realize the full potential if the routing delays and resource utilization were not handled well [1].

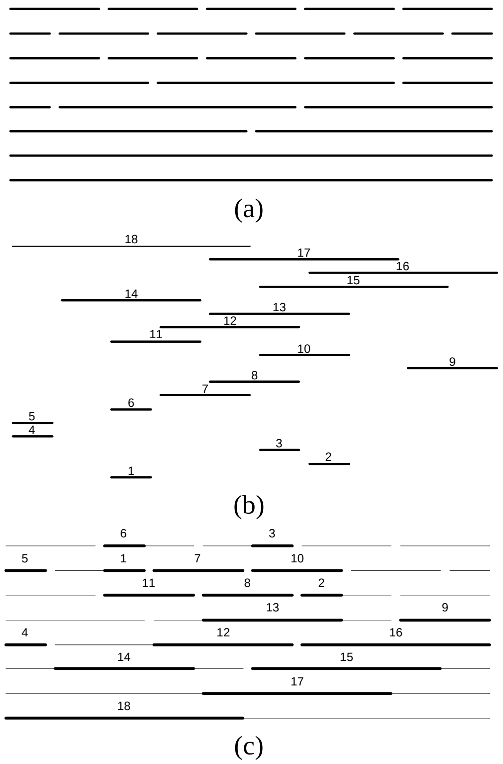


Figure 1. (a) A segmented channel with eight tracks (b) eighteen nets to be assigned (c) minimum-cost routing results

One important feature of a routing architecture is its channel segmentation scheme, which defines the lengths and locations of various routing wire segments. Intuitively, there should be a strong correlation between the routing segmentation and the real net distribution. Routing tracks composed of long segments usually have better performance, but they also result in lower routability and higher wire wastage. On the other hand, tracks composed of short segments provide more flexibility and reduce the waste of wire, but performance is sacrificed [2]. Similarly, the locations of segments with respect to a net span are also very important in determining whether the net can be routed optimally. It is therefore the main object in constructing a routing architecture to match the channel segmentation scheme to the actual net distribution as closely as possible by choosing segments of appropriate lengths and positions. It has been clearly demonstrated that a well-segmented channel can greatly help the router to achieve effectively high routability and resource utilization [3-5]. An example of channel segmentation and matching-based routing is shown in Fig. 1.

A good routing architecture should also minimize the performance degradation caused by interconnect parasitics, on which existing channel segmentation algorithms have paid little consideration. This is allowable for small scale FPAAs, however, as the scale of the FPAA grows, the interconnect delay becomes so significant that it must be taken into account at the earliest stage. There are usually three techniques to reduce the delay of an existing topology: transistor sizing, wire sizing and buffer insertion, which have been studied extensively for free channel routings. The optimum transistor sizing, metal width and metal spacing for programmable interconnect have been studied in [6]. In this paper we focus on buffer insertion, which can either directly reduce the RC delay of a long wire or reduce the net delay by decoupling a large load off the critical path. Theoretical results have been derived and algorithm proposed for computing the optimum buffer insertion for fixed net trees [7]. However, routing wires are pre-fabricated in the FPAAs, approximate buffer insertions are made while the net tree topology is still unknown. This makes it almost impossible to find an overall optimum buffer planning scheme for all scenarios. In this study, we assume there is a delay constraint for each net, and insert the minimum number of buffers based on this constraint. The lengths and staggering of the segments are varied to find the optimum segmentation scheme requiring the fewest buffers while satisfying the delay constraints and routability requirement.

The rest of this paper is organized as follows. Section II overviews the existing channel segmentation algorithms and defines our problem. Section III presents the combined channel segmentation and buffer insertion approach, while the routing algorithm is described in Section IV. Experiment results are presented in Section V and conclusions are given in Section VI.

II. Prior work and our problem

A number of channel segmentation designs have been examined in the literature. El Gamal et al. showed that a segmented routing channel could achieve comparable routability to a freely customized routing channel [3]. Zhu and Wong presented an algorithm for the channel segmentation design problem based also on a stochastic analysis [4]. Pedram et al. presented an analytical model for the design and analysis of effective segmented channel architectures [5]. Recently, Jai-Ming Ling et al. presented a unified segmentation and routing design for array-based FPAAs [8].

In this paper, we deal with row-based architectures only. The conclusions can be extended to array-based FPAAs as described in [9]. The following notations are used in defining the problem:

L: Length of a channel
T: Total number of tracks in the channel
M: Maximum number of segments for routing a net
h(*x*,*l*): Probability of a net with length *l* originating at *x*
D_c: Delay constraint of a single net

Our design problem is formulated as follows: *Given L, T, h(x,l) and* D_c*, design a channel segmentation and buffer insertion scheme that maximize success rate for M-segment routing, while minimizing number of buffers required to meet the interconnect delay constraint.*

III. Combined channel segmentation and buffer insertion

In this Section, we will first describe the optimum buffer insertion algorithm for a given net length, then introduce our algorithm of segment length selection and track assignment.

3.1 Buffer insertion

Assuming a uniform-sized buffer with input capacitance C_b, output resistance R_b and intrinsic delay T_b, the optimum Elmore delay that can be achieved by inserting *k* bufer for a given wire of fixed length *l* has been derived in [7] as

$$D_k = \frac{Rl(kC_b + C_{si}) + Cl(kR_b + R_{so}) + (kC_b + C_{si})(kR_b + R_{so})}{k+1} + kT_b + \frac{RCl^2 - \frac{kR(C_b - C_{si})^2}{C} - \frac{kC(R_b - R_{so})^2}{R}}{2(k+1)} \qquad (1)$$

where *R* and *C* is the unit resistance and capacitance for the wire, R_{so} is the source resistance, C_{si} is the sink capacitance, and the optimum number of buffers for the wire is found to be

$$k_{opt} = \left\lfloor -\frac{1}{2} + \frac{1}{2}\sqrt{1 + \frac{4U}{V}} \right\rfloor \qquad (2)$$

where

$$U = \frac{\left[RCl + R(C_{si} - C_b) + C(R_{so} - R_b)\right]^2}{2RC} \qquad (3)$$
$$V = (K_b + R_b C_b)$$

In a performance-constraint routing, it is usually preferable to have the interconnect delay constrained by D_c. To find the minimum number of buffers, *k*(*l*), to be inserted satisfying that constraint, from (5) we note that

$$D_{k-1} - D_k = \frac{U}{k} - \frac{U}{k+1} - V \quad (4)$$

Assuming $D_c > D_{k_{opt}}$, we have

$$\begin{aligned} & D_c - D_{k_{opt}} \geq D_{k(l)} - D_{k_{opt}} \\ & = D_{k(l)} - D_{k(l)+1} + D_{k(l)+1} - D_{k(l)+2} + \cdots + D_{k_{opt}-1} - D_{k_{opt}} \\ & = \frac{U}{k(l)+1} - \frac{U}{k_{opt}+1} - V\left[k_{opt} - k(l)\right] \end{aligned} \quad (5)$$

Solving for $k(l)$ yields

$$k(l) = \left\lfloor \frac{Q}{2} - \frac{1}{2}\sqrt{Q^2 - \frac{4U}{V}} \right\rfloor \quad (6)$$

where $Q = \frac{U}{V(k_{opt}+1)} + (k_{opt}+1) + \frac{D_c - D_{k_{opt}}}{V}$, U and V is given by (3). If $D_c \leq D_{k_{opt}}$, we set $k(l) = k_{opt}$.

3.2 Combined segment length selection

Like most of previous work, we adopt the staggered, non-uniform segmentation model: A channel is partitioned into several regions. The tracks in the rth region are divided into segments of length Λ_r, also called type r segments designated to route nets whose lengths fall in the range $(M\Lambda_{r-1}, M\Lambda_r]$ (assuming $\Lambda_{r-1}<\Lambda_r$). The segments are arranged in a staggered fashion to allow the maximum flexibility of routing nets starting at different locations. Many important details of the model are still left open, such as how to choose the segment lengths and number of tracks in each region. These details can greatly affect the routability and buffer planning of the resulting channel.

In a design without considering buffer insertion [4], the Λ_r's for an arbitrary net length distribution $f(l)=\Sigma_x h(x,l)$ are determined as follows. We set $\Lambda_J = L$, and choose Λ_r, $r = J$-1, J-2, … as the largest value that satisfies

$$\sum_{l=M\Lambda_r+1}^{M\Lambda_{r+1}} f(l)\cdot l \Bigg/ \sum_{l=M\Lambda_r+1}^{M\Lambda_{r+1}} f(l) \geq \frac{M\Lambda_{r+1}}{\xi} \quad (7)$$

The parameter ξ >1 should be carefully chosen to achieve the best results.

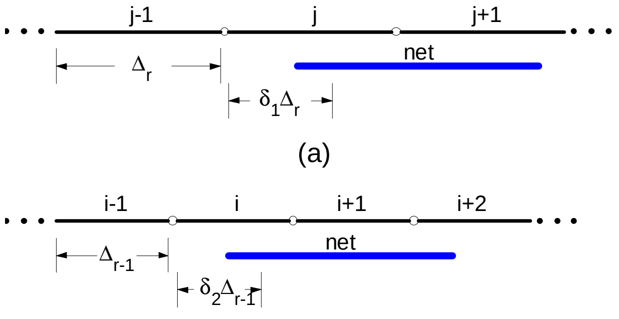
For a combined buffer insertion and channel segmentation, to minimize the number of buffers to be inserted, as well as the number of types of segments, Λ_r is chosen as following

$$\Lambda_r = \begin{cases} \Lambda_{rm} & if \quad k(\Lambda_{rm}) = k(\Lambda_{r+1}) \\ MAX\{\Lambda, \quad k(\Lambda) = k(\Lambda_{rm})\} & else \end{cases} \quad (11)$$

where Λ_{rm} is the value got from (7). The working principle here is to reduce the designated net length range to type r+1 segments, which requires more buffers than type r segments, by up-shifting Λ_r to the longest segments that requires the same number of buffers as Λ_{rm}.

3.3 Track assignment

The number of tracks in each region should be proportional to the expected usage of that type of segments. Since the segments in one track are actually placed one by one, their originations (left ends) can only appear at multiples of the segment length. Those nets originate from other points (called off-grid nets) may fail to be routed by any track in their designated region, and require an extra track in regions containing longer segments. To calculate the expected usage of tracks in region r, we consider two cases similar to those described in [5] and introduce the staggering factors δ_1 and δ_2 to describe how much the off-grid situations are taken into consideration.



(a)

(b)

Figure 2. Illustration of the staggering factors.

The first case is those nets have length in the range $(M\Lambda_{r-1}, M\Lambda_r]$ and can be routed using tracks in region r. For a net with origination x in the range of $[j\Lambda_r, (j+\delta_1)\Lambda_r]$ (0≤ δ_1≤1), its length should be no more than $(j+M)\,\Lambda_r - x$, as illustrated in Fig. 2 (a) (M = 2). The expect number of tracks in region r for such nets is given by

$$n_j^r = \sum_{x=j\Lambda_r}^{(j+\delta_1)\Lambda_r} \sum_{l=M\Lambda_{r-1}+1}^{(j+M)\Lambda_r - x} h(x,l) \quad (1)$$

The second case is those nets have length in the range $(M\Lambda_{r-2}, M\Lambda_{r-1}]$, but cannot be routed using tracks in region r-1. For a net with origination x in the range of $[i\Lambda_{r-1}, (i+\delta_2)\Lambda_{r-1}]$ (0≤ δ_2≤1), its length should be more than $(i+1)\,M\Lambda_{r-1} - x$, as illustrated in Fig. 2(b) (M = 2). The expect number of tracks in region r for such nets is given by

$$m_i^r = \sum_{x=i\Lambda_{r-1}}^{(i+\delta_2)\Lambda_{r-1}} \sum_{l=Max\{M\Lambda_{r-2},(i+M)\Lambda_{r-1}-x\}+1}^{M\Lambda_{r-1}} h(x,l) \quad (2)$$

The total expected number of tracks for type r region is then given by the sum of the maximum expected usage of both cases

$$p^r = MAX_{j=0}^{L/\Lambda_r - 1} n_j^r + MAX_{i=0}^{L/\Lambda_{r-1}-1} m_i^r \quad (3)$$

Note that when δ_1=δ_2=0, $n_j^r = \sum_{l=M\Lambda_{r-1}+1}^{M\Lambda_r} h(j\Lambda_r, l)$ and $m_i^r = 0$, so p^r is simply the probability of a net length falling in the range ($M\Lambda_{r-1}$, MΛ_r].

Once p^r has been calculated for all r, the number of track allocated to region r is allocated proportionally to p^r. if there exist more than one track in a region, the tracks are displaced with an offset evenly chosen in [0, Λ_r).

IV. Routing Algorithm

We adopt the matching-based, timing-driven routing algorithm in [8]. Unlike other algorithms that route net by net, this algorithm routes a maximum clique, defined as the maximum set of nets overlapping each other, utilizing segments more effectively.

Since each routed net becoming a obstacle for subsequently nets, the routing feasibility of subsequent nets can be increased significantly by avoiding the resources that are potentially needed by other nets. We use the concept of *resource demand* to help the router to be aware of the needs of future nets. The *demand* on a segment is the number of nets that subscribe to it. By incorporating the demand for the segments into the cost function, the router automatically chooses the segments with fewer potential subscribers. This increases the chance of routing future nets successfully. The cost of allocating net n to track t is defined as

$$C(n,t) = \frac{\sum_{g=1}^{m} len(s_g)}{len(n)} \left(\sum_{g=1}^{m} Delay(s_g) \right)^{\alpha} \left(\sum_{g=1}^{m} Demand(s_g) \right)^{\beta} \quad (11)$$

where m is the number of segments in track t that are overlapped by net n (for M-segment routing , $m \le M$). $len(n)$ and $len(s)$ are the respective lengths of net n and the segment s. $Delay(s)$ is the Elmore delay of segment s. $Demand(s)$ is the demand of segment s, and a, β are weighting factors. The object is to minimize the total allocation costs, which can be solved in polynomial time by a weighted bipartite matching algorithm.

The initial segment demands are computed by routing each clique independently as if all routing resources are available. Once a net is routed, the demand for every segment used by the net is increased by one. At first β is chosen to be small, so the router has more freedom to choose the best matching segments. While the routing proceeds, the resource demand is updated as follows: If the net avoided a segment that it initially subscribed to, the demand for that segment is decreased by one, or, if the net used a resource initially not subscribed to, the demand for that segment is increased by one. If the circuit routing failed, then the routing demand is restored to its initial value and retried with tighter feasibility constraints by increasing β, and hence redefining the costs of all routing resources. This gradually forces the router to avoid over-subscribed resources.

V. Experimental Results

To evaluate the robustness of the proposed channel segmentation and buffer insertion algorithm over different net distributions, we designed channels for six different net length distributions based on geometric, normal and Poisson distributions, as listed in Table 1. It is assumed that the net left-end points follow a uniform distribution, which is very close to reality as confirmed by empirical studies [5]. We compute the rate of successful routing completion and the average delay for randomly generated routing instances according to those distributions.

Table 1. Net distributions used in the experiments.

	Ge1	Ge2	No1	No2	Po1	Po2
$f(l)$	0.6^l	$0.6^{l/4}$	$e^{-l^2/20}$	$e^{-l^2/120}$	$2^l/l!$	$6^l/l!$

We set the channel length L=20, total number of tracks T=20. The parameters for buffer insertion are chosen from the 0.18μm technology in NTRS'97 roadmap [10]: the unit wire resistance R=0.075$\Omega/\mu m$ and the unit wire capacitance C=0.118$fF/\mu m$. The buffer output resistance R_b =180Ω, the buffer input capacitance C_b =23.4fF, the intrinsic buffer delay T_b=36.4ps. The source and sink of a wire are also assumed to be a buffer. The unit length of a logic block is assumed to be 100μm.

First we investigated the effects of staggering factors δ_1, δ_2 on routability. We chose three different value of δ_1 (0, 0.4 and 1), and let δ_2 vary from 0 to 1. For each value of δ_1, δ_2, a segmented channel is constructed using the algorithm described in Section III. The delay constraints are set as the optimum delay. Five hundred routing instances were generated randomly for each net distribution. These were routed in the channels using the algorithm described in Section IV.

The 1-segment routing success rates for instances with distribution Ge1 are shown in Fig. 3. It is seen that the factor δ_1 doesn't have much effect on the routing results, causing a variation on the success rate of less than 11%. However, the choice on δ_2 does make a huge difference. When δ_2 is small, the success rate increases rapidly with δ_2 till it reaches a value around 0.5. After that, the success rate becomes rather flat and even decreases for larger δ_2. It can be explained that when δ_2 increases, more tracks are allocated to longer segments, which are more useful than short segments for 1-segment routing. However, as more tracks are assigned to long segments, the total number of segments decreases, which cancels the benefits brought by longer segments and finally makes the success rate drop.

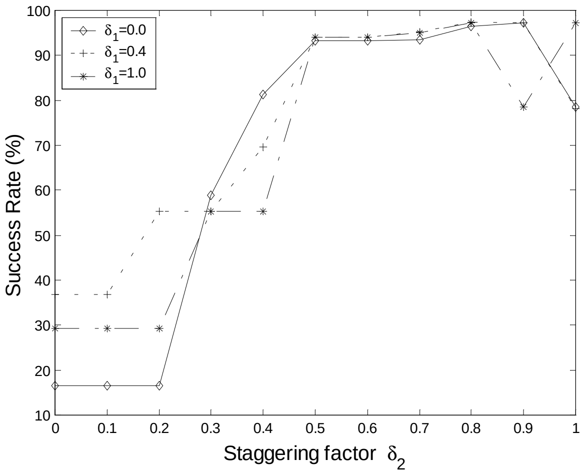
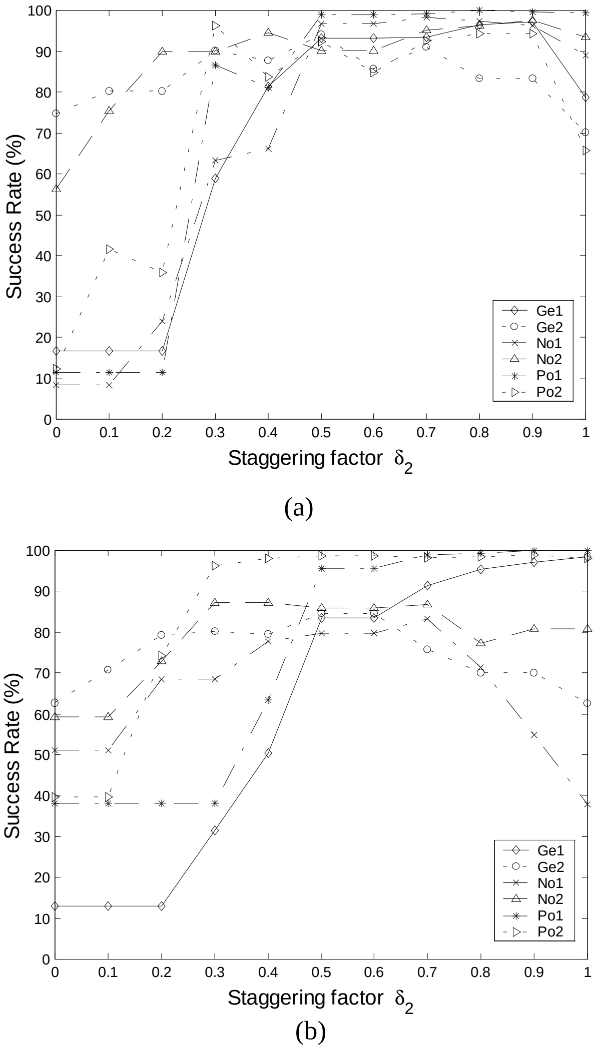


Figure 3. The effects of δ_1, δ_2 on 1-segment routability for net distribution Ge1.



(a)

(b)

Figure 4. The effects of δ_2 on routability for (a) 1-segment (b) 2-segment routing.

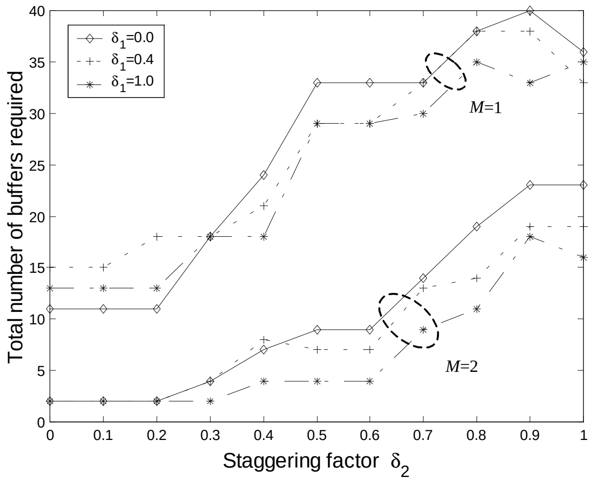


Figure 5. The effects of δ_1, δ_2 on buffer insertion

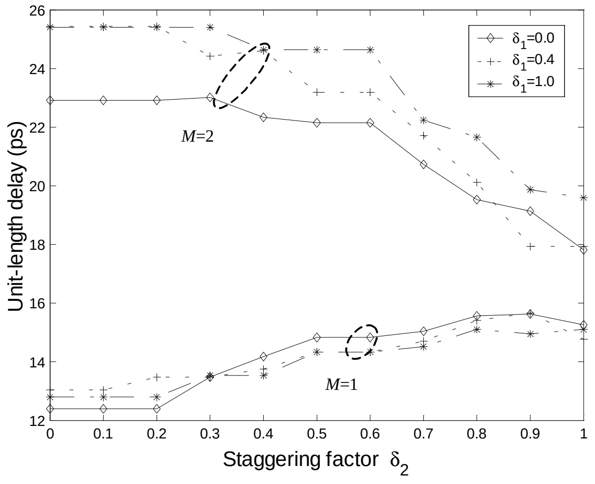


Figure 6. The effects of δ_1, δ_2 on interconnect delay.

Experiments on other five net distributions revealed similar results. The 1 and 2-segment routing success rates for all six distributions are shown in Fig. 4 (a) and (b), respectively. Although δ_2 exhibit different significance on the channel routability for various length distributions, the highest success rates are reached unanimously when δ_2 is around 0.5, indicating it an optimum value rather independent on the net length distribution.

The total numbers of buffer inserted for 1 and 2-segmentation are shown in Fig. 5. It's seen that larger δ_2 generally results in more buffers due to longer segment. Also, 2-segmention channels require much less buffers than those with 1-segmentation. Fig. 6 shows the unit-length delays, which are the average net delay per one logic block length. It's seen that for 1-segmentation the unit-length delay increases with δ_2, since longer segments usually results in larger delay. On the contrary, the unit-length delay actually decreases with δ_2 for 2-segmentation designs, because longer segments reduce the usage of connection switches, which contribute a large portion to the overall interconnect delays.

Obviously there exists an optimum value of δ_2 that gives the optimum result in terms of routability, speed and area costs. We use the metric $A = S^{\gamma} K^{-\nu} D^{-\sigma}$ to evaluate the quality of different channel segmentation and buffer insertion schemes, where S is the success rate, K is the number of buffers required and D is the unit-length delay. The parameters γ, ν and σ are used to trade off between routability and area/speed costs. For γ=3, ν=1 and σ=3, the experimental results for all six distributions are given in Tab. 2. Over the simple case of δ_1=δ_2=0, the optimized channel segmentations have an average improvement of 66.5% (46.4%) on the routing success rate, at the cost of an increase of 16.7 (12.7) on the number of buffers and 1.8*ps* increase (4.1*ps* decrease) on the unit-length delay for 1-segmentation (2-segmentation) design.

Tab. 3 shows the comparison results of the combined approach to a sequential, i.e., buffer insertion after channel segmentation approach. It seen that our combined approach can reduce the number of buffers by an average 13.7% (27.2%) and achieve a 1.5% (2.4%) increase on the routing success rate, with only 0.4% (3.0%) increase on the unit-length delay for 1-segmentation (2-segmentation) design.

Table 2. Experimental results for all six net distributions

		Optimum			δ_1=δ_2=0		
		S	*K*	*D* (*ps*)	*S*	*K*	*D* (*ps*)
M=1	Ge1	86.4%	30	14.51	6.2%	11	12.41
	Ge2	74.0%	69	12.91	39.8%	58	11.69
	No1	65.0%	31	16.40	2.6%	9	14.30
	No2	85.4%	71	13.87	27.0%	51	11.61
	Po1	91.4%	14	15.12	4.8%	2	13.99
	Po2	84.2%	55	12.73	7.2%	39	10.95
M=2	Ge1	98.4%	19	17.95	13.0%	2	22.91
	Ge2	84.6%	43	19.16	62.8%	32	21.27
	No1	76.4%	4	23.63	51.0%	0	25.82
	No2	86.0%	37	19.98	59.2%	26	22.57
	Po1	98.2%	0	25.11	38.2%	0	24.54
	Po2	98.8%	42	16.70	39.6%	9	30.29

Table 3. Comparison with a sequential approach

	M=1			*M*=2		
	ΔS	*ΔK*	*ΔD*	*ΔS*	*ΔK*	*ΔD*
Ge1	-3.6%	-38.8%	5.4%	-0.6%	-60.4%	1.5%
Ge2	1.4%	-8.0%	-1.7%	-12.2%	-32.8%	7.3%
No1	5.5%	-3.1%	2.0%	3.5%	0.0%	-9.4%
No2	-3.2%	-5.3%	0.2%	-10.4%	-32.7%	8.6%
Po1	0.0%	0.0%	0.0%	34.5%	0.0%	-14.4%
Po2	9.1%	-26.7%	-3.4%	-0.4%	-37.3%	24.1%
Avg	1.5%	-13.7%	0.4%	2.4%	-27.2%	3.0%

VI. Conclusions

In this paper, we addressed the channel segmentation and buffer insertion problem for FPAAs. Staggering factors are introduced to provide optimization on terms of routability, interconnect delay and area costs. Experiments show that the combined approach can significantly reduce the total number of buffers required, while improving the routability and minimizing the interconnect delay.

References

[1] J. Rose and D. Hill, "Architectural and physical design challenges for one-million gate FPGA's and beyond," in *ACM/SIGDA Int. Symp. FPAAs*, Monterey, CA, Feb. 1997, pp. 129-132.

[2] Kaushik Roy and Sudip Nag, "Automatic Synthesis of FPGA Channel Architecture for Routability and Performance," *IEEE Tran. on VLSI Systems*, vol. 2, pp. 508-511, December 1994.

[3] El Gamal, J. Greene, and V. Roychowdhury, "Segmented channel routing is nearly as efficient as channel routing (and just as hard)," *Proc. Advanced Research VLSI*, Santa Cruz, CA, pp. 193-221, 1991.

[4] K. Zhu and D.F. Wong, "On channel segmentation design for row-based FPGAs," *Proc. ICCAD*, pp. 26-29, 1992.

[5] M. Pedram, B. S. Nobandegani, and B. T. Preas, "Design and analysis of segmented routing channels for row-based FPGAs," *IEEE Trans. on Computer Aided Design*, vol. 13, no. 12, pp. 1470-1479, Dec. 1994.

[6] V. Betz and J. Rose, "Circuit design, transistor sizing and wire layout of FPGA interconnect," *Proc. CICC*, pp. 171 - 174, 1999.

[7] C. Alpert and A. Devgan,"Wire Segmenting for Improved buffer Insertion," *Proc. DAC*, pp. 588-593, 1997.

[8] Jai-Ming Lin, Song-Ra Pan and Yao-Wen Chang, "Graph matching-based algorithms for array-based FPGA segmentation design and routing," *Proc. ASP-DAC*, pp.851-854, 2003

[9] W. K. Mak and D. F. Wong, “Channel segmentation design for symmetrical FPGAs,” *ICCD*97, pp. 496–501.

[10] Semiconductor Industry Association, *National technology Roadmap for Semiconductors*, 1997.