Late Breaking Results : Scan-chain Optimization with Constrained Single Linkage Clustering and Geometry-based Cluster Balancing

George Antony
IBM Systems, Bangalore

Gireesh Kumar K M
IBM Systems, Bangalore

Naiju Karim Abdul
IBM Systems, Bangalore

Rahul M Rao
IBM Systems, Bangalore

***Abstract*—Scan chain optimization is an optimization step in the physical design flow where connections between placed scannable elements are re-ordered to reduce the total wirelength, there-by improving routability and power. In this paper we present a hierarchical clustering technique using single linkage followed by geometry-based balancing to adhere to test requirements while improving scan chain wirelength. The proposed method is compared against commonly used K-means clustering method and demonstrates a maximum improvement of around 10% for certain design topologies.**

Keywords—Physical design, Single linkage clustering, K-means clustering, geometry-based balancing.

I. Introduction

Scan chains are used in Design For Test (DFT) as they provide means to control and observe the states of various registers in the design. Scan chains are formed as serial connection of registers and to manage scan test time a design is partitioned into several independent scan chains. Since the order in which registers are connected in a scan chain does not reduce test coverage, these connections can be re-ordered during physical design to reduce wirelength of scan nets. Scan optimization (ScanOpt) can be classified as multiple travelling salesman problem in which set of *'n'* locations (registers) are to be visited by *'m'* salesman (scan chains) while minimizing the combined distance travelled by all salesmen. Along with wirelength minimization, there are additional DFT constraints on ScanOpt to improve diagnosability, improve fault isolation and reduce test time.

Classical methods like genetic algorithm [1] and min cost flow [2] have been used for ScanOpt, but the DFT constraints (in Section II) introduce additional complexities for these solutions. Hence, we explored clustering algorithms like partitioning clustering and hierarchical clustering. Authors of [3] uses partition clustering using centroid-based k-means algorithm with honoring DFT constraints. But it can result in sub-optimal results for certain latch topologies, including vertical and horizontally stacked latches and using hierarchical clustering with single linkage has potential to outperform [3].

In this paper, we propose a two-step approach for creating the scan chain clusters. The first step creates clusters using constrained single linkage (Single-Linkage+) method. The second step involves a geometry-based balancing of clusters.

II. Methodology

We utilize a two-step approach for creating and balancing the scan chain clusters.

A. Grouping the registers

First step in clustering the scan registers is to group registers with certain properties as a single node. The criteria used for grouping are as follows:

1) All registers connected to a single Local Clock Buffer(LCB) are considered as a single group. This minimzes the number of fault diagnostic elements which are inserted at every LCB crossing reducing the number of design elements.

2) All registers in non-optimzable sequences are considered as a single group. Non-optimzable sequences represent specific groups of registers whose scan order needs to be maintained as specficed in the RTL. Considering them as a single group ensures that this constraint is honored.

3) Hard IPs and small blocks are considered as a single group, with knowledge of the number of registers in the scan chain inside the IP. The scan chains inside the hard IPs and small blocks are optimized separately. The number of registers in a scan chain is passed to the higher level for optimization, so that DFT constraint for the regsiter count limit can be honored.

Figure 1 depicts a sample design where each dot represents a group of registers. The DFT constraints used in this work are:

1) ***n-constraint*** : Maximum register count per scan chain should be less than or equal to DFT scan length limit.

2) ***ϵ-constraint*** *:* The maximum distance between two clusters that are merged should not exceed a user specified limit.

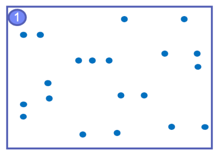

Fig. 1: Sample design where each dot represents a group of registers.

3) ***must-link constraint*** *:* Regsiters connected to a Local Clock Buffer and regsiters marked are non-reconnectable should remain together in a cluster

B. Merge Clusters using single linkage honoring the constraints

The next step is to merge these groups to create clusters using a modified version of Single linkage clustering technique. The number of clusters created in this step will be greater than or equal to the number of scan chains (*m)* in the design.

Single Linkage clustering [4] is a method of hierarchical clustering where two clusters with the smallest distance between each other are merged to create a new cluster in each iteration. We need to modify single linkage clustering to honor *n-constraint* listed above by limiting merging of clusters that violate this constraint. Compared to the K-means clustering method of [3], single linkage clustering offers unique advantage for certain designs. 1) Since single linkage clustering inherently groups closest nodes together, it has the potential to create lowest wirelength connection when the connections are optimized. 2) While k-means is better suited for radial register configurations, in several designs, registers are often vertically or horizontally aligned to facilitate design

979-8-3503-4859-0/DATE24/© 2024 EDAA

data flow. In such cases, single linkage clustering could perform better.

Figure 2 depicts how clusters are merged using single linkage clustering. Assume *n-constraint* is 5, *ϵ-constraint* is 6 and number of scan chain (*m*) is 4. For simplicity, let us assume that each dot represents one register. As the maximum distance that we can merge is 6, based on *ϵ-constraint*, we start with merging registers from distance 1 to 6, assuming step size of 1. In Steps 2 & 3, we create clusters of all latches with distance less than 1. As none of the groups have more than 5

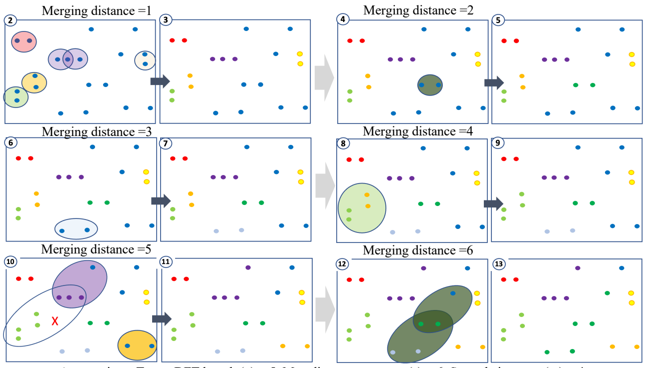

Fig. 2: Merging Clusters using constrained single linkage.

elements, as constrained by *n-constraint*, we get 5 groups after this assignment. In the next step, we merge groups with distance limit of 2 and we get 6 groups after this assignment. In the next step, merging distance is increased to 3, and we get one more group as shown in Steps 6 & 7. When the merging distance is increased to 4, two of the existing groups are merged as one, as it had a total of 4 registers together and it satisfies the *n-constraint*. In the next step with merging distance as 5, we had 3 options to merge. But, as the merged element will have 8 registers in one group which will violate *n-constraint*, we merge only two groups in this step. In the last iteration with merging distance of 6, we merge one more cluster, and the remaining ones are separated as individual clusters. After this step, we have 7 clusters which is more than the required number of scan chains (4).

C. Balance the Clusters honoring the constraints

To balance these clusters to scan chain limit, we first convert all clusters to geometric shape encompassing all nodes. Then a line is drawn from smallest cluster to the nearest free cluster. Registers are then balanced through all the clusters that touches this line. Figure 3 depicts balancing of clusters with geometry-based movement. Step 1 shows geometric shape representation of all clusters from the previous step. In Step 2, the smallest cluster with one register needs to be merged with one of the nearest clusters. Since the nearest cluster is already filled with 5 registers, meeting the *n-constraint*, we pick the second nearest cluster. As it has space to accommodate one register, we merge these two clusters. We repeat the same in Step 4 and 5. In Step 6, the line from smallest cluster to nearest free cluster goes thorough 2 other clusters. Here we move one register from each of these clusters as shown in Step 7. The outcome of that movement is show in Step 8. Since we have 5 clusters now, and the maximum number of clusters is 4, we repeat the movement one more time to create 4 clusters as shown in Step 9. The number of registers moved across clusters is determined based on the number of elements in the smallest cluster and free space available in the nearest cluster. If the number of scan clusters after this iteration is still greater than the scan chain

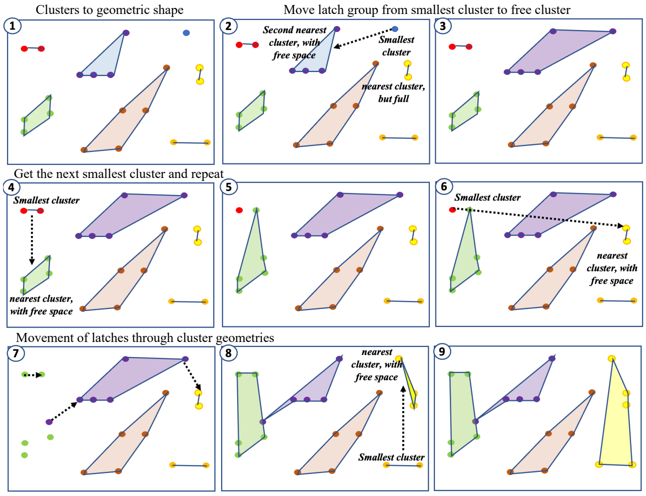

Fig. 3: Geometry based cluster balancing.

count m, we relax the grouping criteria in A.1 as a soft-constraint and repeat balancing of clustering.

III. Evaluation and Results

The proposed approach underwent evaluation across 168 designs, all of which were based on a high-performance processor implemented in sub-7nm technology. These designs encompass a broad spectrum of physical dimensions, with aspect ratios (width/height) ranging from 0.1 to 9.2, and the number of registers ranging from 40 to 60K. During the single linkage clustering, a maximum limit (*ϵ-constraint)* of 20um was set for register movement across the clusters. The overall scan wire length in the design is calculated as the cumulative sum of wire lengths from different scan chains within the design. Out of the 168 designs we tested, 68 of them have demonstrated better scan wire lengths compared to K-means clustering, with improvements ranges from 0.1% to 12%. Table 1 shows the result in 5 representative designs.

Table 1: Comparison in 5 representative designs

Designs	Width x height	Register count	Total Scan Wire Length		WL Improvement	
			K-Means	Single Linkage+	Wire Length	%
Design 1	341x451	39504	2105352	1855270	250082	11.88%
Design 2	49x510	2609	528786	476568	52218	9.88%
Design 3	61x104	1076	44941	42385	2556	5.69%
Design 4	99x399	3222	187789	180063	7726	4.11%
Design 5	245x300	2889	201679	218625	-16946	-8.40%

Figure 4 shows the comparison of K-means to Single-Linkage+ when the register groups are placed vertically in a design with 4 scan chains. K-means clustering will group the registers in a radial manner, which could result in crisscross connections between the vertical structures, whereas Single Linkage+ clusters align with the vertical arrangement, thereby creating a reduced wirelength solution.

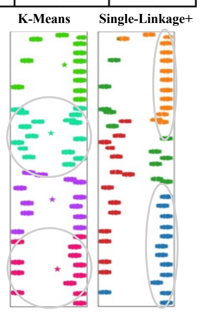

Fig. 4: Kmeans vs Single-Linkage+

IV. References

[1] B. B. Paul, R. Mukhopadhyay and I. S. Gupta, "Genetic Algorithm based Scan Chain Optimization and Test Power Reduction using Physical Information," *TENCON 2006 - 2006*

[2] S. Kobayashi, M. Edahiro and M. Kubo, "Scan-chain optimization algorithms for multiple scan-paths," *Proceedings of 1998 ASPDAC*,

[3] N. K. Abdul, G. Antony, R. M. Rao and S. T. Skariah, "Scan Chain Clustering and Optimization with Constrained Clustering and Reinforcement Learning," *2022 MLCAD*, pp. 83-89

[4] https://en.wikipedia.org/wiki/Single-linkage_clustering