Sequence Pair Based Voltage Island Floorplanning

Dipanjan Sengupta[1], Andreas Veneris[1], Steve Wilton[2], Andre Ivanov[2], Res Saleh[2]

***Abstract*—In the nanometer era of VLSI design, high power consumption is considered to be a "show-stopper" for many applications. Voltage Island design has emerged as a popular method for addressing this issue. This technique requires multiple supply voltages on the same chip with blocks assigned to different supply voltages. Implementation challenges force blocks with similar supply voltages to be placed contiguous to one another, thereby creating "islands". Classical floorplanners assume a single supply voltage in the entire SoC and thus require additional design steps to realize voltage islands. In this paper we present a new floorplanning algorithm based on the sequence pair representation that can floorplan blocks in the form of islands. Given the possible supply voltage choices for each block, the floorplanner simultaneously attempts to reduce power and area of the chip. Our floorplanner integrates the tasks of assigning blocks to different supply voltages and the placing of the blocks in the chip. Compared to previous work, the proposed floorplanner on average reduces the area overhead of the chip by 13.5% with 34% runtime improvement. Additionally we explore the tradeoff between power and area for different floorplan solutions.**

I. Introduction

For over four decades, semiconductor industry has witnessed geometric feature-size reduction with resultant increase in chip capacity and performance. With every technology generation, improved performance and density has designers and consumers to enjoy the benefit of faster, cheaper and yet more complex chips. In the past, the principal goal of chip designers was to deliver the maximum possible performance, often at the expense of area and power. However, with continued technology scaling, the levels of integration have pushed the power profiles of many systems beyond acceptable power density limits. The pursuit of efficient methods for reduction in power consumption has moved to the forefront of the VLSI design challenge.

One of the popular techniques for power reduction is the use of multiple supply voltages (in the form of *Voltage Islands*). In [1], Lackey et al first proposed a block-based multi-supply design using voltage islands. This technique requires IP blocks with same supply voltage to form contiguous groups called "islands".

Voltage island design can be implemented at either the floorplan/placement stage or post-floorplan/post-placement stage. Voltage island design challenges at the floorplan/placement stage have been studied extensively in [2]–[5]. An approach involving perturbation for voltage assignment of cores, and simulated annealing for floorplanning is described in [2]. A similar approach is taken in [3] with the additional objective of achieving a thermally-balanced, island-based System-on-Chip (SoC) design. In [4], based on the block timing requirements, voltage assignment is performed first. Next, level shifters are inserted and floorplanning is performed using a power network aware floorplanner such that the sum of perimeters of voltage islands is minimized. Fine-grained voltage assignment during placement of standard cells was proposed in [5].

In [6]–[9], voltage islands are created from fixed floorplans. In [6], [8], placement adjacencies of logic blocks are considered to create voltage islands to optimize power versus design-cost tradeoffs under performance constraints In [7], a 0-1 integer-linear programming method is proposed for voltage assignment to the cores in an SoC. Next, from a finite choice of floorplan solutions, a candidate floorplan that respects the voltage assignment is chosen. In [9], a zero-slack algorithm is used to perform delay budgeting and voltage assignment. Based on physical proximity, the actual voltage islands are then created.

The objective of classical floorplanners is to pack all hard/soft blocks, such that the enclosing layout region is minimum area (with no overlapping blocks) while satisfying the aspect ratio of each block as well the timing requirement of the entire chip. With the emergence of fixed-outline floorplanner, multi-objective minimization of area and wirelength, via linear combinations, was no longer an important design issue [10]. As mentioned in [11], classical floorplanners fall short of addressing new circuit challenges such as power. In all the previous work [2]–[9], power optimization and floorplanning has been considered as two distinct steps. Simultaneous voltage assignment, voltage island generation and floorplanning using Normalized Polish Expression was first considered in [12]. However, island partitioning and voltage assignment is performed for every candidate floorplan.

Our main contributions are summarized as follows:

- In this paper we propose a new voltage island aware floorplanning technique using Sequence pair representation. Simulated Annealing is used for searching the solution space.
- To ensure the proper construction of voltage islands, certain constraints have been imposed on the position of the blocks in the sequence pair. Unlike [12], we do not require island partitioning and voltage assignment for every candidate floorplan, thereby saving runtime. Results show that our floorplan solution require less area, lower power and has a faster run-time than previous proposals.
- In our floorplanning method, any candidate floorplan will always provide a feasible solution.
- As power and area/wirelength have to be jointly reduced, we explore floorplan solutions that have multiple objectives. Based on the maximum allowable deadspace and power, different solutions can be explored by the designer using proper choice of relative weights of area and power.

The remainder of the paper is organized as follows. The floorplan representation using sequence pair is described in Section II. In Section III, we propose a new sequence pair representation method in order to incorporate Voltage Islands. Section IV discusses the overall design flow. Experimental results are provided in Section V and conclusions are summarized in Section VI.

[1]University of Toronto, ECE Department, Toronto, ON M5S 3G4 (dipanjan.sengupta@utoronto.ca, veneris@eecg.utoronto.ca)
[2]University of British Columbia, ECE Department, Vancouver, BC, V6T 1Z4({stevew, ivanov, res}@ece.ubc.ca)
978-1-4577-1221-0/11/$26.00 ©2011 IEEE

II. Floorplan Representation

Simulated Annealing (SA) is one of the most popular and widely-used approaches for floorplan optimization. Floorplanners differ in the internal floorplan representation. Topological representation of the floorplan is critical to the effectiveness and optimization of the algorithm. VLSI floorplans representations can be classified into two categories [13]: (i) slicing floorplan and (ii) non-slicing floorplan. The sequence pair representation falls under the second category of floorplan representation.

A sequence pair is a pair of sequences of N elements representing a list of N blocks [14]. Each sequence consists of permutation of the N blocks. The two permutations define the geometric relations between each pair of non-overlapping blocks as follows:

$$\begin{array}{l} (<..,a,...,b,..>,<..,a,...,b,..>) \Rightarrow a \text{ is to the left of } b \\ (<..,a,...,b,..>,<..,b,...,a,..>) \Rightarrow a \text{ is above } b \end{array} \qquad (1)$$

The placement of individual blocks in the floorplan is computed by translating the sequence pair (X, Y) to the horizontal (G_H) and vertical (G_V) constraints graphs. Each graph consists of $(N + 2)$ vertices – one for each N blocks and a source and sink node to represent the boundaries. A directed edge (a, b) exists in G_H if block a is placed to the left of block b. Similarly a directed edge (b, a) exists in G_V if block a is above block b. Vertices without any outgoing edge are connected to the sink, and vertices without incoming edges are connected to the source. The weight of each vertex in G_H and G_V represent the width and height of the block respectively. A longest common subsequence ($lcs()$) algorithm can be applied to determine the individual location of each block as well as the overall dimension of the chip. The width of the chip can be computed using the function $lcs(X, Y)$ and the height of the chip can be computed using the function $lcs(X^R, Y)$ where X^R is the reverse of X. Such a computation can be performed in $O(n^2)$ time using algorithm in [14] and in $O(nlogn)$ time using the more sophisticated data structure in [15].

III. Contiguous Constraint

One of the primary constraints for voltage island design is to have blocks within the same island (i.e. operating at the same supply voltage) contiguous to one another to form *islands*. Such a constraint simplifies the power grid design for each supply voltage in the chip. Unlike H-alignment and V-alignment [16] the contiguous constraint has greater freedom of placement as the relative positions of the blocks are not strictly constrained. Similar constraints have been considered in [17] for B* Tree representation of multivoltage floorplans.

A pair of blocks can be contiguous to one another horizontally (H-contiguous) or vertically (V-contiguous). **H-contiguous** can be formally defined as follows:

Definition 1 *Given k blocks, b_i, $i = 1, 2, \ldots, k$ with dimension $w_i \times h_i$ and coordinates (x_i, y_i) referring to the bottom-left corner for each block; $i = 1, 2, \ldots, k$ respectively, the k blocks are H-contiguous iff $x_i + w_i = x_{i+1}$ (horizontal constraint) and either $y_{i+1} = y_i$ or $y_i + h_i > y_{i+1}$ if $y_i < y_{i+1}$ or $y_{i+1} + h_{i+1} > y_i$ if $y_{i+1} < y_i$ (vertical constraint), $1 \leq i \leq k - 1$.*

The definition of **V-contiguous** can be written in a similar fashion as:

Definition 2 *Given k blocks, b_i, $i = 1, 2, \ldots, k$ with dimension $w_i \times h_i$ and coordinates (x_i, y_i) referring to the bottom-left corner for each block; $i = 1, 2, \ldots, k$ respectively, the k blocks are V-contiguous iff $y_i + h_i = y_{i+1}$ (vertical constraint) and either $x_{i+1} = x_i$ or $x_i + w_i > x_{i+1}$ if $x_i < x_{i+1}$ or $x_{i+1} + w_{i+1} > x_i$ if $x_{i+1} < x_i$ (horizontal constraint), $1 \leq i \leq k - 1$.*

In order to assign blocks to the same island in the sequence pair, we consider the H-contiguous case first. From (1), blocks a and b are H-contiguous in sequence pair $(X, Y) = (\ldots aX_2b\ldots, \ldots aY_2b\ldots)$ if a and b maintain the same order. Additionally, a must be strictly ahead of b, i.e. there should be no common block in (X_2, Y_2) [16], i.e. $lcs(X_2, Y_2) = 0$. Blocks a and b can reside in the same island if function $H - Island(a, b)$ is true. The function is described below:

Algorithm 1 H-Island

```
1:  $(X, Y) = (\ldots aX_2b\ldots, \ldots aY_2b\ldots)$;
2:  if $(X = \emptyset)$ and $(Y = \emptyset)$ then
3:      return $true$;
4:  else if $lcs(X_2, Y_2) = c$ then
5:      $(X, Y) = (\ldots a\ldots c\ldots b\ldots, \ldots a\ldots c\ldots b\ldots)$;
6:      if $(V_{ab} = V(c))$ then
7:          return $true$;
8:      else
9:          return $false$;
10:     end if
11: else
12:     if $(X_2, Y_2) = (cX_3d, cY_3d)$ then
13:         if $(V_{ab} = V(c))$ and $(V(c) = V(d))$ then
14:             H-Island$(X_2, Y_2)$;
15:         else
16:             return $false$;
17:         end if
18:     else
19:         return $false$;
20:     end if
21: end if
```

In the function **H-Island(a,b)** three different scenarios have been considered. The pair of blocks under consideration (a, b) may have no common block between them, have a single block c, or may have multiple blocks (cX_3d, cY_3d) placed between them. Here V_{ab} represent the voltage in which both a and b can reside and $V(c)$ and $V(d)$ are legal voltage assignments of c and d respectively. For multiple blocks the function is recursively called in order to ensure that all blocks between a and b reside in the same island and are horizontally contiguous. This can be formally defined as follows:

Theorem 1 *Given two blocks a and b in sequence pair $(X, Y) = (X_1aX_2bX_3, Y_1aY_2bY_3)$, a and b can be assigned to the same island iff. **H-Island(a,b)** is true.*

If $lcs(X_2, Y_2)$ is empty then a and b are abutted to one another whereas when $lcs(X_2, Y_2)$ is non-empty, there are blocks between a and b in the horizontal direction and they must be assigned to the same island.

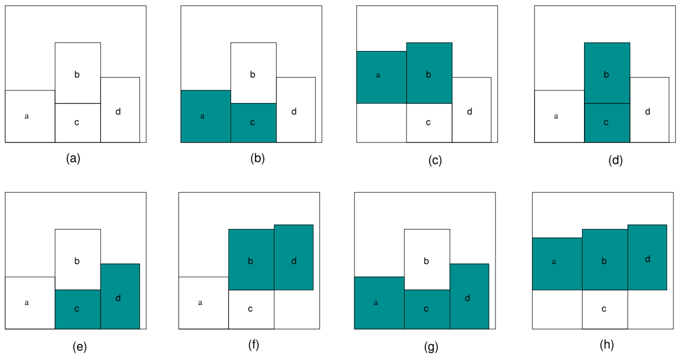


Fig. 1. Floorplan for sequence pair $(XY) = (< a, b, c, d >, < a, c, b, d >)$ (a) without islands (b) island (a, c) (c) island (a, b) (d) island (b, c) (e) island (c, d) (f) island (b, d) (g) island (a, c, d) (h) island (a, b, d)

Example 1 *For the sequence pair representation* $(X, Y) = (< a, b, c, d >, < a, c, b, d >)$*, shown in Fig. 1(a), voltage islands can be created between blocks* (a, c) *(Fig. 1(b)),* (a, b) *(Fig. 1(c)),* (b, c) *(Fig. 1(d)),* (c, d) *(Fig. 1(e)) and* (b, d) *(Fig. 1(f)). As* $lcs(bc, cb) = b, c$*, blocks* a *and* d *can be placed in the same island iff either* b *or* c *is placed in the same island as well, thus forming the island* (a, c, d) *(Fig. 1(g))or* (a, b, d) *(Fig. 1(h)).*

Extending the sequence pair representation to multiple voltage islands for H-contiguous constraint leads to:

Theorem 2 *For a given floorplan with* N *blocks and* k *distinct voltage islands, represented by the Sequence Pair* $(X, Y) = (X_1X_2X_3 \dots X_k, Y_1Y_2Y_3 \dots Y_k)$*, where* X_i *and* Y_i*,* $1 \leq i \leq k$*, is a permutation of blocks in voltage island* i *such that each pair of block* (a_j, b_j)*, in voltage island* i*, satisfies* $H - Island(a_j, b_j)$*.*

Any sequence pair that satisfies the above condition can be floorplanned as islands in the horizontal direction.

Example 2 *Fig. 2(a) shows the floorplan of 6 blocks placed in 2 islands where* $X_1 = Y_1 = (1, 2, 3)$ *and* $X_2 = Y_2 = (4, 5, 6)$ *and the sequence pair is* $(X, Y) = (X_1X_2, Y_1Y_2)$*. Such a representation not only places blocks within the same island horizontally contiguous to one another but also ensures that islands are not placed within one another. For example, the sequence pair* $(X, Y) = (< 1, 4, 5, 6, 2, 3 >, < 1, 4, 5, 6, 2, 3 >)$ *would create the island* $4, 5, 6$ *within the island* $1, 2, 3$*, as shown in Fig. 2(b). In reality blocks* $1, 2, 3$ *would be assigned to chip-level supply voltage while* $4, 5, 6$ *would have a different voltage. But such a floorplan would complicate power grid routing, buffer insertion etc. and thus are avoided.*

Thus **Theorem 1** and **Theorem 2** enable us to create voltage islands with islands and blocks **H-contiguous** to one another.

A similar analysis is possible for the V-contiguous property. Here we consider the sequence (X^R, Y) instead of (X, Y). From (1), blocks a and b are V-contiguous in the sequence $(X, Y) = (\dots aX_2b \dots, \dots bY_2a \dots)$ and there is no common block between a and b, i.e. $lcs(X_2, Y_2) = 0$. Now, blocks a and b can reside in the same island if function **V-Island(a,b)** is true. Function **V-Island(a,b)** is similar to **H-Island(a,b)** except that it is concerned with the placement of the blocks vertically as opposed to horizontally.

Algorithm 2 V-Island

```
1: $(X, Y) = (\dots aX_2b \dots, \dots aY_2b \dots)$;
2: if $(X_2 = \emptyset)$ and $(Y_2 = \emptyset)$ then
3:     return $true$;
4: else if $lcs(X_2, Y_2) = c$ then
5:     $(X, Y) = (\dots a \dots c \dots b \dots, \dots a \dots c \dots b \dots)$;
6:     if $(V_{ab} = V(c))$ then
7:         return $true$;
8:     else
9:         return $false$;
10:    end if
11: else
12:    if $(X_2, Y_2) = (cX_3d, cY_3d)$ then
13:        if $(V_{ab} = V(c))$ and $(V(c) = V(d))$ then
14:            V-Island($X_2^R, Y_2$);
15:        else
16:            return $false$;
17:        end if
18:    else
19:        return $false$;
20:    end if
21: end if
```

Theorem 3 *Given two blocks* a *and* b *in sequence pair* $(X, Y) = (X_1aX_2bX_3, Y_1aY_2bY_3)$*,* a *and* b *can be assigned to the same island iff.* ***V-Island(a,b)*** *is true.*

Sequence pair representation for multiple voltage islands which satisfy the V-contiguous property can be formalized as:

Theorem 4 *For a given floorplan with* N *blocks and* k *distinct voltage islands, represented by the Sequence Pair* $(X, Y) = (X_1X_2X_3 \dots X_k, Y_1Y_2Y_3 \dots Y_k)$*, where* X_i *and* Y_i*,* $1 \leq i \leq k$*, is a permutation of blocks in voltage island* i *such that each pair of block* (a_j, b_j)*, in voltage island* i*, satisfies* $V - Island(a_j, b_j)$*.*

Example 3 *Fig. 3(a) shows the floorplan of the blocks in Fig. 2(a) with sequence pair* $(X, Y) = (< 6, 5, 4, 3, 2, 1 >, < 1, 2, 34, 5, 6 >)$*. As in the case of H-contiguous, islands must not be placed within another island as well (Fig. 3(b)) and thus we avoid sequences such as* $(X, Y) = (< 3, 2, 6, 5, 4, 1 >, < 1, 4, 5, 6, 2, 3 >)$*.*

Voltage Islands have blocks that are contiguous to one another in any direction i.e. *some blocks can be H-contiguous while others can be V-contiguous*. In order to ensure that all blocks in the same island are contiguously placed to form an

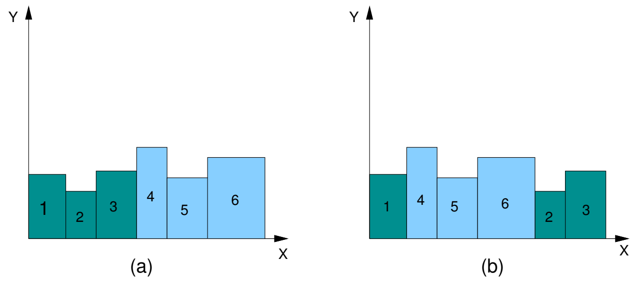


Fig. 2. Placement of islands that (a) satisfy H-contiguous property and (b) donot satisfy H-contiguous property

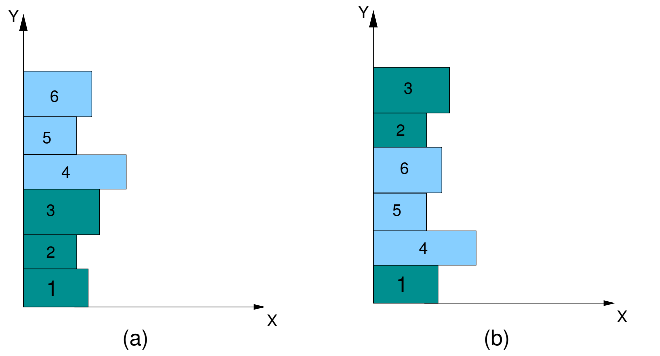


Fig. 3. Placement of islands that (a) satisfy V-contiguous property and (b) donot satisfy V-contiguous property

island we ensure that the function **Island(a,b)** is true, where blocks a and b reside in the same island.

Algorithm 3 Island(a,b)

```
1: $(X, Y)$ is the Sequence Pair representation of a floorplan
2: $a$ and $b$ are two distinct blocks in the floorplan
3: $P = \emptyset$
4: $c \leftarrow a$
5: $d \leftarrow b$
6: if $(X, Y) = (\ldots dX_1c\ldots, \ldots dY_1c\ldots)$ then
7:     $swap(c, d)$
8: end if
9: if $(X, Y) = (\ldots cX_1d\ldots, \ldots cY_1d\ldots)$ then
10:     if H-island(c,d) $= 1$ then
11:         return $true$;
12:     else
13:         return $false$;
14:     end if
15: else if $(X, Y) = (\ldots cX_1d\ldots, \ldots dY_1c\ldots)$ then
16:     if V-island(c,d) $= 1$ then
17:         return $true$;
18:     else
19:         return $false$;
20:     end if
21: end if
22: $P = P \cup \{$all univisited blocks H-contiguous to $a\} \cup \{$all unvisited blocks V-contiguous to $a\}$
23: for all $e$, where $e \in P$ do
24:     $a = e$
25:     goto 4
26: end for
```

In **Island(a,b)**, blocks a and b are assigned to two temporary variables c and d. As the relative positions of blocks a and b in (X, Y) are not known **Island(a,b)** first swaps c and d for two specific cases. It then determines if the blocks are horizontally contiguous or vertically contiguous. If either of these conditions is true we can deduce that blocks a and b are contiguous to one another in the floorplan of the island. If the blocks are not horizontally/vertically contiguous then the set of blocks that are horizontally/vertically contiguous to a are noted in the set P. For every block e in set P, the function attempts to determine if it is horizontally/vertically connected to b. The function either returns $true$ or $false$ when set P is empty. Thus the necessary condition for sequence pair representation of blocks in Voltage Islands is as follows.

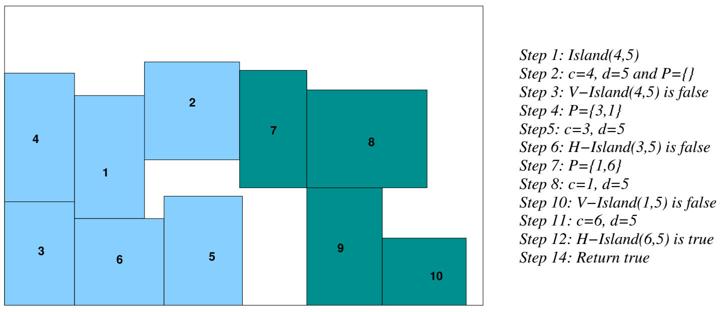

4
1
2
7
8
3
6
5
9
10
Step 1: Island(4,5)
Step 2: c=4, d=5 and P={}
Step 3: V−Island(4,5) is false
Step 4: P={3,1}
Step5: c=3, d=5
Step 6: H−Island(3,5) is false
Step 7: P={1,6}
Step 8: c=1, d=5
Step 10: V−Island(1,5) is false
Step 11: c=6, d=5
Step 12: H−Island(6,5) is true
Step 14: Return true

Fig. 4. Steps to check block 5 is reachable from block 4 through contiguous blocks in Island 1

Theorem 5 *For a given floorplan with N blocks and k distinct voltage islands, represented by the Sequence Pair $(X, Y) = (X_1X_2X_3\ldots X_k, Y_1Y_2Y_3\ldots Y_k)$, where X_i and Y_i, $1 \leq i \leq k$, is a permutation of blocks in voltage island i such that each pair of block (a_j, b_j), in voltage island i, satisfies $Island(a_i, b_i)$.*

Example 4 *Consider the floorplan of 10 blocks in two islands as shown in Fig. 4. The sequence pair representation of the blocks are $(X, Y) = (< 4, 3, 1, 6, 2, 5, 7, 8, 9, 10 >, < 3, 6, 5, 4, 1, 2, 7, 9, 10, 8 >)$. Blocks 1-6 are in $Island$ 1 and the rest of the blocks are in $Island$ 2. The steps to determine if the blocks in $Island$ 1 are contiguous consists of determining if blocks 5 can be reached from block 4 via contiguous blocks in $Island$ 1.*

Note that the blocks in the same island being placed together in the sequence pair negates certain types of floorplans to be realizable. For example, blocks $1, 2, 3$ in Fig. 5 can be placed in $Island$ 1 while blocks $4, 5, 6$ can be placed in $Island$ 2. But such floorplan will not be created due to the constraints placed on each block in an island.

IV. Overall Design Flow

We assume that the chip contains a set of N blocks with areas $A_1, A_2, \ldots, A_N$. Additionally the aspect ratio of each block i has a minimum and maximum bound $[l_i, u_i]$, $1 \leq i \leq N$. It must be noted that in case of hard blocks the value of the maximum and minimum aspect ratio is same and cannot be altered during floorplanning. The supply voltage choice for each block is specified in the Voltage Assignment Table [18].

The proposed floorplanner uses the sequence pair representation for the floorplan. The sequence pair is initialized by assigning each blocks in the lowest possible voltage level, and is of the form $(X, Y) = (X_1X_2X_3\ldots X_k, Y_1Y_2Y_3\ldots Y_k)$,

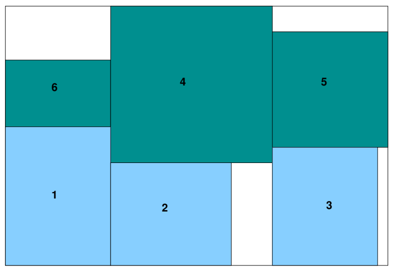


Fig. 5. Floorplan not realizable due to constraint in the sequence pair pattern

where X_i and Y_i is the permutation of blocks whose minimum possible supply voltage is $V_{DD}(i)$. The floorplanner relies on Simulated Annealing framework [19].

A. *Moves*

Within a simulated annealing iteration, there are four kinds of moves generated. Each such move produces a new floorplan solution.

1) **Swap Islands**: The position of the islands in either or both sequences can be swapped. For example, in the sequence pair (X, Y), if island 1 and 3 are swapped only in the X sequence then the new sequence pair would be $(X', Y') = (X_3X_2X_1 \ldots X_k, Y_1Y_2Y_3 \ldots Y_k)$.
2) **Swap Blocks within an Island**: Blocks b_{ji} and b_{ki}, assigned to voltage island i, can be swapped in either or both the sequences.
3) **Swap Blocks between two Islands**: Assuming the following relation holds true, $V_{DD}(i) < V_{DD}(l)$, block b_{ji}, in voltage island i, is swapped with block b_{kl}, in voltage island l, if and only if block b_{kl} can reside in voltage island i. Such a swap is performed in both the sequences.
4) **Create & Merge Island**: Unlike previous move, one block is moved from one island to another such that either a new island is created or one island is merged with an existing island. The move can change the relative position of a block in either of the sequences.

In addition to all these moves we can also change the aspect ratio of the soft blocks during floorplanning.

B. *Cost Function*

The cost function used to evaluate a floorplan can be written as:

$$cost = \alpha * A + \beta * W + \gamma * P \qquad (2)$$

where A is the chip area, W is the total wirelength and P is the total power consumption of the chip. The values of the parameters α, β and γ can be assigned by the designer [3] or be set using random walks at the beginning of the annealing process [12]. In regards to the inclusion of fixed-outline floorplan constraint, sequence pairs that do not meet the chip width/height are initially considered to be legal solution. But as the annealing progresses such solutions are disregarded by assigning a very large value to both A and W.

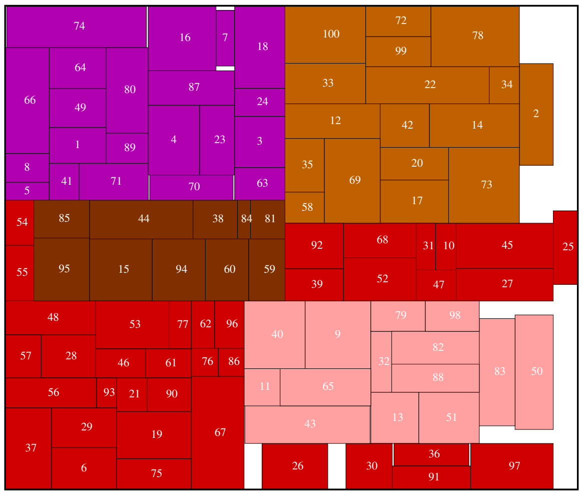
C. *Annealing Schedule*

The initial temperature is set to be very high ($T_{initial}$ = 100000) for each floorplan solution. A geometric cooling schedule [10] with a varying cooling factor is used.

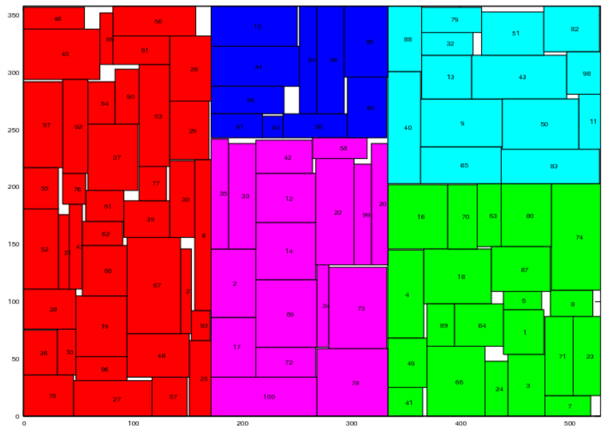
The **Swap Islands** move is more likely in the higher temperatures while swapping of blocks within the islands and changes in aspect ratio are more likely in the lower temperatures. The remaining move types have equal probability throughout the cooling process. Simulations show that swapping islands has a greater effect on the overall area of the floorplan while the other move types have a smaller impact on the total area.

V. Results

Experiments were performed using GSRC benchmark circuits. The proposed floorplanner was implemented in C++ and complied with gcc version 4.3.2. All experiments were run on Intel P4 CPU 2.8 GHz workstation with 2GB RAM.



(a) Previous Work [12]



(b) Proposed Work

Fig. 6. Floorplan Comparison

The area and runtime of the proposed floorplanner was compared to the previous related floorplanner in [12]. Table I compares the runtime and deadspace of the proposed floorplanner with the previous work in [12]. Here α, β and γ values are equal to 1. As shown in the table, in most of the cases, our algorithm finds a better floorplan solution in terms of area and requires much less run-time. Fig. 6(a) and Fig. 6(b) show the floorplan solutions of n100 with 5 islands using [12] and our proposed algorithm respectively.

In order to explore the floorplan solutions with different power and area/wirelength objectives, the cost function in eqn.(2) is modified to:

$$cost = \beta * (\alpha * A + (1 - \alpha) * W) + (1 - \beta) * P \qquad (3)$$

Table II compares the area, power and wirelength of each design with different values of α and β. Depending upon the relative weigth of each cost factor the area and power values differ. Solutions with $\beta = 0.5$ are better in terms of power while solutions with $\beta = 0.75$ have less area and wirelength. Proper choice of these values can deliver the suitable floorplan satisfying both the area and power budget. In order to implement voltage islands such multi-objective cost functions

TABLE I
AREA AND RUNTIME COMPARISON

Circuit Info			Dead Space (%)			Run Time (s)		
Instance Name	# Blocks	# Islands	Previous Work [12]	Proposed Floorplanner	Savings (%)	Previous Work [12]	Proposed Floorplanner	Savings (%)
n10	10	2	1.71	1.3	23.9	1.17	0.66	43.58
n10	10	3	1.31	0.9	31.2	0.92	0.6	34.78
n30	30	2	4.43	2.6	41.3	10.03	5.16	48.55
n30	30	3	3.55	3.69	-3.9	11.92	5.36	55.03
n50	50	2	4.5	3.2	28.88	30.54	22.38	26.71
n50	50	3	2.8	3.1	-10.71	32.38	28.69	11.39
n100	100	2	5.71	4.8	15.99	134	87	35.07
n100	100	3	5.65	5.16	8.59	124	89	28.22
n100	100	4	6.47	5.3	18.1	136	91	33.08
n200	200	2	5.22	4.4	15.87	603	441	26.86
n200	200	3	6.83	6.59	3.51	654	403	38.37
n200	200	4	8.58	8.08	5.82	637	407	36.10
n300	300	2	8.62	8.39	2.66	585	431	25.98
n300	300	3	9.02	8.5	5.76	587	432	26.4
n300	300	4	10.29	9.37	8.9	591	433	26.73
Avg					**13.5**			**34**

TABLE II
AREA, WIRELENGTH AND POWER TRADEOFF

Circuit Info			$\alpha = 0.75$ $\beta = 0.5$			$\alpha = 0.25$ $\beta = 0.5$			$\alpha = 0.5$ $\beta = 0.75$		
Instance Name	# Blocks	# Islands	Area	Wirelength (x10)	Power	Area	Wirelength (x10)	Power	Area	Wirelength (x10)	Power
n10	10	2	228448	19699	1561	227098	20197	1561	235438	20186	1583
n30	30	3	217145	55609	4341	219294	59305	4520	221057	56026	4907
n30	30	4	227760	57591	4480	227052	61070	4507	224896	56742	4523
n50	50	3	207690	107220	7798	213900	105715	7790	216204	105681	7900
n50	50	4	214110	105445	7101	210312	99275	7144	212528	105469	7546
n100	100	2	191760	160495	15717	197550	149583	15651	196612	151094	16125
n100	100	3	201928	170515	14682	193230	174178	15027	196272	159843	15293
n100	100	4	196184	175556	15966	197540	171559	15282	201928	165641	15940
n200	200	3	198712	353394	29755	204670	355220	29582	201930	343418	30397
n200	200	4	191260	375406	29574	201488	369460	29537	202950	345094	31895

need to be optimized. Thus integrating voltage assignment, island partitioning and floorplanning together produce higher quality results than that was obtained by performing these steps separately.

VI. CONCLUSION

In this paper, we have proposed a new simulated annealing-based voltage island-aware floorplanning algorithm that integrates the voltage assignment and floorplanning processes into a single design step. Constraints are imposed on the sequence pair representation in order to realize island-based floorplanning. The cost function used by the floorplanner takes into account the power, area and wirelength of candidate solutions. We showed that our algorithm reduces dead space by a further 13.5% and with 34% runtime reduction, compared to previous work. Additionally we have explored different floorplan solution based on the relative weights of power and area and showed the necessity of using both power and area as the two objectives in modeling the cost function.

REFERENCES

[1] D. Lackey, P. Zuchowski, T. Bednar, S. Stout, D. Gould, and J. Cohn, "Managing power and performance for system-on-chip designs using Voltage Islands," in *Int'l Conf. on Comp. Design*, 2002, pp. 195–202.

[2] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu, "Architecting voltage islands in core-based system-on-a-chip designs," in *Int'l Symposium on Low Power Electronics and Design*, 2004, pp. 180–185.

[3] W. L. Hung, G. Y. X. Vijaykrishnan, N. Dhanwadaf, and J. N. Conner, "Temperature-aware voltage islands architecting in system-on-chip design," in *Int'l Conf. on Comp. Design*, 2005, pp. 689–694.

[4] L. Wan-Ping, L. Hung-Yi, and C. Yap-Wen, "Voltage Island Aware Floorplanning for Power and Timing Optimization," in *Int'l Conf. on CAD*, 2006, pp. 389–394.

[5] L. Bin, Z. Q. C. Yici, and H. Xianlong, "Power driven placement with layout aware supply voltage assignment for voltage island generation in dual-Vdd designs," in *ASP Design Automation Conf.*, 2006, pp. 582–587.

[6] R. Ching, E. Young, K. Leung, and C. Chu, "Post-placement voltage island generation," in *Int'l Conf. on CAD*, 2006, pp. 641–646.

[7] W. Mak and J. Chen, "Voltage Island Generation under Performance Requirement for SoC Designs," in *ASP Design Automation Conf.*, 2007, pp. 798–803.

[8] H. Wu, Y. Wang, and M. Wong, "Post-placement voltage island generation under performance requirement," in *Int'l Conf. on CAD*, 2005, pp. 309–316.

[9] W. Huaizhi, M. Wona, and L. I-Min, "Timing-constrained and voltage-island-aware voltage assignment," in *Design Automation Conf.*, 2006, pp. 429–432.

[10] S. Adya and I. Markov, "Fixed-outline Floorplanning: Enabling Hierarchical Design," *IEEE Trans. on VLSI Systems*, vol. 11, no. 6, pp. 1120–1135, 2003.

[11] A.B.Kahn, "Classical floorplanning harmful?" in *Int'l Symp. on Physical Design*, 2000, pp. 207–213.

[12] Q. Ma and E. Young, "Multivoltage Floorplan Design," *IEEE Trans. on CAD*, vol. 29, no. 4, pp. 607–617, 2010.

[13] R. Otten, "Automatic Floorplan Design," in *Design Automation Conf.*, 1982, pp. 261–267.

[14] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani, "VLSI module placement based on rectangle-packing by the sequence pair," *IEEE Trans. on CAD*, vol. 15, no. 12, pp. 1518–1524, 1996.

[15] X. Tang, R. Tian, and D. F. Wong, "Fast Evaluation of Sequence Pair in Block Placement by Longest Common Subsequence Computation," in *Design, Automation and Test in Europe*, 2000, pp. 106–111.

[16] X. Tang and D. F. Wong, "Floorplanning with alignment and performance constraints," in *Design Automation Conf.*, 2002, pp. 848–853.

[17] M.-C. Wu, M.-C. Lu, H.-M. Chen, and J.-Y. Jou, "Performance-constrained voltage assignment in multiple supply voltage SoC floorplanning," *ACM Transactions on Design Automation of Electronic Systems*, vol. 15, no. 1, pp. 9–15, 2009.

[18] D. Sengupta and R. Saleh, "Application-driven Floorplan-aware Voltage Island Design," in *Design Automation Conf.*, 2008, pp. 155–160.

[19] N. Sherwani, *Algorithms for VLSI Design Automation, Third Edition*. Kluwer Academic Publishers, 1999.