C-010

Power Consideration Multilevel Partitioning Using Voltage Islands

Wang Wei† Lin Tao† Watanabe Takahiro†

1. Introduction

With the broaden market interests, meeting the critical power consumption is more and more important in the SoC design. Under this condition multi-supply voltage (MSV) has been proposed, which is an effective way to achieve the low power. Voltage island is a group of contiguous on-chip cores powered by the same voltage level. In the following Fig.1(a), we can see that the circuit without voltage islands are powered by 1.5v from C1 to C5. However, in the Fig.1(b), C1, C2, C4, C5 are powered by 1.2v and C3 is powered by 1.5v.

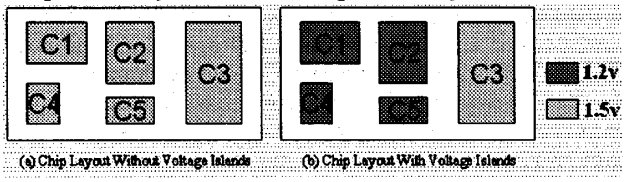

Fig.1 Chip Layout

SoC circuits can be represented as hypergraphs, but the optimum bisection of a hypergraph is an NP-hard problem, and we can not get the precise partition with 50% to 50%. Multilevel hypergraph [2] is the most efficient way to calculate it in the current research.

Our research will show a connection between voltage islands and multilevel hypergraph partitioning, realizing low power consumption.

2. Features of our proposed method

There are three main features in our proposed multilevel hypergraph partitioning considering voltage islands, as follows. The first feature is that our method incorporates a well-known multilevel partitioning package 'hMetis' [3], because it can efficiently generate a good partitioning based on Min-cut algorithm. The second is that voltage islands are considered to achieve low power consumption in circuit partitioning process at the same time. The third is that there exists a tradeoff among objective functions by the min-cut partitioning and low power consumption. The three basic processes of hMetis, that is, coarsening phase, initial partitioning and uncoarsening and refinement phase, are applied adjusting to our model.

For example, power consumption is taken into account using voltage island information at a coarsening phase, initial partitioning phase and uncoarsening and refinement phase. Using these features, a min-cut based partitioning with low power consumption can be executed.

3. Outline of a partitioning process

Fig.2 shows a flow of a multilevel partitioning considering voltage island information. Given a circuit data and voltage island information, all data are set up to execute partitioning process and a hash table structure is generated to improve the processing time for partitioning. Then coarsening phase, initial partitioning phase and the uncoarsening and refinement phase are executed considering voltage islands. Those multilevel partitioning phases are based on hMetis with some modifications to deal with voltage island information.

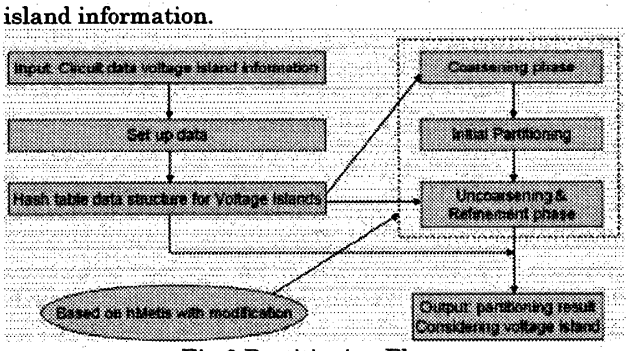

Fig.2 Partitioning Flow

3.1 Coarsening Phase

During the coarsening phase, a sequence of smaller hypergraphs ('layer') is constructed by merging vertices. This coarsening phase is somewhat different from the phase in hMetis, that is, the factor of voltage islands must be considered. First all vertices are classified into two groups, one group is composed of high voltage vertices (marked 'H') and another is of low voltage vertices ('L'). Then the weights of all vertices are calculated and they are sorting in a descending order of weights. Here, weight of a vertex (V) is defined inEq.(1). Let a vertex vertex (Va) be the first in the sorted list. We want to merge (Va) with another vertex (Vb) to make a small-sized hypergraph. The vertex (Vb) is chosen by searching the list from the second element to the last under the constrained defined in Eq.(2).

$$Weight(V) = \sum_{V \in H(i)} \frac{1}{NumberofVertices(H(i))} \quad (1)$$

$$Area(Va) + Area(Vb) \le \frac{TotalArea}{NumberofVertices/2} \quad (2)$$

Merging results a new vertex in a new layer in a coarser level and this new vertex is newly marked 'H' or 'L'. However, sometimes a vertex after merging may include high and low voltage vertices. In that case, a new mark 'HL' is used. It means that the vertex is generated by merging 'H' vertex and 'L' vertex. In a new sorted list, 'HL' vertex is inserted between 'H' and 'L' vertices. Next we will choose the first vertex in the current list and execute the same process repeatedly until all the vertices are marked in the current list. Finally a new layer is constructed.

Layers are generated again and again until the number of vertices in the current layer is smaller than a threshold called 'Vterminal' which is usually a value several hundreds.

3.2 Initial Partitioning

The preceding coarsening phase yields a hypergraph with much less vertices. During the initial partitioning phase in hMetis, a bisection of the coarsest hypergraph is computed under the constraint of 'UBfactor' which an upper bound of the difference between areas of two groups partitioned in min-cut process. However, our partitioning aims at not only min-cut but also less power consumption. Therefore, besides on smaller cut,

† Graduation School of IPS, Waseda University

it is preferable that all high voltage vertices marked 'H' are gathered in one area after bisection. And if the total area satisfies UBfactor, then vertices marked 'HL' and vertices marked 'L' are allocated in the same area until the area becomes larger than half of the total area. After bisection, refinement by FM algorithm is executed to improve the result of this bisection.

3.3. Uncoarsening and Refinement Phase

Uncoarsening and refinement phase is a reverse process of coarsening phase. Smaller hypergraphs are reversed to the original hypergraphs layer by layer. This process is also different from the hMetis's uncoarsening and refinement phase, especially FM algorithm. In the traditional FM refinement, gains defined in Eq.(3) are first calculated for all the vertices, where N(Vin) means the number of vertices in the group including V and connecting with V, and N(Vout) means the number of vertices in the other group and connecting with V.

$$gain(V) = N(Vout) - N(Vin) \quad (3)$$

Next, a vertex having the largest gain is selected from each of the two groups. If those two gains are positive and the area after swapping satisfies the UBfactor, they are swapped and labeled. This swapping process is repeated until there is no positive gain in both groups. Our uncoarsening and refinement phase is different. The priority and scope of selecting two vertices from each group is determined and ordered as shown in Fig.3.

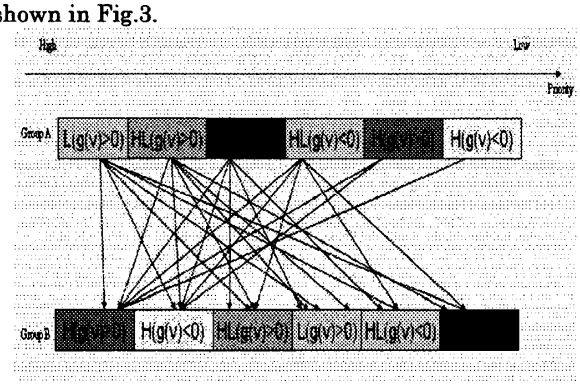

Fig.3 Priority and select scope in FM

Group A contains most of the high voltage vertices and a vertex is chosen from Group A first according to the priority from high to low. The other vertex from Group B is also selected in the same way. But we should pay an attention here that the selected vertex from group B has a scope to match the selecting vertex from Group A. For example when vertex with H(g(V)>0) from group A is selected, the scope of selecting the other vertex from Group B is only H(g(V)>0) and H(g(V)<0).

3.4 Hash table data structure

In the coarsening and uncoarsening phase, weights and gains of vertices should be calculated, but this calculation is time-consuming. Especially FM refinement during the uncoarsening phase takes most of the total computation time. When a vertex is merged with another vertex, searching the other vertex takes a lot of time if the sequential search is used. So a hash table data structure is applied. This structure for voltage islands including the information of weight is shown in Fig.4.

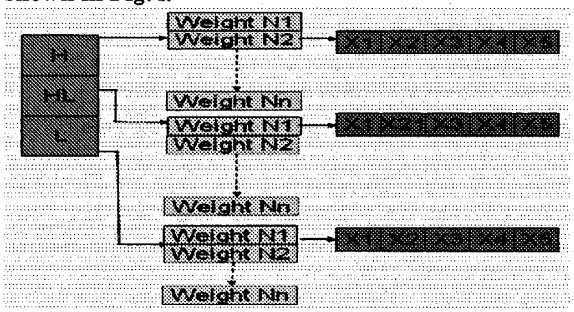

Fig.4 hash table data structure for voltage islands

4. Simulations and Results

ISPD98 benchmark data was used to evaluate our method, but the information of voltage islands is not included. So some data were appended or changed. For example, areas which are assigned with value 0 are changed to 1. High or low voltage are also assigned to each vertex in a circuit data, where the ratio of high voltage vertices are conveniently set to 20% for all the vertices. C# language here is used to realize our method under the experiment environment of Intel Core2 E6700 and 2 GB memory.

After data setting, k-way partitioning was experimented (k=2, 4, 8). Some results of 2-way partitioning are shown in the following Tables.

Tab.1 Computation time (2-way & voltage islands)

Data File	Total average Time(s)	Coarsen AvergeTime(s)	Initial Partition Average Time(s)	Uncoarsen&Refinement AverageTime(s)
ibm02	268.1	02.3	00.0	265.8
ibm05	631.5	03.9	00.0	627.5
ibm07	613.9	06.8	00.0	607.1
ibm09	958.1	09.9	00.1	948.1
ibm10	768.6	11.6	00.1	756.8
ibm14	1340.3	23.5	00.2	1316.6
ibm15	1291.4	22.6	00.2	1268.5
ibm18	2902.3	27.6	00.3	2874.4

Tab.1 and Tab.2 show the information of computation time of 2-way partitioning separately considering voltage islands and not considering voltage islands. We can see that the uncoarsening and refinement phase takes most of the computation time. When considering voltage islands, it takes long time.

Tab.2 Computation time (2-way & no voltage islands)

Data File	Total Aaverage Time(s)	Coarsen AvergeTime(s)	Initial Partition Average Time(s)	Uncoarsen&Refinement AverageTime(s)
ibm02	32.1	01.2	00.0	31.8
ibm05	115.0	03.8	00.2	113.5
ibm07	58.5	06.7	00.1	56.3
ibm09	63.9	04.0	00.1	62.5
ibm10	55.2	02.5	00.1	54.4
ibm14	108.3	05.0	00.3	106.4
ibm15	105.4	21.5	00.4	98.1
ibm18	200.4	21.3	00.6	192.9

Tab.3 Voltage allocation (2-ways & voltage islands)

Data File	Total Area	Total Group1 Area	Group1 High Voltage Area	Group1 Low Voltage Area	Total Group2 Area	Group2 High Voltage Area	Group2 Low Voltage Area
ibm02	8458595	4417282	1691774	2725508	4041313	0	4041313
ibm05	4472721	2220115	894790	1325325	2252606	0	2252606
ibm07	11830143	5578060	2366078	3211982	6252083	0	6252083
ibm09	17529597	8345294	3505947	4839347	9184303	0	9184303
ibm10	47535080	25955057	9507149	16447908	21580023	0	21580023
ibm14	28727813	11850240	5745639	6104601	16877573	0	16877573
ibm15	36535327	18769744	5753675	13016069	17765583	0	17765583
ibm18	33686832	15651524	6737395	8914129	18035308	0	18035308

Tab.4 Voltage allocation (2-ways & no voltage islands)

Data File	Total Area	Total Group1 area	Group1 High Voltage Area	Group1 Low Voltage Area	Total Group2 area	Group2 High Voltage Area	Group2 Low Voltage Area
ibm02	8458595	3908411	1054123	2854288	4550184	637651	3912533
ibm05	4472721	2189206	622401	1566805	2283515	272389	2011126
ibm07	11830143	6496626	1159311	5337315	5333517	1206767	4126750
ibm09	17529597	9636588	1661222	7975366	7893009	1844725	6048284
ibm10	47535080	25446277	2661324	22784953	22088803	6845825	15242978
ibm14	28727813	15231061	3254891	11976170	13496752	2490748	11006004
ibm15	36535327	20020928	4391584	15629344	16514399	1362091	15152308
ibm18	33686832	16120295	3144480	12975815	17566537	3592915	13973622

Tab.5 Min-cut (2-way)

Data File	2-way Partition without considering Voltage Islands	2-way Partition considering Voltage Islands
ibm02	1934	5021
ibm05	2123	6264
ibm07	3093	7492
ibm09	5949	11905
ibm10	8533	16372
ibm14	12558	26602
ibm15	17635	37054
ibm18	19750	32946

Tab.3 and Tab.4 have the information of voltage allocation after 2-way partitioning considering voltage islands and not considering voltage islands. We can see it clearly that in Tab.3, all the high voltage vertices are allocated into group1 and the high voltage areas of group2 are all 0, which forms the voltage islands. However, high voltage vertices are allocated in group1 and group2 randomly in Tab.4 without considering voltage islands. Our algorithm considering voltage islands consumes less power than that of not considering voltage islands obviously.

In Tab.5, the information of min-cut after 2-way partitioning separately considering voltage islands and not considering voltage islands is shown, from which we can see the min-cut considering voltage islands is larger than the occasion not considering voltage islands.

5. Conclusions and Future work

This paper proposes a multilevel partitioning considering voltage islands. Experimental results show that a multilevel partitioning with power consideration can be obtained by our method, while computation time and min-cut is larger compared with a partitioning method without voltage islands information, however, our method consumes less power. Faster and higher quality partitioning algorithm considering voltage islands is a future work.

Reference:

[1] Shengqi Yang, Wayne Wolf, N. Vijaykrishnan and Yuan Xie: "Reliability-Aware SOC Voltage Islands Partition and Floorplan", Proceedings of ISVLSI'06, pp.343-348, 2006

[2] George Karypis, Rajat Aggarwal, Vipin Kumar and Shashi Shekhar: "Multilevel Hypergraph Partitioning Applications in VLSI Domain", Proceedings of the Design and Automation Conferencce, pp.69-79, 1997

[3] http://glaros.dct.umn.edu/gkhome/metis/hmetis/